A Novel Release Controller for Robot to Human Object Handover via Combined Joint Torque and Vision-based Sensing

by

Mohammadhadi Mohandes

B.Sc. (Mechanical Engineering), Sharif University of Technology, 2019

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Applied Science

in the
School of Engineering Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Summer 2023

Declaration of Committee

Name:	**Mohammadhadi Mohandes**
Degree:	**Master of Applied Science**
Thesis title:	**A Novel Release Controller for Robot to Human Object Handover via Combined Joint Torque and Vision-based Sensing**
Committee:	**Chair:** Michael Hegedus Lecturer, Engineering Science **Kamal Gupta** Co-Supervisor Professor, Engineering Science **Mehran Mehrandezh** Co-Supervisor Adjunct Professor, Engineering Science **Mehrdad Moallem** Committee Member Professor, Mechatronic Systems Engineering **Siamak Arzanpour** Examiner Associate Professor, Mechatronic Systems Engineering

Ethics Statement

The author, whose name appears on the title page of this work, has obtained, for the research described in this work, either:

a. human research ethics approval from the Simon Fraser University Office of Research Ethics

or

b. advance approval of the animal care protocol from the University Animal Care Committee of Simon Fraser University

or has conducted the research

c. as a co-investigator, collaborator, or research assistant in a research project approved in advance.

A copy of the approval letter has been filed with the Theses Office of the University Library at the time of submission of this thesis or project.

The original application for approval and letter of approval are filed with the relevant offices. Inquiries may be directed to those authorities.

Simon Fraser University Library
Burnaby, British Columbia, Canada

Update Spring 2016

Abstract

Our research focuses on the physical exchange phase in the robot-to-human object handover task. We present a novel torque and vision sensor-based release controller for the physical exchange phase. Our system is implemented on a 7-DOF manipulator Kinova arm with joint torque sensors and equipped with an eye-in-hand camera and a 3-finger mechanical Schunk Dextrous Hand. The system performs a fully autonomous and robust object handover to a human receiver in real-time. Our control algorithm relies on two complementary sensor modalities: joint torque sensors on the arm and an eye-in-hand RGB-D camera for vision sensor feedback. Our approach is entirely implicit, i.e., there is no explicit communication between the robot and the human receiver. Information obtained via the aforementioned sensor modalities is used as inputs to their respective deep neural networks. While the torque sensor network detects the human receiver's action, such as: pull, hold, or bump, the vision sensor network detects if the receiver's fingers have wrapped around the object. Networks' outputs are then fused, based on which a decision is made to either release the object or not. Our release controller is then compared to an existing handover controller, which performs handover using only the force sensor on the wrist. Our approach overcomes substantive challenges in sensor feedback synchronization, object and human hand detection, and achieves robust robot-to-human handover with 98% accuracy in our real experiments with human receivers.

Keywords: Robot to Human Object Handover; Object Detection; Torque Sensors; eye-in-hand camera

Dedication

I dedicate my M.Sc. thesis to all those innocent lives around the world who fought for freedom, justice and equality. I want to honour the innocent lives of people in woman, life, freedom revolution in Iran. I also dedicate this work to my friend, Mehdi Eshaghian, who was in the PS752 tragedy. In the end, I want to dedicate my work to my family, Mom, Dad, my two brothers, and my lovely little sister, whose support and help have been there with me my whole life.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisors, Professor Kamal Gupta and Professor Mehran Mehrandezh, whose unwavering support, guidance, and expertise have helped me during this journey. Their dedication to academic excellence, tireless commitment to my growth, and invaluable insights have greatly enriched my research experience.

I want to thank my dear lab mate, Mr. Behnam Moradi, whose support and knowledge were a great deal in this thesis.

I also would like to express my appreciation to my examiner for taking the time to review my thesis and provide his valuable feedback. Additionally, I acknowledge that this thesis has been proofread with the permission of my supervisors, ensuring that it meets the highest standards of academic excellence.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Robots are becoming increasingly integrated into our daily lives, performing tasks that range from manufacturing and logistics to healthcare and domestic assistance. The direct interaction between humans and robots has become a core topic of interest in the field of robotics. Human-Robot interactions have faced a great improvement in the sense of both hardware and software aspects, which will lead the robots to a better perception of their surrounding environment and make more efficient interactions with humans. In the industry, robots apply actions on and do interactions with other robots and humans [3][10]. Previously, Robots used to have minimal interactions with humans and mainly would get used in highly structured environments, and they would work based on pre-programmed algorithms that did not have to be as intelligent as they should be in comparison to interaction with humans. However, there has been a demand to develop more intelligent algorithms that can be deployed on robots to enable them to work beside people [50].

Cooperative robots have been useful for improving workspace productivity. Although human beings are better at decision-making and dexterous manipulation of objects, robots can perform countless numbers of preferred actions over and over again. Human-robot teams include more employment of human workers in intelligent and cognitive aspects while the robots are being used in more labour, low skill and undesirable hard work.

One of the critical capabilities in this field is the development of Robot to Human (R2H) object handover techniques. The ability of robots to hand over objects to humans seamlessly can significantly enhance their usefulness in various applications, such as manufacturing, healthcare, and home assistance.

Despite the large number of works done in this field, there are still substantial problems that need to be solved. The main challenge in robot-to-human object handover is the lack of implicit communication from the human to the robot for the robot's decision-making. Uncertainty regarding the activities of humans throughout the handover process becomes one of the most significant obstacles. Unanticipated movement on the part of humans can result in objects being dropped or otherwise damaged. In addition, people have a variety of hand postures and grips that they employ when they are trying to get a hold of something, which makes it difficult for robots to determine the best handover approach [50].

To address these challenges, recent studies have separated the object handover process into three main phases and have explored each phase separately. Figure 1.1 represents the 3 phases during R2H object handover. Moreover, they have concentrated on creating techniques for robots to hand over objects to humans while dealing with the variability and unpredictability of human actions. These techniques involve incorporating multiple sensors, like cameras and force sensors, to identify human movements and modify the robot's handover method accordingly. Furthermore, machine learning algorithms are being employed to enable robots to learn from prior experiences and refine their handover precision [50] [12].

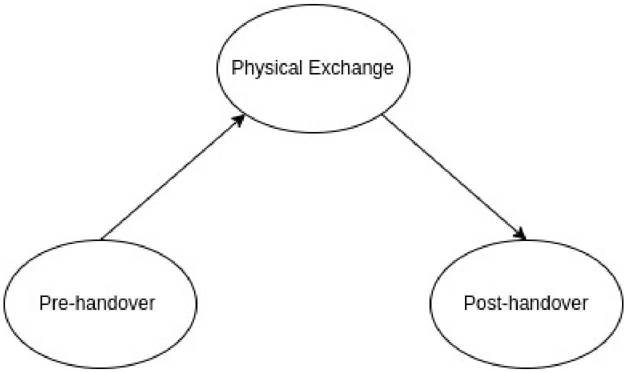


Figure 1.1: Three different phases in Robot-to-Human object handover. Our main focus is on the physical exchange phase.

This thesis aims to investigate and evaluate the current state-of-the-art R2H object handover techniques in the middle phase, the *Physical Exchange* phase. We examine the challenges and limitations of existing techniques and propose novel approaches to address these challenges. The proposed approach makes use of sensor modalities, torque and vision, on our robotic handover system to perform a safe and robust handover. The thesis will also evaluate the effectiveness and efficiency of the proposed techniques through experiments.

1.1 Problem Description

A long-term objective of the Robotic And Motion Planning Laboratory (RAMP Lab) group is autonomous planning and execution of a fetch and handover task with a mobile manipulator, i.e., it will autonomously navigate across hallways and rooms using its onboard sensors, will avoid colliding with obstacles (both the mobile base and the manipulator's arm will avoid obstacles), both static ones such as furniture and dynamic ones such as humans walking around, thus ensuring motion safety, both of the humans around it and itself. It will plan its view, i.e., where to look, e.g., to grasp an object, say a medicine bottle occluded behind a computer on a table and hand it over to a human user. The core sub-problems in this overall task involve process motion planning, grasping and robot-to-human object handover. Previous researchers in the RAMP lab have worked on motion planning of mobile manipulators, e.g., Pilania and Gupta [56], autonomous grasping, e.g., Hegedus et al. [24]. This thesis addresses a component of the next core sub-problem, i.e., Robot to Human (R2H) object handover, in particular, the physical exchange part of the R2H handover, as elaborated in the next paragraph below.

The main objective of our research is to reduce failure and automate the process of R2H object handover tasks. An R2H system consists of 2 main participants in action, which are *Giver* and *Receiver* and an *Object*, which will be exchanged during the process. In the robot-to-human object handover, the giver is the robot and the receiver is the human. Figure 1.2 depicts our robot-to-human object handover system in the RAMP lab and is described in the following section.

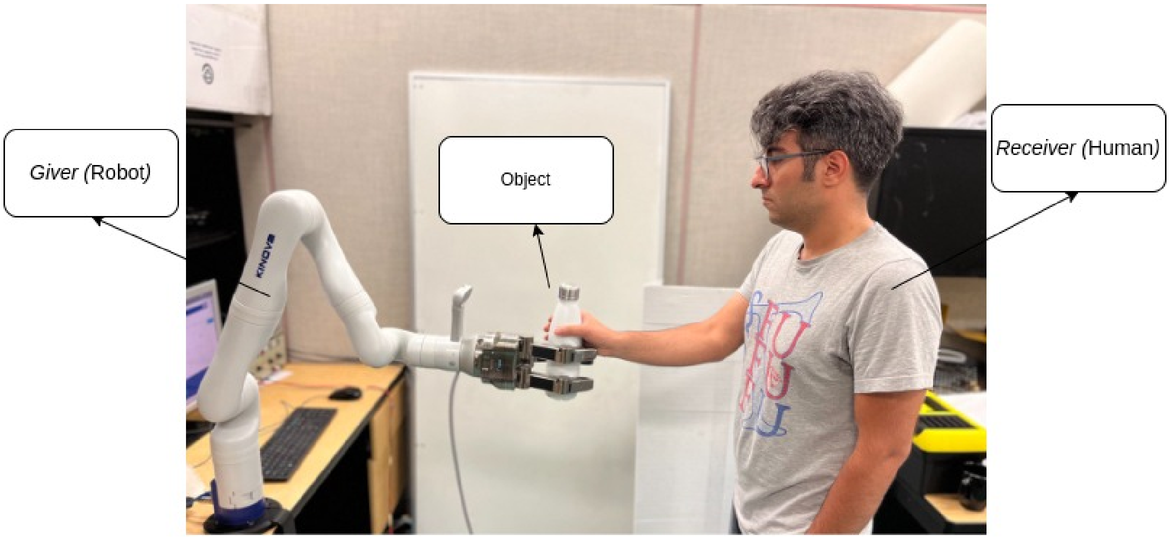


Figure 1.2: R2H object handover system containing giver, receiver, and the object.

In our R2H handover scenario, the robot, *Giver*, will grasp an object and will continue holding on to it as long as there isn't any physical (haptic) interaction initiated by the human *Receiver*. Once the *Receiver* initiates the process of receiving the object by grasping it and applying a force toward him/herself, the robot *Giver* will make a decision on whether or not to release the object. In different situations, the receiver may accidentally apply unwanted actions on the object, which may cause some wrong interpretations by the giver and may result in dropping the object and failure. Moreover, the *Receiver* may not fully grasp the object, which may cause a false release by the giver.

1.2 R2H Handover System Test-bed

Our system hardware [31] [65] shown in Figure 1.3 consists of a 7-DOF robotic manipulator (Kinova Gen3 arm), a 7-DOF 3-fingered Schunk Dexterous Hand (SDH).

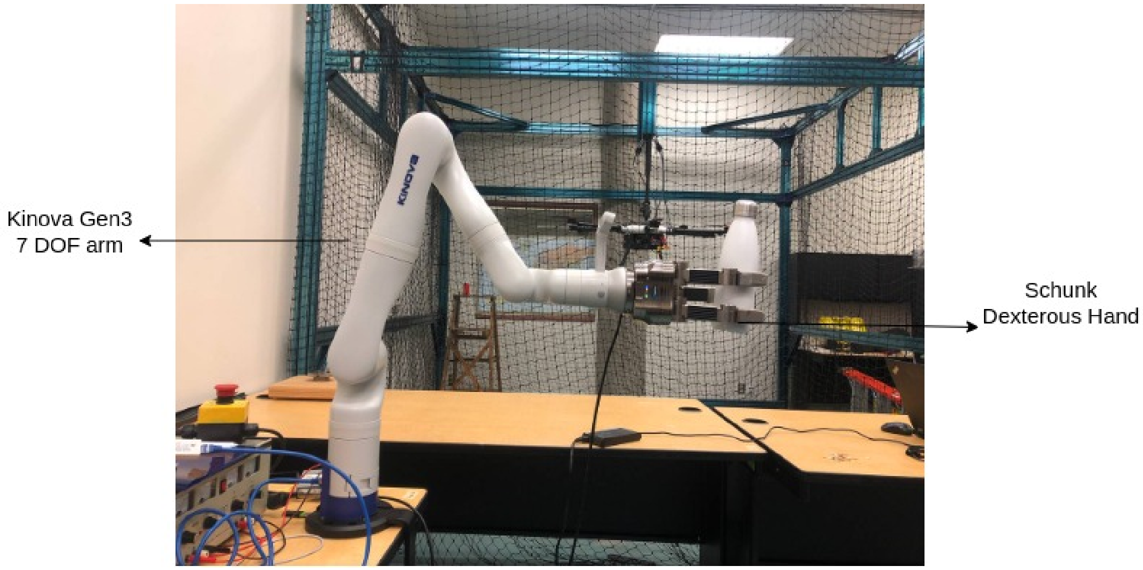


Figure 1.3: Kinova gen3 robotic arm and Schunk Dexterous gripper in handover configuration.

1.2.1 Kinova Gen3 7 DOF Robotic Arm

There are seven rotary actuators in the Kinova arm and each actuator comes with a torque-sensor and is equipped with a 100:1 strain wave gear to endow the arm with smooth motion. There are two different sizes of actuators, including small, which can tolerate up to 34 N.m, and large, which can bear up to 54 N.m.

Kinova, on our request, added a custom-designed interface module such that the SDH gripper, described in the next subsection, could be mounted on the arm. Figure 1.4 shows a detailed view of the test bed, torque sensors, and vision sensor.

The arm is also equipped with a wrist-mounted camera - an Omnivision OV5640 colour sensor along with Intel® RealSense™ Depth Module D410 stereo depth sensor. More specifically, the camera is mounted on the top part of the robot's end-effector and is able to capture and stream image data in the camera's field of view. The colour sensor publishes a 2D array of RGB data which is gathered from the perspective of the sensor's field of view. The depth sensor captures a 2D array, including the depth and position of each pixel. Figure 1.5 shows RGB and depth information from the camera during the R2H handover. For more information on the RGB-D data registration formulation, please refer to Appendix A.

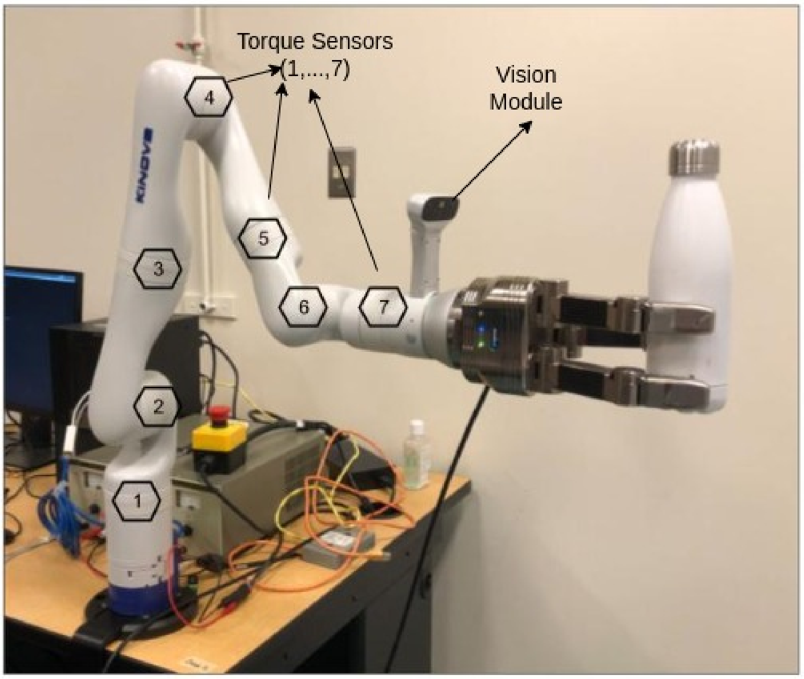
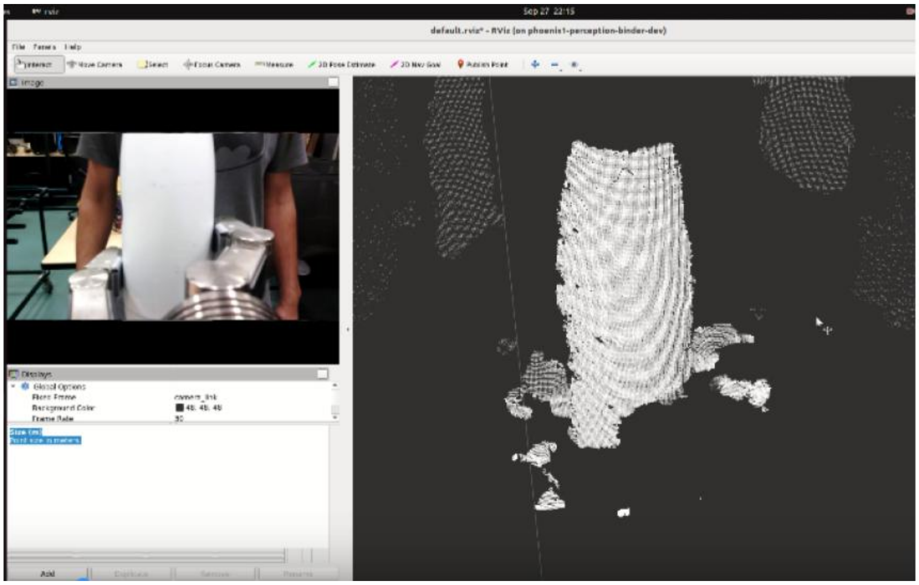


Figure 1.4: Torque sensors and vision module on Kinova gen3 robotic arm.



Fixed Frame camera_link
Background Color 48; 48; 48
Frame Rate 30

Size (m)
Point size in meters.

Figure 1.5: RGB and depth information from camera's field of view.

1.2.2 Schunk Dexterous Hand

Schunk Dexterous Hand (SDH) can be used for a wide range of tasks requiring a firm grasp. The SDH is mounted on the end-effector of the Kinova arm to perform the grasping and releasing in the handover task. This gripper includes three fingers, each of which has 2 degrees of freedom and the three fingers have an independent rotational DOF in the palm, summing up to 7 DOF total. One joint in each of the three fingers is fixed to the palm of the hand, while the other is located in the finger's middle. Each finger consists of proximal and distal parts, and each part is equipped with a tactile sensor pad [65]. Figure 1.6 shows the SDH joints and tactile sensors. A more detailed description of the hand is given in Section 3.1 when we explain the precise implementation of our algorithms.

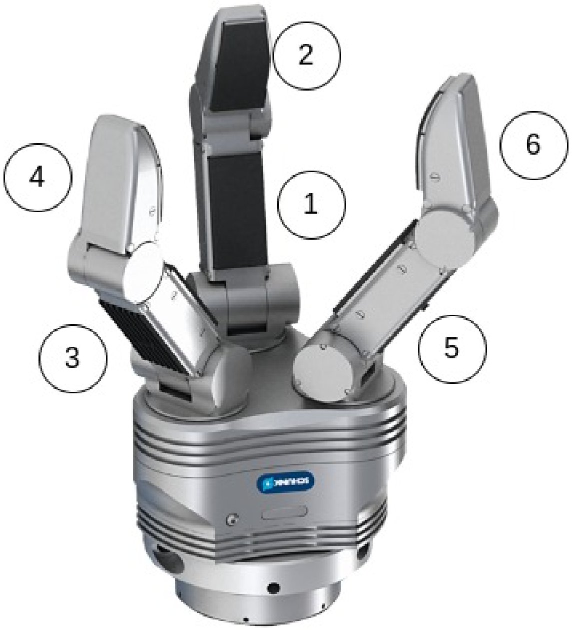


Figure 1.6: Distal (even numbers) and proximal (odd numbers) tactile pads on SDH gripper [2].

The precise form of the three different sensors' data from joint torque sensors, the eye-in-hand camera, and tactile sensors will be depicted later in Chapter 3 when the algorithmic implementation is explained.

1.3 System Overview

A State diagram of our proposed R2H object handover system is shown in figure 1.7. In this figure, the first node represents the state in which the object is grasped by the giver. The robot grasps the object and maintains the grasp until the **receiver's action** is detected. The Proper action from the giver is to **release** or **not release** the object. The next state would be accordingly **object released** or **object grasped**.

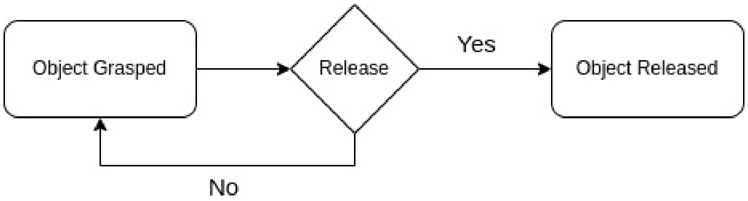


Figure 1.7: State diagram of R2H handover during physical exchange.

A key idea in our work is to detect different types of actions applied by the receiver on the object while it is grasped in the giver's hand and also detect the fingers of a human receiver wrapped around the object. This way, the giver only releases the object when a proper action and proper grasping is applied to the object by the receiver.

The methodology shown in figure 1.7 is represented using a state machine in figure 1.8. The robot is initialized and starts moving to the handover configuration to be ready to grasp the object. SDH grasp controller, relying on the tactile sensors on the fingers, takes the role of grasping the object with a certain amount of force which is maintained via a PID controller. As soon as a human, *receiver*, applies an action on the object, the vision sensor data and the torque sensor data will be captured for further analysis. The Vision Processing package, Shown as SSD (Single Shot Detector) inference package in the block diagram, relying on the vision data, determines whether the action is applied with fingers wrapped around the object or not.[1] Meanwhile, the Torque Processing package, Shown as CNN (Convolutional Neural Network) Inference package in the block diagram, relying on the joint-level torque data, has the role of detecting if the proper action is being applied to the object. The results of these two networks will be fed into the conditional statement, and a decision on the release will be made. Terms and Methods used in our algorithm, such as SSD and CNN, Will be later explained in Chapter 2, and different sensor data will be explained in Chapter 3.

Having described our system in a broad brush, we now review the literature in the field.

[1] The vision processing package was developed by my labmate Mr. Behnam Moradi. His contribution is gratefully acknowledged.

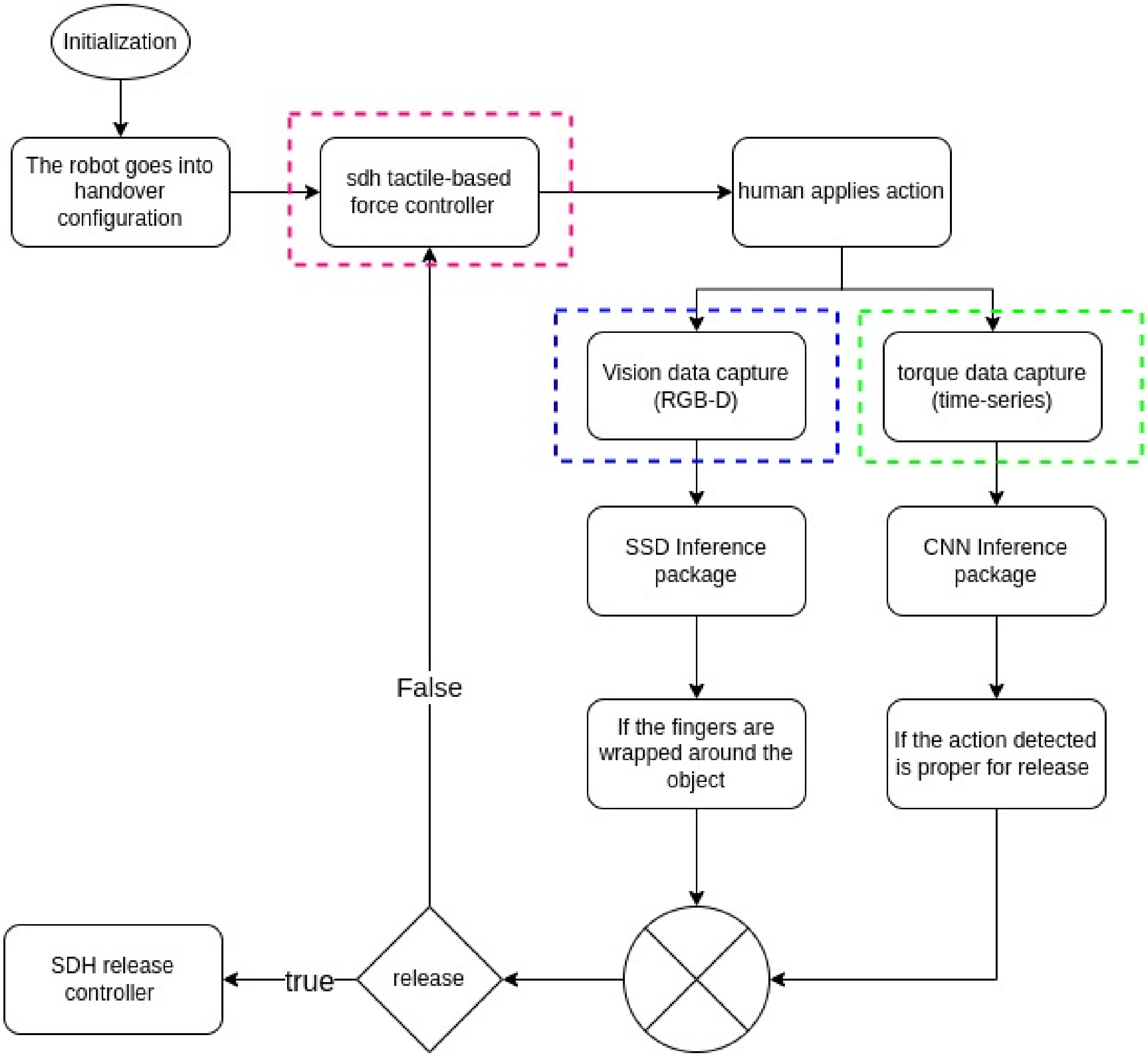

Initialization
The robot goes into handover configuration
sdh tactile-based force controller
human applies action
Vision data capture (RGB-D)
torque data capture (time-series)
SSD Inference package
CNN Inference package
If the fingers are wrapped around the object
If the action detected is proper for release
False
release
true
SDH release controller

Figure 1.8: Detailed R2H handover system overview. Different sensor modalities are indicated by a differently colored dashed box.

1.4 Background

Robots have entered people's lives in their homes. With the high interest of humans in facilitating the difficulties in everyday activity, the research community has begun to implement different capabilities in robots. Most of these applications are developed to transfer different objects from a given origin to a destination. As a result, server robots are being used in hospitals to take off some of the burden on the nurses' work. Since the nurses' workload increases during the hospital rush hour, these robots use smart navigation to deliver the necessary medications to patients. This is also useful when the patients carry a contagious disease, and the use of human nurses will increase the risk level of getting exposed to that disease. As mentioned before, a high percentage of robot applications is in human-robot interactions. In comparison to pre-defined actions implemented on robotic platforms used in the industry, one of the challenges that arise when exploring human-robot interactions is dealing with uncertainties and acting intelligently considering the human's unpredictable actions.

1.5 Object Handover

Transferring an object from one agent to another agent is known as "Object Handover". The whole system consists of three main parts. The "giver" is the term used to refer to the agent who hands over the "object" to the "receiver" which refers to the agent who receives the object from the giver. To achieve a successful handover, both the giver and receiver cooperate with each other. The giver visually locates the receiver and plans the transfer route. The receiver waits for the object to reach a suitable location to take control. The giver also uses haptic sensing during this phase. If the receiver struggles to gain control and the risk of dropping increases, the giver adjusts their actions to prioritize object safety [43].

In the literature, three distinct phases of object handover have been identified: the pre-handover phase, the physical exchange phase, and the post-handover phase [50]. Researchers have explored these phases in different scenarios such as Human-to-Human handover (H2H), Human-to-Robot handover (H2R), Robot-to-Robot handover (R2R), and finally Robot-to-Human (R2H) handover. Our work focuses on the physical exchange phase of R2H handover.

H2H handover characteristics of the giver and the receiver have inspired some interesting works in autonomous object handovers [50]. Authors in [43] examined grip forces in giver and receiver and in [12], the giver's and receiver's load force and grip force on the object are measured during handover and it is shown that the receiver's grip force is proportional to the load force on the object and the giver's grip force has an inverse ratio to the object load sensed on giver's side. Another H2H handover exploration has been established by [54] and [55] which shows that people, especially the giver, relies on vision to fulfill a safe object handover process. In the absence of vision, failure happens in 33% of the experiments. Also, an investigation was conducted by Basili et al.[6] and Shibata et al. [66] on how the giver reaches to the receiver and does trajectory planning for transferring the object in H2H handovers in the pre-handover phase.

There are other characters from H2H handovers, such as gaze, which would be useful for H2R and R2R handovers. Gaze is a particularly effective tool for expressing and coordinating action intent which is a collective bodily orientation plus eye, head, and that responds to a joint movement [48].

In H2R handovers, it's shown that usage of Gaze during the process has a beneficial impact on the interaction, causing the human givers to reach their target faster and see the interaction as more natural [21][47][68].

For increasing the success rate in H2R handovers, Wei Yang et al. in [76] propose an algorithm trained with humans' grasp data-set in which the robot will reach the handover position and will plan a trajectory and adjust the orientation according to the human giver's hand. Also, Pan et al. propose a training pipeline which examines the accurate recognition of object handovers from the viewpoint of an object receiver using kinematic motions identified by a support vector machine (SVM)[51]. The classifier analyses the kinematic behaviours of the provider (such as joint angles and joint distances from one another and with respect to the receiver) to ascertain the giver's intention to transfer an object. In most cases, the research mainly digs into motion planning of the robot and it's presumed that the human intention is understood by the robot. For instance, in [73], A wearable sensory system with a natural configuration is used by the human, the giver, and the same sensory data will be fed into the robot's system. This way enables the giver naturally control the hand-over process by teaching the robot to understand the human's hand-over intentions.

In R2R handovers, the robots will rely on vision sensors, i.e., eye-in-hand and eye-to-hand cameras, to detect objects and estimate their positions and use haptic sensing, i.e., wrist-mounted force sensors, tactile sensors on the fingers, joint-level torque sensors for the physical exchange of the object [67]. In terms of vision, methods like Faster R-CNN, YOLO and SSD have been used to detect the object and estimate its position. Regarding physical exchange, the robots use specialized force/tactile sensors to communicate, measuring the interaction force and moment. In [13], the giver employs a slipping detection algorithm to anticipate situations where the receiver may struggle to maintain object orientation, preventing dangerous releases.

1.6 Robot to Human Object Handover

As helper robots are becoming more popular in both industry and people's houses, the need for improving the communication between robots and humans increases. One of the main modalities used when doing a robot-to-human handover is haptic, which plays an important role in robot and human's decision makings. Having said that, to improve the handover, verbal communication, like speech, and non-verbal communication, like eye gaze, can be used [50].

As mentioned earlier, an important challenge in Robot-to-human object transfer with respect to the other two handovers, R2R and H2R, is that there is no real-time sensor data exchanging between the human receiver and the robot other than the on-board sensors of the robot. This means ignoring the methods, which include adding sensors to the human body for sending to the robot. Various works use different sensor modalities and rely on solving and improving different challenges of R2H object handover. Our finding is that previous works have not explored R2H handover with action classification. Their approaches are limited and based on specific sensors they have used on the robot, object, and the human. None of the methods are able to distinguish the action and detect the grasp by the receiver.

For instance, Deyle et al. [17] proposea threshold-based object handover based on the forces measured at the base of the robot's fingers. This algorithm would result in a sudden release from the robot. Although this method seems to be smooth, it's probable that the robot releases the object in accidental collisions like bumping. Bohren et al. [11] have solved this issue by adding the gripper displacement as a limit so that the gripper moves for a certain distance to release the object [11]. This way, the robot, giver, makes sure that the action is not accidental. However, their results show the excessive amount of force needed for the handover.

Chan et al. in [12] propose a proportional release controller based on performing human-to-human experiments. As mentioned before, in their human studies, the giver and receiver's grip and load force on the object is measured and it's shown that the receiver's grip force and object load force have a linear relationship that could be used for implementing a release controller for the robot. They have used a force sensor on the object to measure the grip force from both giver and receiver, a force sensor on the middle of the object to measure pulling force, and also a wrist-mounted force sensor on the robot to measure load force. Although showing good survey results from their subject, this works is sensors specific. They have used a special object with force/torque and FSR grip sensors to measure the forces from the object during handover. Moreover, this work cannot differentiate between actions and prevent failure.

Parastegari et al.[54], explore that vision plays an important role for giver's release decision and make use of this for preventing failure. Implemented on a 2-finger gripper installed on the Baxter robot, the major innovation is that the system is able to re-grasp the object in case of downward slippage when it's still reachable by gripper's fingers. The detection of the downward acceleration is performed by an optical sensor installed on the gripper, feeding a closed-loop controller which is able to control the gripping force with respect to downward acceleration as one of the main control parameters. The proposed algorithm is able to release the object smoothly in wide ranges of pulling directions. If the receiver fails to keep grasping the object while pulling or wants to manipulate the object, the feedback from the optical sensor indicates a downward acceleration and then the controller will re-grasp the object. Although reporting good results, this work will detect failure after it has happened. Again this method is sensor specific. The parallel gripper makes it easier to regrasp the object after falling. So it may not be practical for this method to be applied to other grippers.

A technique for a safe R2H object handover is presented in a research and is demonstrated on a Shadow Robot hand equipped with tactile sensing. Eguiluz et al. [18] propose an algorithm that can control the robot's hand in situations where the force by the receiver is not considered safe. In these case, the action is considered a disturbance and may break the robot's fingers or may result in the object being dropped. Assuming a stable grasp configuration, they can quantify the contact forces delivered to the object using BioTAC sensors and provide a feedback signal to a joint effort controller to keep grip forces constant despite disturbances. The method can distinguish between pulling forces on an object that should lead to an object handover and other disruptions. The suggested approach is validated by experimental findings that only when pulling happens, will the hand let go of the object. Compared to our work, we have proposed a stable grasp using tactile sensors on the SDH fingers, but since we don't have joint-level torque sensors in SDH joints, we cannot use this method to detect pulling.

1.7 Contributions

Key contributions in this research are as follows:

1. Proposed a new method for robot-to-human object handover using joint torque data.
2. Using CNN-based classifier, differentiated between different actions applied on the object by the receiver using torque data.
3. Proposed a new algorithmic fusion approach, for reducing the failure (increasing the success rate) in R2H handover. Our algorithm combines inference from torque data with the inference from vision data from an eye-in-hand RGB-D camera that detects visually proper grasp of the object by fingers in real-time (30fps)
4. Our fusion algorithm leads to higher success rates for R2H object handover as compared to solely torque-based approach.

Some parts of the materials presented in this thesis are reproduced and modified from our published conference article [46]:

Mohandes, M., Moradi, B., Gupta, K., Mehrandezh, M. (2023, March)Robot to Human Object Handover Using Vision and Joint Torque Sensor Modalities. In Robot Intelligence Technology and Applications 7: Results from the 10th International Conference on Robot Intelligence Technology and Applications (pp. 109-124). Cham: Springer International Publishing[46]

1.8 Thesis Overview

- **Chapter 1** introduces the motivation, the proposed work, the problem description, and the expected contributions to the research community. Moreover, it comprehensively reviews the literature on robot-to-human object handover systems and techniques.

- **Chapter 2** presents the theoretical background for different techniques used in this thesis, such as machine learning methods. The content includes a thorough literature review on Time-series classification and Convolutional neural Networks since it's used as the main network architecture to implement action classification on torque data. Moreover, the theoretical background on Real-Time object detection and sensor fusion will be elaborated as they are used for making our system robust by including a vision sensor as a safe-grasp detector. Finally, the dynamic and kinematic model of the 7-DOF Kinova robotic arm is presented.

- **Chapter 3** proposes our main approach to perform robot-to-human object handover, Increasing the success rate by using machine learning techniques, and how sensors' data like torque and vision are analyzed via AI techniques during handover. This includes explaining the pipeline for Torque data acquisition, the CNN classifier, and SSD object detection implemented on the robot while in the home position interacting with the receiver for object handover. The extracted Jacobian model of the arm in the previous chapter will be used to formulate the force exerted on the arm and is used to simulate a traditional force-based R2H handover [12].

- **Chapter 4** presents the experimental setup and results of the proposed technique. The CNN and SSD networks are implemented and tested in a real-time experiment. Moreover, the SSD and CNN-based methods have been compared together and alone with the force-based object handover, which is the classical and state-of-the-art way of performing the handovers. Finally, the overall results are discussed.

- **Chapter 5** summarizes the contributions, limitations, and conclusions of this thesis and provides future directions for research in this field.

Chapter 2

Theoretical Background

In this section, we go over the methods and the theories behind them. The kinematic and dynamical model of the Kinova gen3 Robotic arm is extracted. Subsequently, the extracted model is used to calculate the force exerted on the end-effector of the arm to simulate the wrist-mounted force sensor used in the previous R2H methods. We then explain the fundamental idea behind time-series classification-based handover, which is used for analyzing torque sensor data, and the object detection model implemented in SSD network, which is used for detecting fingers wrapped around the object to improve the handover performance better.

2.1 Dynamic and Kinematic Modelling

A manipulator is made up of several rigid bodies (links) joined together by kinematic pairs or joints [14]. The two main types of joints are revolute and prismatic. The whole system is representing a kinematic chain. An end-effector (gripper, tool) is attached to one end of the system, which is also restrained to a base. Degrees of freedom (DOFs) determine the whole structural and mechanical design of the manipulator and moreover will result in different postures in the manipulator. Every DOF typically corresponds to a joint articulation and is a component of the joint variable [14]. Kinova robot is developed with 40hz built-in torque sensors in every joint and will produce the torque values as time-series data in ROS platform [31].

In this section, we will explore the manipulator in static mode to extract the external static forces applied on the arm. We will use forward dynamics with the calculation of Jacobian of the manipulator to extract the external force applied on the gripper of the arm. Forward Dynamics' objective is to determine the applied force on the end-effector in relation to the joint/torque variables.

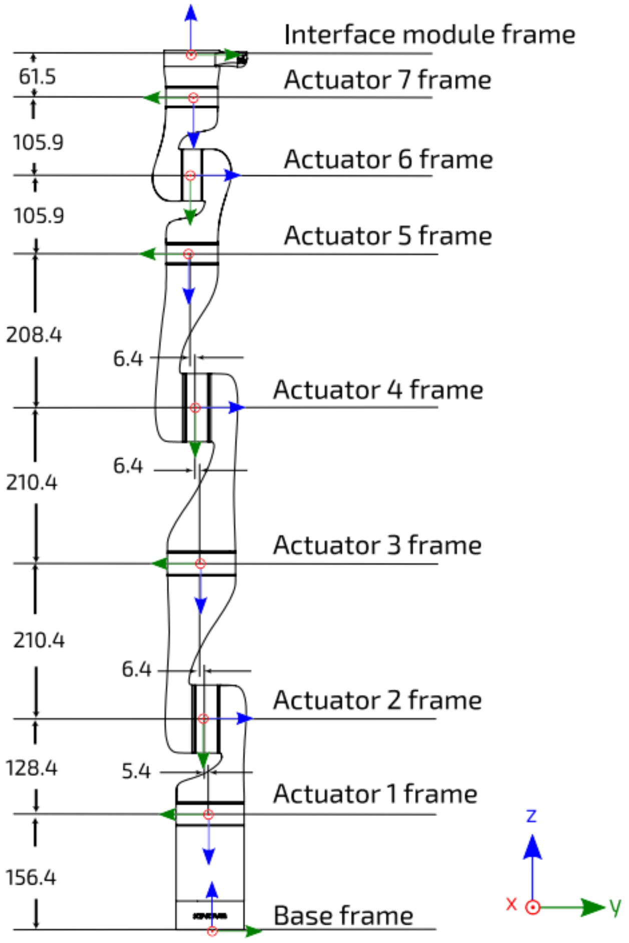


Figure 2.1: 7 DoF robot frame definitions and dimensions (all joints at 0 position, dimensions in mm) [31].

The 7 DOF Kinova arm consists of seven joints and eight links which we are using for robot-to-human object handover. The forward kinematics of the 7 DOF kinova fig 2.1 can be represented by a sequence of reference frames containing the rotational and translational parameters in the form of homogenous transformation matrices [14]. For ease of access, the complete information on Kinova forward kinematics is given in Appendix A.

The relation between joint torque values and the force components at the end-effector is given by the transpose of the Jacobian matrix written with respect to the end-effector frame [14], i.e.,

$$\tau = J^T F$$

where acting F on the end-effector and τ are respectively a 6×1 Cartesian force-moment vector and a 7×1 vector of joint torques. For ease of access, the complete information on how to relate external force to the torque values using Jacobian is explained in Appendix A.

2.2 Time-series Classification

We used time-series classification to detect the unknown actions mentioned in the previous chapter. These actions will change the torque sensor data, which will then be recorded and used for training in our classification network. Time series classification is the art of using a supervised machine learning algorithm to classify labelled time-series data into pre-defined classes. Time series data are analyzed using several labelled classes, and supervised machine learning is used to predict or categorize the class to which a new data set belongs. Data scientists put a lot of effort into making sure that their time series classifiers are as accurate as possible since classification accuracy in these circumstances is crucial [27].

2.2.1 Types of Time-series Classification

A variety of algorithms are made specifically for time series classification. One type may produce classification accuracy that is higher than others, depending on the data. That's why it's important to take different algorithms into account. We give a summary of the time-series classification method from classical to new state-of-the-art ML-based methods. We then choose some methods to use in our algorithms.

Distance-based Approach

A distance measure is a criterion in which the resemblance of two objects is measured with respect to their distance. The lesser the distance is, the more similar the two objects are considered to be. Distance measures such as: 1) Euclidian distance, 2) Hamming distance, 3)Manhattan distance, and 4)Minkowski distance are the most used ones in machine learning algorithms. Along with several well-known distance-based algorithms like k-nearest neighbours (KNN), these distance metrics are employed. It calculates the gap between every object in the training data set and the test object. The new item is then given the class that is most prevalent among the k objects from the training set based on the k shortest distances. When k is set to 1, the algorithm reduces to the one-nearest neighbour, and the test item is given the class of the sample from the training set that is closest in proximity [9].

Also, there are other kernel-based using distance measurements. One of the famous algorithms, the support vector machine, generates a hyperplane (or line in two dimensions) to categorize objects. The class is subsequently assigned in accordance with the test object's position with respect to the hyperplane [9].

Another distance-based approach called dynamic time warping (DTW) calculates the difference between two time series by measuring the gap between each point in the time series and adding these values to determine the overall difference. The technique is designed to handle very small changes [9].

Shapelet

Time series data frequently displays distinctive data shapes that are representative of the time series' class. A classifier that uses a shapelet transform algorithm to examine time series subsequences can produce results that are helpful for classification. As an example, this method would be ideal for classifying the distinctive ECG forms that appear in the heartbeat time series data [78]. The output of a shapelet algorithm applied to time series data displays the minimum distance between the shapelet and each subsequence in the data set. As a result, it is a kind of distance-based technique comparable to DTW, with the exception that the shapelet transform only calculates the distance for selected data subsequences rather than the complete time-series. This method has shown some promising results for time-series classification as it has been used in numerous high-accuracy classifiers[78].

Model Ensembles

A collection of classification models, each of which performs its own class discrimination on the data set, is known as an ensemble model for time series classification. The class that is applied to the data set is the one that emerges most frequently from the classifiers that have been collected. For this error to be effective, any classifier error should not be correlated. The approach's strength is lost if the errors are correlated, and the ensemble's accuracy falls below that of a single classifier. It is possible to lessen the impact of correlated error by grouping the classifiers by type and selecting one classification from the group when several types of algorithmic classifiers are combined into an ensemble. The ensemble classification is then performed using this, along with the classification from the other groups. These kinds of ensembles serve as the foundation for the HIVE-COTE algorithm, and unlike bigger data sets, excel at relatively small time series classification [75].

Dictionary Approaches

Dictionary structure, which is used to describe the meaning of a work in most cases, serves as the basis for another popular form of classifier algorithm. However, a dictionary method can also be used to describe things other than words. For instance, the number of occurrences of a specific shapelet in a time series could be described using a dictionary [79].

An application of using dictionaries is implemented in Bag of Words algorithm. This method, aka BOW algorithm, counts the occurrence of words inside a document and uses this information to train a model robustly classifying different classes of text. BOW uses the assumption that two texts are identical if similar properties (e.g., similar word counts) occurs. In order to make this method compatible with time-series classifications, Bag of Patterns comes to play. This algorithm takes a time window and examines the magnitude of the signal. This signal goes to another transformation resulting in a mean value as a representative of the time-series data within the defined window and later, fed into a dictionary-based classifier [7].

Interval-based Approaches

This method is almost a mixture of model ensemble and BOP method. Comparable to the Bag-of-Patterns method previously examined, interval-based methods separate the time series into discrete intervals. Similar to Model ensembles method, a unique machine learning model will then be assigned to use specific intervals of each data set training data. This way ensemble of classifiers, each of which operates on a different subsequence or interval will determine the class of the time-series on the basis of the frequent output classes in the classification results [59].

Deep Learning

What is DNN? A complex type of neural networks with numerous numbers of parameters is called deep learning. Compared to other kinds of algorithmic models, these models are often far more sophisticated and have a much higher number of parameters. The time-series data will first go through the initial layers of the network, which extract and encrypt the fundamental shapes within the data and the final layers are responsible for assigning different classes to the data based on encoding the data interpretations [36].

Up until a few years ago, the research community would avoid using deep learning methods compared to the conventional methods (the ones mentioned above) not only for the huge computation burden and time usage it had but also for the lower accuracy. Now the results are showing promising in terms of both decreased training time and accuracy of the classification output. For instance, successful approaches such as sktime ROCKET transform show to better perform in time series benchmark data sets from the UCR/UEA archive, than HIVE-COTE being one of the best ensemble methods [22].

As classification methods are improving in terms of accuracy and lower computation burdens, time-series classification has shown a great catch for the research community. Many algorithms have been developed for this matter, but just a few are using deep learning to address this issue and have given Deep Neural Networks (DNNs) any thought as a potential solution. Despite the great success of DNNs in computer vision, speech recognition and image classification empowered with emergence of CNNs and RNNs, it's surprising that this method is used by a few researches [27]. In 2012, Krizhevski et al. [33] made a huge impact on computer vision using Deep Neural Network and it resulted in classifying 1.2 million high-quality images into 1000 classes with 17% error. In 2015, it was explored that in order to reach a human-like accuracy in image recognition, a deeper and wider network was implemented [70]. Given the huge success of the research community in solving computer vision problems using DNNs, some have explored different applications of deep learning in texts. For instance, to implement language translation, specifically, English to French, [69] and [5] have reached some achievements in doing so. In [69], LSTM layers are used to first map the input series into a vector and later to perform the feature extraction on the target vector while [5] proposes the idea that a fixed-length vector which is used for encoding and mapping the input sequence, is a bottleneck to the increasing of the translation performance and it's better to use a flexible-length vector.

In document classification, input to most of the machine learning algorithms is best expressed as a feature vector of fixed length. Bag-of-words which is explained in the above sections, despite their widespread use, suffer from two main flaws: they fail to preserve the original word order and pay no attention to the words' meanings.[35] addresses this issue and proposes an unsupervised approach, which will represent sentences as vectors. This technique turns a document into a dense vector to represent each document and learns to anticipate the words it contains. Due to fundamental similarities such as being sequential, NLP and speech recognition methods have used pretty much the same methods for classification and recognition. Speaking of being sequential, time-series data hold the same feature making it easy for the research community to solve TSC problems.

We will elaborate on some definitions, which then will be followed by the process of DNN-based time-series classification:

Some definitions[27]:

1. $X = [x_1, x_2, ..., x_T]$ is a univariate time-series in which x represents real values in the T number of sequence. x_i represents the ith element in the time series.

2. $X = [X^1, X^2, ..., X^M]$ is a M various univariate time-series where $X^i \in R^T$

3. $D = (X_1, Y_1), (X_2, Y_2), ..., (X_N, Y_N)$ is a Dataset in which X_i is either a univariate or multidimensional time-series data, whereas Y_i is a vector including same number of binary values as the number of class labels K. The mentioned vector caan show what classes are the nearest ones that can be used to assign the data vector to.

Time-series classification, aka TSC, has the objective to map the input data onto a probability distribution and output the most related classes to the input based on the probability percentages.

Deep Learning for Time-series Classification

In this work, we have used Deep Neural Network for our time-series classification task. Below is a common figure for showing general structure of a DNN system.

The purpose of a typical DNN is to acquire a structure for data characterization. These networks are mainly constituted of several layers L. Containing a determined number (N) of nodes, also known as "neurons", each individual layer provides multiple (N) outputs. Each "layer" in a deep neural network is a set of parametric functions that collectively reflect some

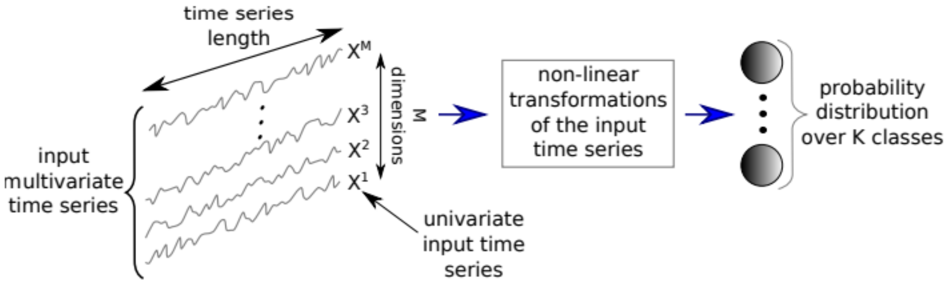


Figure 2.2: Structure of DNN for a sequential data like time-series classification [27].

aspect of the input domain [53]. As mentioned earlier, the whole calculation is happening in a hierarchical process in which each layer, l_i such that $i \in 1, ..., L$ is fed with the output from nodes of the previous layer, l_{i-1}, and l_i's output data will be used as the input to the neurons of the next layer l_{i+1}. In order for the outputs to be calculated, the inputs are passed through a non-linear function. A collection of parameters unique to each layer governs how these non-linear transformations will behave on the layer they are applied to. These parameters θ_i, which are referred to as weights in the context of DNNs, are what relate the output of the layer below the current layer to the input of the layer before [27]. Therefore, if a neural network is given an input x, the computations below will be carried out in order to forecast the class:

$$f_1(\theta_1, x) \tag{2.1}$$

where 2.1 represents the output of the first layer.

$$f_2(\theta_2, f_1(\theta_1, x)) \tag{2.2}$$

where 2.2 represents the output of the second layer.

$$f_3(\theta_3, f_2(\theta_2, f_1(\theta_1, x))) \tag{2.3}$$

where 2.3 represents the output of the third layer accordingly. As the equations are written, the following equation 2.4 for the final layer appears,

$$f_L(\theta_L, x) = f_{L-1}(\theta_{L-1}, f_{L-2}(\theta_{L-2}, ..., f_1(\theta_1, x))) \tag{2.4}$$

where f_i is the non-linearity that was applied at that layer. In the academic literature pertaining to deep learning, this method is sometimes referred to as feed-forward propagation [27].

Throughout training, the whole network undergoes processing with input and output data. The determination of the weight parameters θ_i in each layer i is one of the vital aspect of training. Despite the fact that the weights, the network model, can be extracted and fine-tuned using a model that has already been trained on a source dataset [52], some researchers suggest that initial value of the weights be determined randomly, or usually initialized with zero values, by the algorithm implemented on the dataset [38][15]. The last layer, i.e., layer L will have K neurons, corresponding to K classes. The output result is thus a K element vector, and its component values are the odds that x belongs to each class according to the estimates. The error between the defined target values and the measured-initial weight-values will be determined using a cost function such as negative log likelihood [27]. The final goal is to reduce the loss between the predicted values and the target values. To achieve this part, a backward iteration accompanied by gradient descent method [15], starts updating the weights from the last layer so that the error is propagated. This process is done in numbers of iteration of forward calculation and backward propagation until a minimum loss is reached. After training, the model will be tested on a stack of unused dataset that has not been seen yet and will be fed forwardly to the network. This process is also known as inference phase in which the probabilty model is evaluated based on the dataset that has not been used for training. This phase will evaluate the model's accuracy via some methods like accuracy measure [4] which is basically the percentage of the successful predictions over the total number of trials. The excellence of DNNs over conventioanl TSC methods like DTW is that the result shows the level of confidence in perdiciting the correspnding class of the input data [34]. We now discuss some end-to-end DNN algorithms.

Multi-layer Perceptrons Multi-layer perceptrons, also known as MLP, because of it's non-complex design is considered to be the most straight-forward and conventional DNN method. Due to the fact that subsequent layers are fully-connected to each other via their neurons being connected to all the other neurons in the previous and subsequent layer, This network is known as FC network. The weights in the model determine these connections between the layers. Equation 2.5 shows how a non-linear function is applied on the input data through the model layers:

$$A_{l_i} = f(w_{l_i} * X + b) \tag{2.5}$$

Where A_{l_i} represents the output of the activation function applied on the input, w_{l_i} represents the weights in layer i and b is the bias [27]. The number of Neurons in each layers is then determined as a hyperparameter in the network setup.

The lack of spatial invariance is a drawback (barrier) when time-series data is adopted as MLP input. Time series data consist of temporal information which will be neglected if each time-stamp (time label) is assigned with a specific weight which means the timeseries' elements will be processed independently from the other elements in the time series. Each neuron d will hold a weight set having $T \times M$ values where M is the dimension of the time-series and T represents the length of the time-series [27].

The final layer of a TSC model is typically a discriminative layer. This type of layer takes as its input the activation from the layer that came before it and outputs a probability distribution over the class variables contained in the dataset. The vast majority of deep learning strategies for TSC make use of a softmax layer, which is analogous to an FC layer in which softmax is used as the activation function f and in which the number of neurons in the layer is equal to the number of classes present in the dataset. The usage of the softmax activation function is motivated by the following three key helpful properties: It is an adaption of logistic regression to the multinomial case, and the sum of probabilities is guaranteed to equal 1 and the function is differentiable [27]. The softmax function will give the following Eq. 2.6 as a result:

$$\hat{Y}_j(X) = \frac{e^{A_{L-1} * w_j + b_j}}{\sum_{k=1}^{K} e^{A_{L-1} * w_k + b_k}} \tag{2.6}$$

where $\hat{Y}_j$ represents the likelihood that X is assigned to class Y equal to class j, from the total of K classes in the dataset. For each class j, the set of weights w_j (and the accompanying bias b_j) are connected to every prior activation in layer l_{L-1} [27, 37].

It is recommended that an optimization technique that seeks to minimize an objective cost function be used in order to learn the weights that are used in equations 2.6 and 2.5 automatically. Defining a differentiable cost function that evaluates the error of a given value of the weights is necessary for performing an approximation of the error [27]. There are multiple methods to carry forward the process of updating the weights for classification problems solved by DNNs by considering the decrease in the loss function. Categorical cross Entropy is a popular loss function which is used for calculating loss for each input (Eq. 2.7) and average loss (Eq. 2.8) for the whole input [27].

$$L(X) = -\sum_{j=1}^{K} Y_j log \hat{Y}_j \tag{2.7}$$

$$J(\Omega) = \frac{1}{N} \sum_{n=1}^{N} L(X_n) \tag{2.8}$$

Eq. 2.7 provides the loss value when time series X is fed into the network. Eq. 2.10 shows that the set of weights of the network, denoted by ω are being modified by minimizing the loss function. The relationship between the loss function and updating the weights is shown in Eq. 2.10.

$$w = w - \alpha \frac{\partial J}{\partial w} | \forall w \in \omega \tag{2.9}$$

α is preset manually as the learning rate [37].

Convolutional Neural Network Many real-world applications such as speech recognition, natural language processing, and etc are considered to be some data over the course of time which makes them known as time-series. Researchers have begun using CNN architectures for time series analysis as a result of their widespread success in a variety of applications [36][20]. Achievements of deep CNNs almost began with AlexNet [33] [32] winning the image competition in 2012. [70] performs a human-level image classification.

One way to think about a convolution is as the process of applying and rolling a filter across a time series. In contrast to photos, the filters only display one dimension, which is time, rather than both the spatial dimensions simultaneously (width and height). Applying convolution to a time series can be shown as a non-linear conversion (transformation) [27]. As an example of a general formula for using the convolution of a certain length l for a time window of T for a time-series we have:

$$C_t = f(w * X_{t-l/2:t+l/2} + b) | \forall t \in [1, T] \tag{2.10}$$

where C represents the output of applying a convolution (dot product) to a time series X of length T using a filter of length l, b shows bias, and a non-linear function f, such as the Rectified Linear Unit (ReLU). When applied to a time series X, convolution (a single filter) transforms it into a filtered univariate time series C. If many filters are applied to a time series, then the resulting time series will be multivariate with as many dimensions as filters were applied. The idea behind using several filters on a single input time series is to learn more than one discriminative feature that can aid in the classification process [36, 27].

A strong feature of CNNs which makes it better in learning rate is that it uses the same filter for all the time-stamp which will result in an invariant filter learned for 1 time-stamp. The filter in CNN when a multivariate time-series is fed as input will have dimensions same as the number of dimensions in MTS [36, 27].

2.3 Real-Time Object Detection

Real-time object detection has been a popular research topic in computer vision and deep learning over the past few years.

- YOLO (You Only Look Once) - This is a popular real-time object detection algorithm that was introduced in 2016. YOLO uses a single neural network to predict bounding boxes and class probabilities directly from full images in one evaluation. It is known for its simplicity and speed, achieving real-time performance on a GPU [57].

- SSD (Single Shot MultiBox Detector) - Introduced in 2016, SSD is another popular object detection algorithm that achieves real-time performance. Like YOLO, SSD also uses a single neural network to predict bounding boxes and class probabilities.

However, SSD uses a different architecture that includes multiple feature maps of different sizes to handle objects of varying scales [41].

- Faster R-CNN (Region-based Convolutional Neural Network) - This is a two-stage object detection algorithm. Faster R-CNN achieves state-of-the-art accuracy on various object detection benchmarks. However, it is slower than YOLO and SSD due to its two-stage approach that involves generating region proposals followed by object detection on those proposals [58].

- RetinaNet - This is a one-stage object detection algorithm. RetinaNet uses a novel focal loss function that down-weights the loss assigned to easy examples and focuses on hard examples during training. This allows RetinaNet to achieve high accuracy on object detection tasks while maintaining real-time performance [40].

- EfficientDet - This is a family of object detection models. EfficientDet uses a compound scaling method to efficiently scale up the model size while maintaining computational efficiency. The largest model in the family, EfficientDet-D7, achieves state-of-the-art accuracy on COCO object detection benchmarks while maintaining real-time performance on GPUs [72].

2.3.1 Single Shot Multibox Detector (SSD)

The Single Shot Multibox Detector (SSD) architecture is a deep convolutional neural network (CNN) that is designed for real-time object detection. The main idea behind SSD is to use a single neural network to directly predict the class and location of multiple objects in an input image [41].

The SSD architecture consists of two parts: a base network and a detection head. The base network is usually a pre-trained CNN, such as VGG or ResNet, that is used to extract features from the input image. The detection head is a set of convolutional layers that are used to predict the class and location of objects in the image [41].

The detection head is composed of several convolutional layers that are used to produce feature maps at multiple resolutions. These feature maps are then used to predict object class probabilities and bounding box coordinates at each spatial location. Specifically, the detection head consists of:

- Convolutional layers: The first set of convolutional layers in the detection head are used to process the feature maps from the base network and generate a set of intermediate feature maps.

- Multibox layers: These layers are used to predict the class and location of objects at different scales and aspect ratios. Each multibox layer generates a set of bounding box predictions, where each box is associated with a specific anchor box. An anchor box is a fixed-size and aspect ratio box that is placed at each spatial location on the feature maps.

- Prediction layers: These layers are used to combine the predictions from the multibox layers and generate the final set of object class probabilities and bounding box coordinates.
- During training, the SSD architecture is trained end-to-end using a loss function that combines the classification loss and the localization loss. The classification loss penalizes the model for incorrect object class predictions, while the localization loss penalizes the model for inaccurate bounding box predictions [41].

Chapter 3

Algorithmic Implementation

In this chapter, we will go through the methodology and implementation of our different algorithms on our robotic testbed consisting of the Kinova robotic arm and Schunk Dexterous Hand.

Due to time limitations, we explored two popular methods for time-series classification methods, CNN and MLP [27] [26]. Since the results on MLP were not satisfactory, hence we used CNN and report those results in this thesis.

Please note that that the arm is equipped with two sensor modules: i) a joint-level torque sensor and ii) an RGB-D camera mounted on the robot's end effector, and these two sensors are used in our object handover strategy. We deploy a Convolutional Neural Network (CNN)-based classifier to categorize different actions via torque data, while an SSD object detector makes use of the RGB-D data to recognize and estimate the location of human fingers approaching and grabbing the object. The joint torque data is fed into a convolutional neural network (CNN) classifier, which then provides an outcome class (e.g. pull, push, bump, pull-up, hold, and no-action). With the help of the SSD object detector, we can get a bounding box for each detected finger alongside its relative position to the Object. The results of the torque-based classification are combined (fused) with those obtained from the finger detection (bounding box and pose) network in order to arrive at a binary RELEASE decision for the object handover mission.

3.1 Grip Force Controller

A grasping PID Controller [1] is specifically developed to optimize the performance of SDH gripper for grasping objects. This controller utilizes a Proportional-Integral-Derivative (PID) algorithm to regulate the finger movements and gripping force during object grasping. The algorithm is designed to interpret sensory data from the hand's sensors and make intelli-

[1] In fact it is a pure proportional controller in our current implementation, with only k_p, the proportional gain being used.

gent decisions regarding grasping strategies. By continuously monitoring both position and velocity of the finger joints and force feedback from the tactile sensors on the fingers, the PID controller calculates the appropriate control signals to ensure precise and stable grasping. It dynamically adjusts the finger positions, grip strength, and response time based on real-time sensory information, allowing the robotic hand to adapt to various object shapes, sizes, and weights.

As shown in figure 3.1, there are two joints in each finger, adding up to 6 joints in three fingers that allow for extension and flexion, each with a 90-degree range of motion. The seventh actuator facilitates abduction and adduction by allowing two fingers to rotate in opposite directions at the same time. This actuator can be rotated between 0 and +90 degrees. Also, the two phalanges that make up each finger are called the proximal and distal phalanges, with respect to the palm. Each phalange has a tactile sensor matrix built within it. The tactile sensors are comprised of 13 by 6 tactile cells. Most pressure sensors provide a resolution of 12 bits, which translates to a digital pressure range of 0–4095 (or 0-250 kPa). Fig 3.1 represents an informative image of SDH gripper [65].

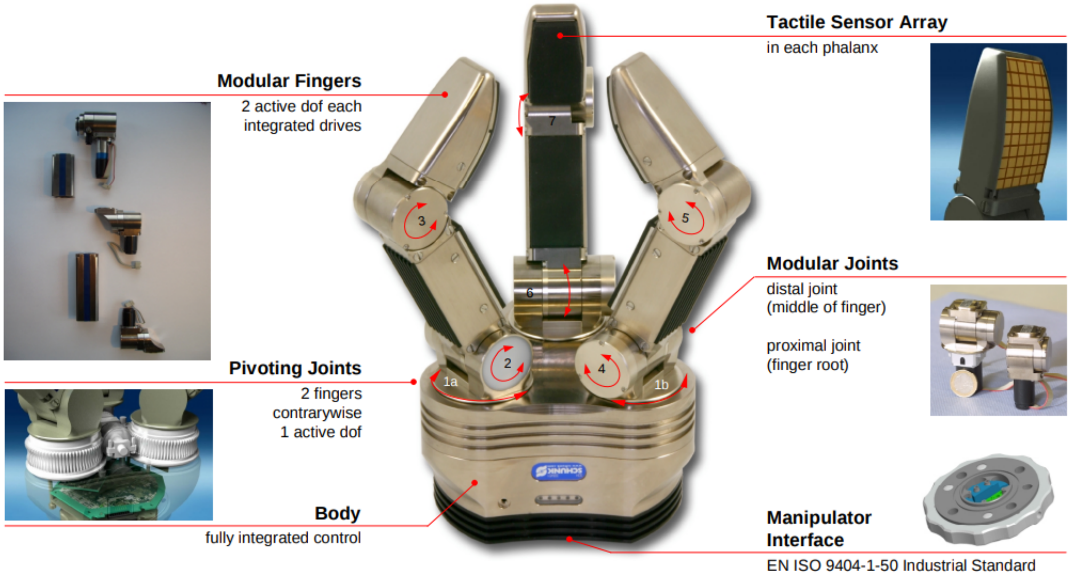


Figure 3.1: 7 DoF Schunk dexterous hand [65].

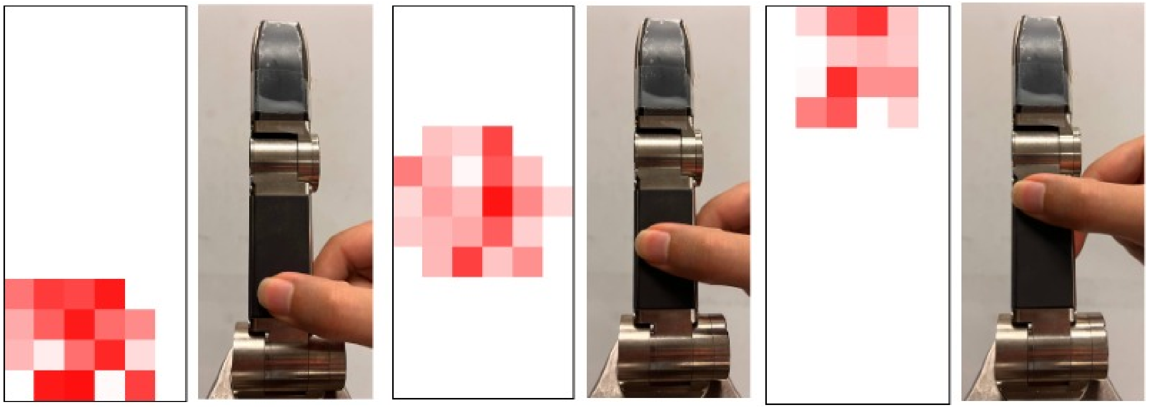


Figure 3.2: Tactile data samples showing how different taxels change their values based on the applied pressure on them.

3.1.1 Pre-Grasp Pose

Although it's not our main focus, but before the process of physical exchange of the object happens, we need to figure out a pre-handover pose for the SDH to be able to grasp the desired object [2]. There exist a couple of grasping configurations to be deployed on SDH. All the grasp types will be categorized in two groups of **Power** and **Precision** grasp [77]. For the cylindrical object in our experiment, we decided to go with Power cylindrical and Precision pinch for grasping. We wanted our object to be grasped fully and to include at least three fingers of the SDH in grasping. With this being said and the application of the system, we experimented with two different types of robot grasps: pinch grasp and power grasp. It's worth mentioning that since we are recording torque data in the Kinova joints, the type of grasp will have little effect on the joint torque values because the resultant force in both cases (for the same object) would be quite similar under the assumption that the action applied acts in a quasi-static manner. Furthermore, even in both power grasp and pinch grasps that we used, the resultant forces are planar. The key reason that power grasped is used for recording training data-set is that power grasp holds the object tightly, and since the object will go through lots of actions by each subject, we wanted to make sure that the object will not move during data-set collection just to save time. For R2H experiments (later in the studies reported in the next chapter), we used the pinch grasp since it will provide a smoother object handover for the human receivers.

[2] This work was modified from a previous work in RAMP lab that was done by Anurag Agrawal.

Power Grasp

- Pre-grasp configuration: Fig 3.3 shows the proper pre-grasp configuration for power grasp which is controlled and achieved by sdh position controller.
- Grasp controller: Fig 3.4 shows the velocity controller via which the fingers are positioned around the object with the desired amount of force. In this mode, two separate PID controllers for proximal and distal joints are monitoring the fingers' positions. The "*halt check*" block checks if the calculated error is within a threshold value ($Threshold$ value is mentioned in Table 3.1). If not, then the proximal joint starts moving till it reaches the desired value till it reaches the threshold and stops. Then the distal joint will go through the same process till the corresponding tactile value reaches its desired value within the threshold. Fig 3.5 shows the steps that the power grasp controller runs to grasp an object. Table 3.1 shows the hyperparameters and gains used in this controller.

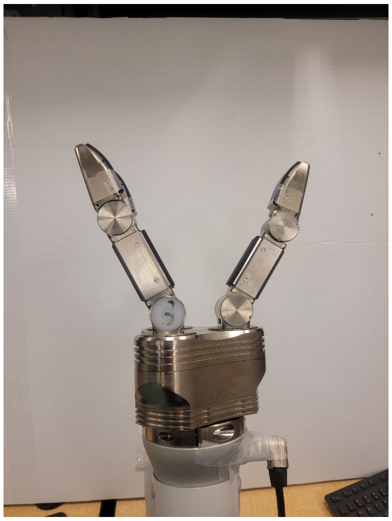


Figure 3.3: Power grasp pre-pose.

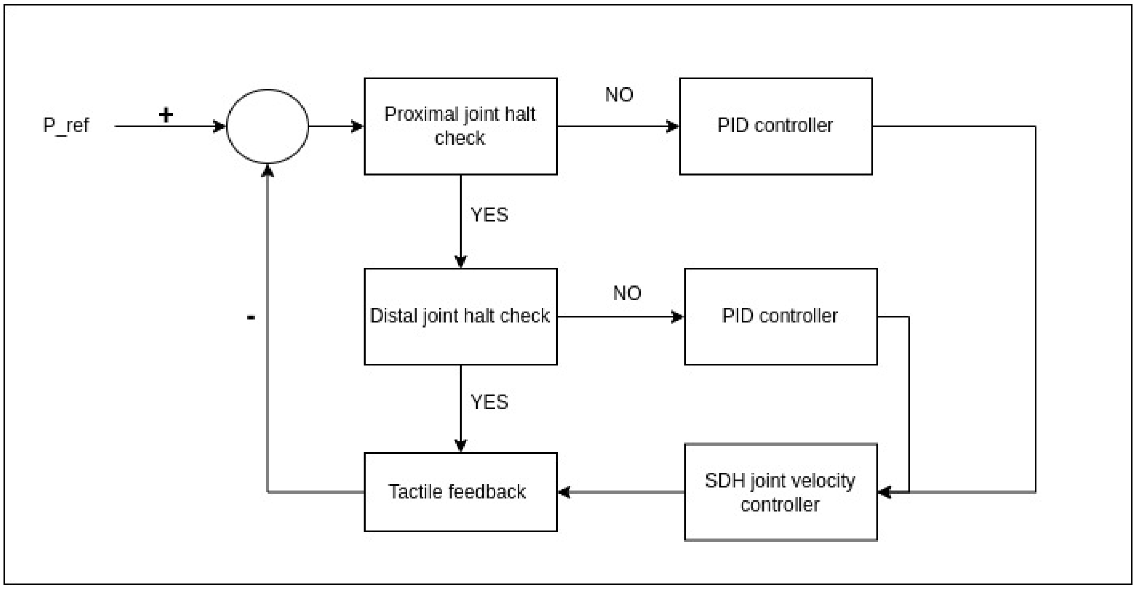


Figure 3.4: Power grasp controller implemented on SDH.

Table 3.1: Hyperparameter values for power grasp controller.

$k_p \times 100$	$P_{ref} \times 100$	$threshold \times 100$
[0.005, 0.01, 0.005, 0.01, 0.005, 0.01]	[12, 12, 12, 12, 12, 12]	[1, 1, 1, 1, 1, 1]

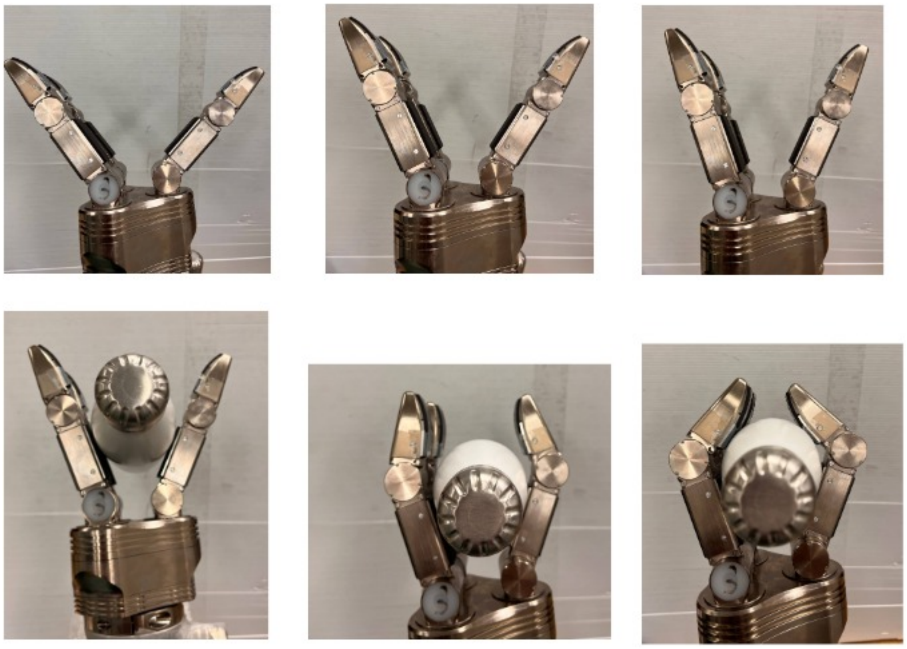


Figure 3.5: Power grasp process.

Pinch Grasp

- Pre-grasp configuration: Fig 3.7 shows the proper pre-grasp configuration for pinch grasp.
- Grasp controller: Fig 3.6 shows the velocity controller via which the fingers are moved to obtain the desired amount of force when pinch grasp is achieved. The "*halt check*" block checks if the calculated error is within a threshold value (variable *threshold* in Table 3.2. If not, then the joints start moving till it reaches the desired value till they reach the threshold and stop. In this stopped state, each finger controls the position of its two joints with a distal/proximal coupler defined in the code. The coupler is a block which moves the joints of one finger move in opposite directions with similar values to maintain the pinch grasp configuration. This achieved by using $-k_p$ for the proximal joint and k_p for the distal joint, as shown in Table 3.2. Fig 3.8 shows how the coupler works and also shows the steps that the power grasp controller runs to grasp an object. Table 3.2 shows the hyperparameters and gains used in this controller.

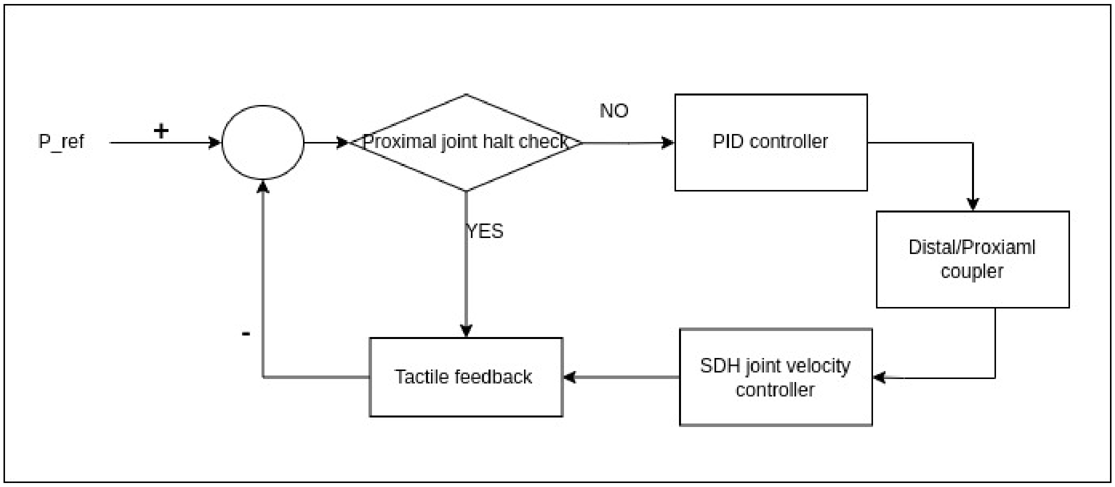


Figure 3.6: Pinch grasp controller implemented on SDH.

Table 3.2: Hyperparameter values for pinch grasp controller.

$k_p \times 100$	$P_{ref} \times 100$	$threshold \times 100$
[-0.005, 0.005, -0.01, 0.01, -0.01, 0.01]	[0, 30, 0, 10, 0, 10]	[0, 2, 0, 1, 0, 1]

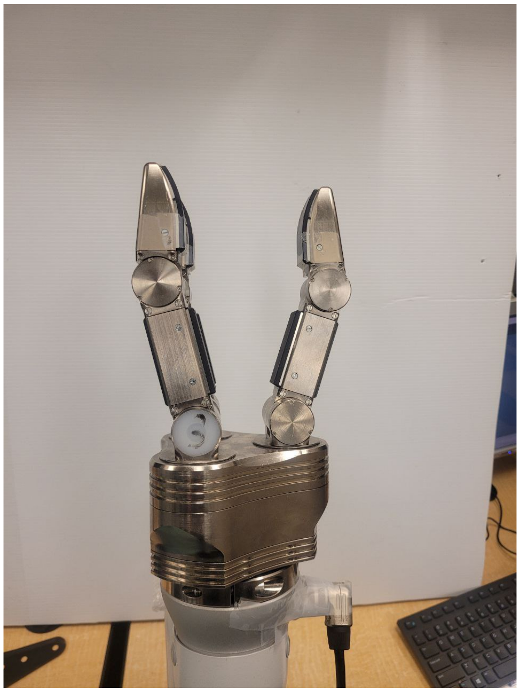


Figure 3.7: Pinch grasp pre-pose.

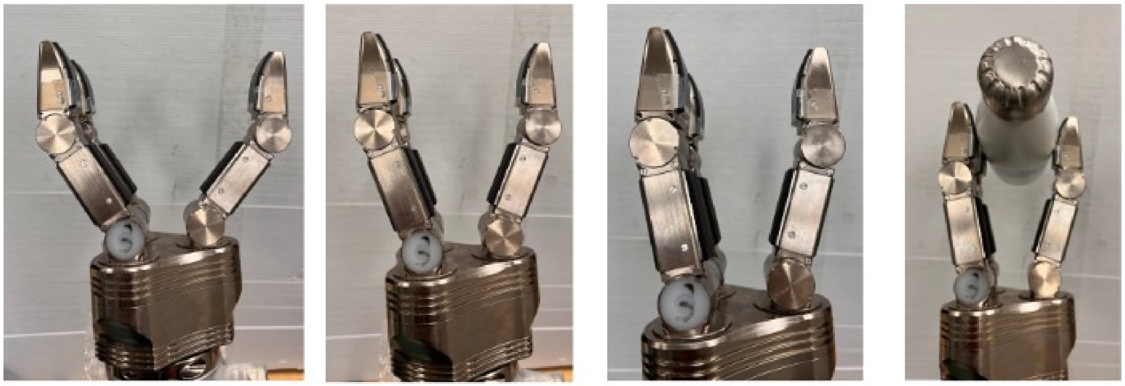


Figure 3.8: Pinch grasp process.

3.2 Torque-Based Object Handover

Our robot is endowed with tactile, torque and vision sensors. We can use torque as well as tactile sensors to detect action on the object from the human receiver. Previous work in our RAMP lab [16] exolored detecting actions applied by human on various objects in different configurations of the end-effector using tactile sensors on the fingers. The recorded tactile data streamed by SDH was processed using Bag of Words (BOW), K-means clustering and SVM to differentiate between four actions: pull, push, bump, and hold. Among the methods for solving the R2H handover problem, we noticed that most of the works by researchers have used force sensors to adjust the giver's gripping force to the pulling force by the human receiver. These force sensors were mounted on the giver's end-effector, acting as a wrist-mounted force sensor, on the object to detect the gripping force, and on the human receiver's wrist to measure the pulling force in H2H experiments [54, 12, 11]. Although the previous works propose smooth algorithms for R2H handover, they don't detect different probable actions that may be applied to the object. Since we have access to joint-level torque sensors in the Kinova arm instead of force sensors, we tend to learn different actions by recording seven different time-series data from torque sensors for each action. This way, we get to have at least seven features for each action via torque sensors compared to one feature provided by the force sensor.

Classifying between these different classes (actions) is not doable using previous simple threshold-based methods. Especially because each action consists of seven time-series data from torque sensors. Using Artificial Neural Networks like CNN or MLP have several thousands of parameters to train a model and learn each action. The ANN can also be trained with a variety of actions extracted from different human subjects to account for variability in applying actions to an object.

First, we present an action classification method, based on the torque data [46]. These actions and their correlated torque data will be recorded in a time-series. The actions will be recorded while the robot is in home configuration as shown in 3.9. With access to 7 joint-level torque sensors on kinova arm, our proposed method is to train the robot through torque data to be able to differentiate between the actions. We also use a Finite State Machine to make the release decision based on the network output. To regulate the actions of a robot, "ROS Finite State Machine" refers to a design pattern or software architecture frequently employed in robotics development. As a result of sensor readings, user interaction, or other events, it facilitates the management of the system's state transitions and subsequent actions. [64]

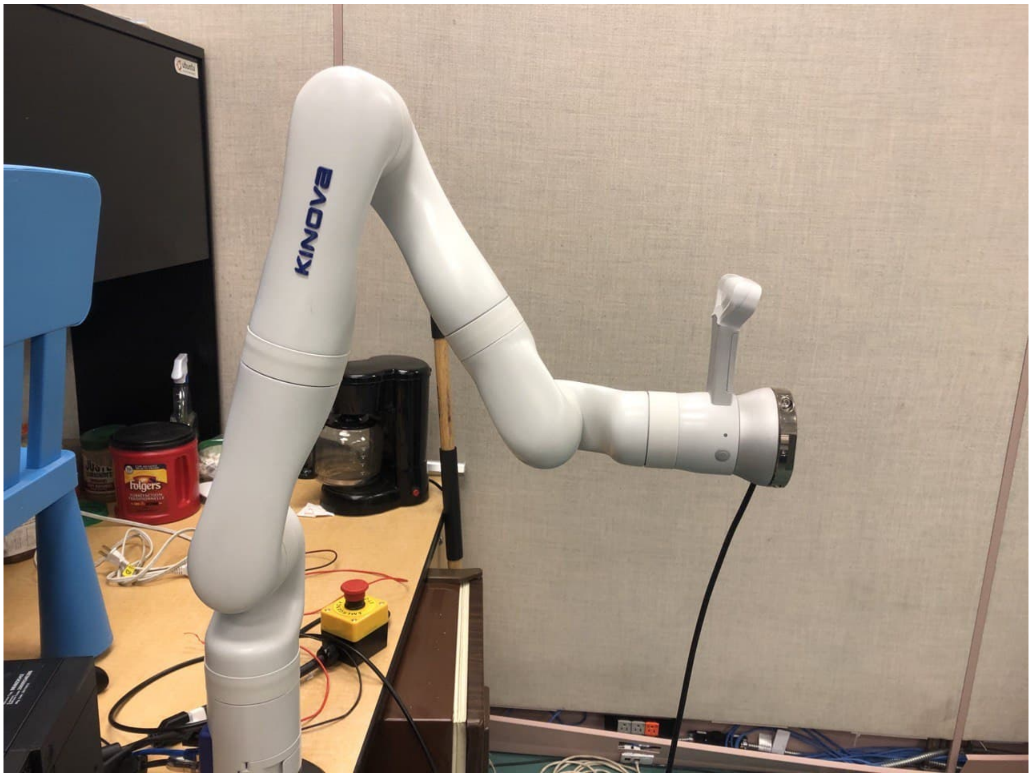


Figure 3.9: Kinova configuration for object handover.

3.2.1 Torque Sensors on the Kinova Gen3 Robot

The torques sensors in Kinova Gen3, one sensor per joint, publish torque data in a time-series structure. Fig 3.10 shows joint-level torque data in time-series samples.

Each torque sensor is sampled at 40 Hz and when the robot is put in position control mode, torque sensor values for each joint will stay constant until an external force is applied on the robot and changes the related torque sensor values. Figure 3.10 shows the change in the torque data under a random action applied on the end-effector for 10 consecutive seconds. For more information about Kinova Robot specification, please refer to Appendix B.

3.2.2 Defining Different Actions

In this work, we are planning to differentiate between the actions which are considered for receiving the object and the actions which are intentionally or unintentionally applied that are not meant for receiving the object. After reviewing the previous work [16] and experimenting human-to-human experiments, the total actions were narrowed down to 6 actions. During the handover, the receiver may accidentally carry out any of these actions and some are not meant for receiving. The actions are explained as follows:

1. Pull: It happens when the receiver pulls the object toward him/herself.

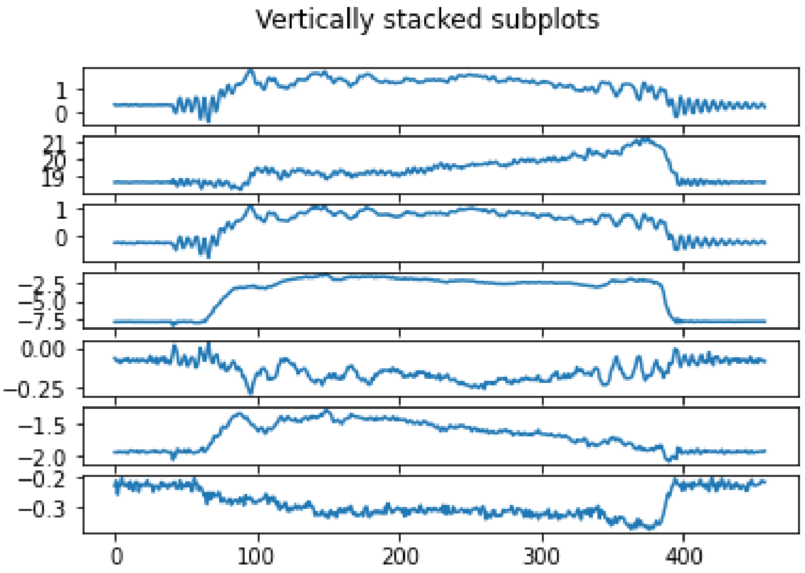


Figure 3.10: Torque data samples in time-series for seven different joints.

2. Bump: It happens when the receiver accidentally bumps the object.

3. Pull-up: It happens when the receiver pulls the object toward him/herself with a diagonal or upward direction.

4. Push: It happens when the receiver accidentally pushes the object toward the giver and also in a downward direction.

5. Hold: It happens when the receiver holds the object.

6. No-action: It happens when no action is being done by the receiver and the robot is in steady mode.

The visual representation of each of the above actions is shown in fig 3.11. Table 3.3 shows what is expected from the giver in response to each action.

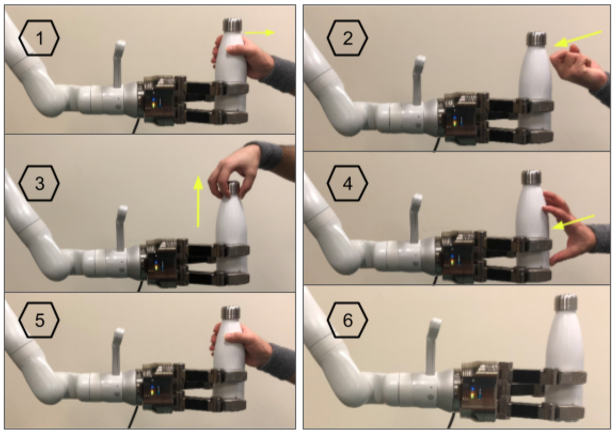


Figure 3.11: Kinova robot in handover position which is called home position [46].

Table 3.3: Each action in the object handover mission is evaluated using pre-defined success criteria [46].

actions	success criteria
no action	Do not release the object
bump	Do not release the object
push	Do not release the object
hold	Release the object smoothly
pull	Release the object smoothly
pull-up	Release the object smoothly

Figure 3.12 shows the final setup for data-set collection in different actions for R2H handover.

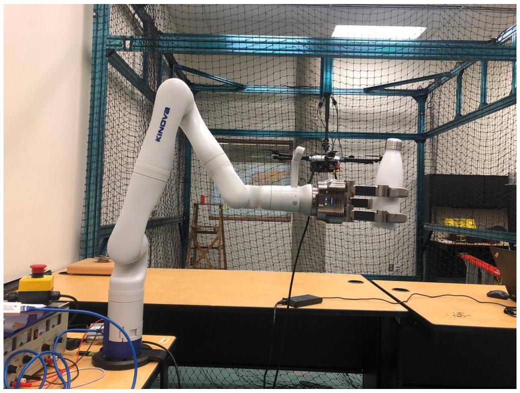


Figure 3.12: Kinova robot in handover position which is called home position.

3.2.3 Dataset and Annotation

We went through an extensive data gathering and annotation process, which is detailed below, to give us enough data points for the training purpose [46]. When classifying actions based on torque, joint-level torque data are taken into consideration. Kinova's application programming interface (API) and its ROS driver report the torque data. The procedures that we followed to acquire the data are depicted in Figure 3.13. The process of data recording for a given action takes 4.5 seconds. After the prompt is shown on the screen, the subject has about 4.5 seconds to apply the desired action on the object. Meanwhile, the data is being recorded through a ros node. Then the next prompt will be shown for the next action till it reaches 100 actions. Figure 3.14 shows the screenshot from the UI. We solicited ten volunteers, split evenly between males and females, to carry out the six actions that make up our data-gathering procedure. Every action was carried out by each volunteer a total of hundred times.

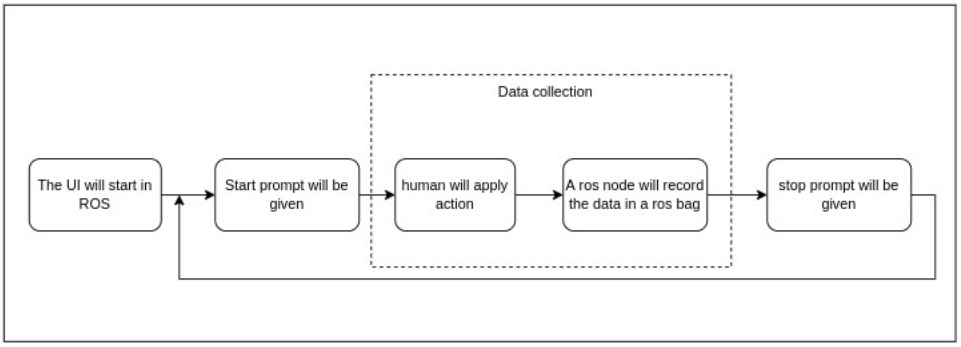

Data collection
The UI will start in ROS
Start prompt will be given
human will apply action
A ros node will record the data in a ros bag
stop prompt will be given

Figure 3.13: Block diagram of data collection.

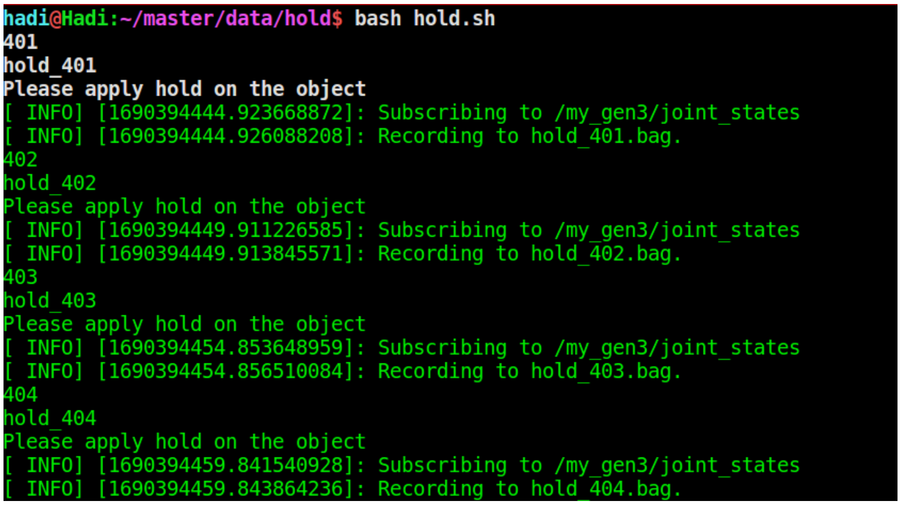

```
hadi@Hadi:~/master/data/hold$ bash hold.sh
401
hold_401
Please apply hold on the object
[ INFO] [1690394444.923668872]: Subscribing to /my_gen3/joint_states
[ INFO] [1690394444.926088208]: Recording to hold_401.bag.
402
hold_402
Please apply hold on the object
[ INFO] [1690394449.911226585]: Subscribing to /my_gen3/joint_states
[ INFO] [1690394449.913845571]: Recording to hold_402.bag.
403
hold_403
Please apply hold on the object
[ INFO] [1690394454.853648959]: Subscribing to /my_gen3/joint_states
[ INFO] [1690394454.856510084]: Recording to hold_403.bag.
404
hold_404
Please apply hold on the object
[ INFO] [1690394459.841540928]: Subscribing to /my_gen3/joint_states
[ INFO] [1690394459.843864236]: Recording to hold_404.bag.
```

Figure 3.14: Screenshot of a typical screen shown to the human subject.

When performing tasks that require the application of force, the participants are instructed to exert varying amounts of force in varying directions. This is shown in Figure 3.15. This ensures that the dataset has an accurate representation of a wide range of forces.

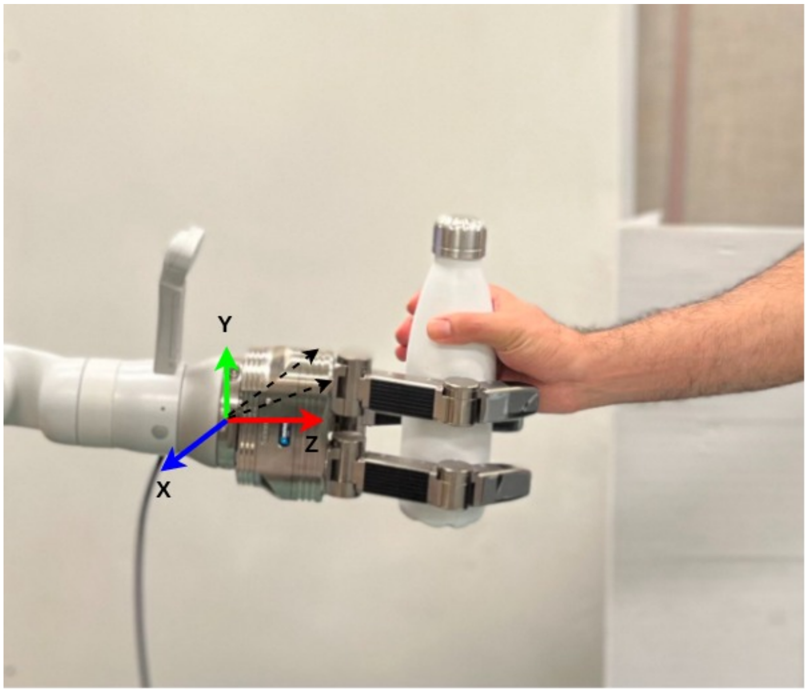


Figure 3.15: Varying directions of force applied in collecting the training dataset.

In addition, the participants were permitted to use either hand, as well as to approach the robot from a variety of directions and to stand in front of it in a variety of positions. It was established that this variability in the dataset would make it possible for the training network and the model that was produced to more represent the real-world scenarios. The detailed procedure of data collection is as follows:

- The required action will be explained to the human subject
- Human subject and human experimenter will practice the action multiple times
- The UI for the robot will be run through ROS in Ubuntu
- The UI will prompt the human subject to start and stop applying the action. This will go on to 100 samples per action. The recording starts and takes 4.5 seconds right after the prompt is given.
- The torque data for 7 joints will be saved in a bag file labelled with the name of the action and the trial count

Following the completion of the dataset collection procedure, we were left with a total of 6000 samples across all of the datasets and 1000 samples for each of the action classes. Each sample is a Rosbag [60] file that contains the relevant Rostopics [62]. These rostopics include information like the Kinova joint states (torque, position, and velocity). Each Rosbag file contains 4.5 seconds of joint-state data captured. After that, the algorithm will isolate one second of data out of the entire period of four seconds, starting from the very first instant the derivative is non-zero (in practice, we use a small value ϵ), i.e., the value starts to change. This one second date is then utilized as training data to feed the network. In summary, every action has a duration of one second, starting from the point where the action begins.

3.2.4 Training Pipeline

In this section, the results of the training pipeline is explored, which previously were explained in the theoretical background section. We note that the training was tested with two different methods, CNN and PCA-MLP. Since the training and validation results on CNN was showing good success rate than the low success rate of MLP, we skip reporting the MLP results in the thesis.

Action Classification Using CNN

The collected torque data in the Rosbag file is converted into a time series input. A sample of the torque time series that we have obtained is depicted in 3.16. After that, the time series and each of the seven joint-level sensor data are joined together to build a 1D vector. This vector is then used as the input data for our CNN classifier, which was created using the previous step's results.

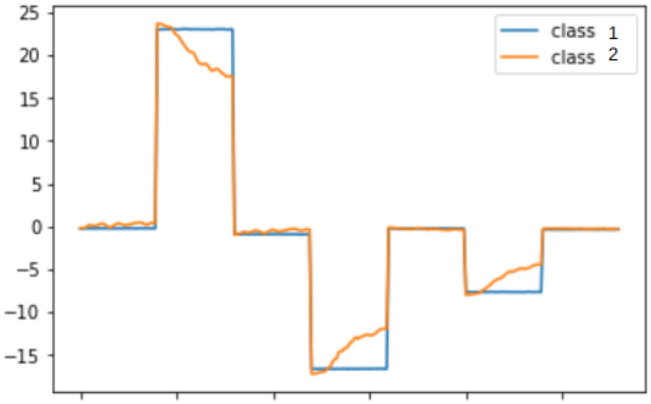


Figure 3.16: Visual representation of seven joint-level torque data appended together for two different classes [46].

We developed a deep neural network-based time series categorization based on the FCN introduced in [74]. The end-to-end nature of the suggested baseline model shows no extensive preprocessing of the raw data or feature construction. The Fully Convolutional Network (FCN) presented offers superior performance in comparison to other state-of-the-art methods. With global average pooling, the utilization of the Class Activation Map (CAM) is made possible in the convolutional model [74].

The results of FCN's semantic segmentation on photos have demonstrated impressive quality and efficiency [42]. With the help of the category-wise semantic segmentation annotation, the networks may be trained pixel-by-pixel, with each output pixel functioning as a classifier corresponding to the receptive field. The FCN is utilized here as a feature extractor for our particular problem settings. The softmax layer is still responsible for producing the final output. Convolutional layers, batch normalization layers, and ReLU activation layers make up the basic block. The convolutional layers [25] come first. The Convolution block is designed as follows:

$$\begin{aligned} y &= W \otimes x + b \\ s &= BN(y) \\ h &= ReLU(s) \end{aligned} \tag{3.1}$$

in which $\otimes$ acts as the convolution operator. The final network is designed by appending 3 convolution blocks with filter sizes of 64,64,64 [74]. Batch normalization can be applied to any layer of a neural network, including convolutional and recurrent layers. It helps improve the training speed and accuracy of the model by reducing the amount of time needed to train the model and by regularizing the model to reduce overfitting [74].

In order to achieve a lower computational burden on the processing system, this network cuts down the number of weights by using a pooling layer instead of a fully connected layer [39].

As can be seen in Figure 3.17, the network architecture is made up of four different blocks or layers. Since we are using seven torque sensors data in one second (frequency of 40 Hz) appended together, the input to the network will be 7×40 which is a 1D array of 1×280. The first three blocks are all laid out in the same manner. In the very back of the model is a Relu activation, which is preceded by a batch normalization layer that receives the output of a convolutional layer and then receives more input from that layer. Due to the fact that torque data are transformed into a one-dimensional vector, the kernel size must be a one-tuple that specifies the length of the convolution window.

We go with a kernel size of three and a filter count of sixty-four, with the latter number indicating the total number of kernels that will be convolved with the input volume. Each kernel operation will produce a one-dimensional activation map. Torque readings can range anywhere from 0 to 30 N.m. depending on the design of the robot and the actions that occur. The torque data were subjected to a batch normalization in order to generalize the information and to promote faster convergence. Concatenating three different blocks together yields the entire, complete network, which consists of a global average pooling layer at the very end, followed by a softmax classifier that labels the output.

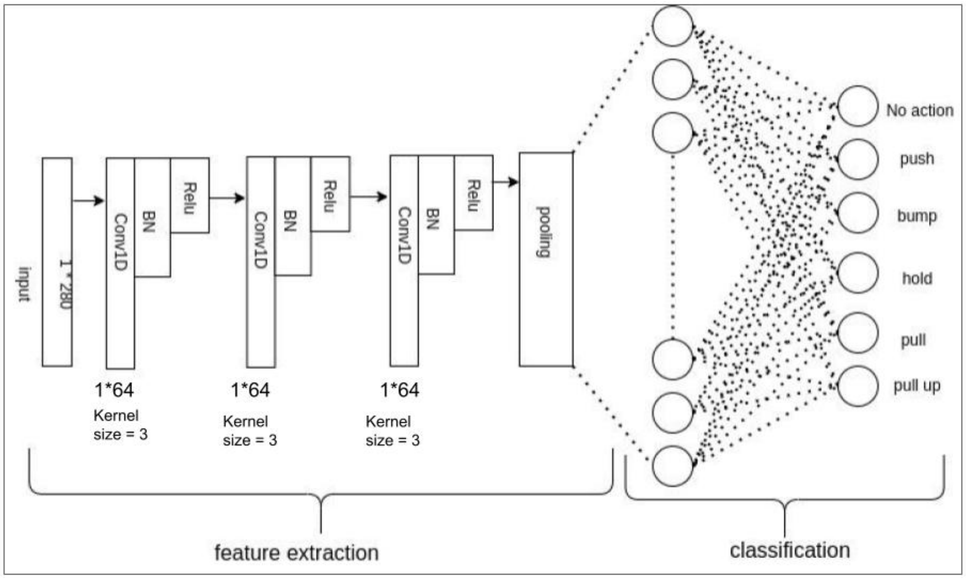


Figure 3.17: CNN network architecture for torque data classification.

After around 500 epochs, the network training loss trend gradually moves to 0. Fig. 3.18 indicates the validation and training accuracy which is based on a sparse-categorical-accuracy method [29]. According to our overall results reported in Table II, out of 180 missions (i.e., the total number of experiments for validation), our success rate in torque-based object handover was 90%. Figure 3.4 shows the hyperparameters used in the CNN model for training.

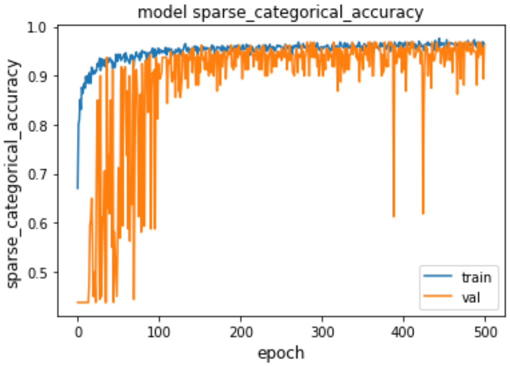


Figure 3.18: Overall training loss in CNN torque data classification.

Table 3.4: Hyperparameters used in the CNN model.

Learning Rate	0.0001
Batch size	32 per iteration
Optimizer	Adam
Epochs	500
Loss unction	Sparse Categorical Crossentropy
Validation split	0.2
Metrics	Sparse Categorical Accuracy

Torque-based Handover Block Diagram

As shown in fig 3.19, the ROS Kortex package, along with developed algorithms, manage the data processing and analysis. The torque data, which is represented as time-series data, will be fed into the CNN inference package and the action will be detected. ROS Finite State Machine then decides whether or not to release the object.

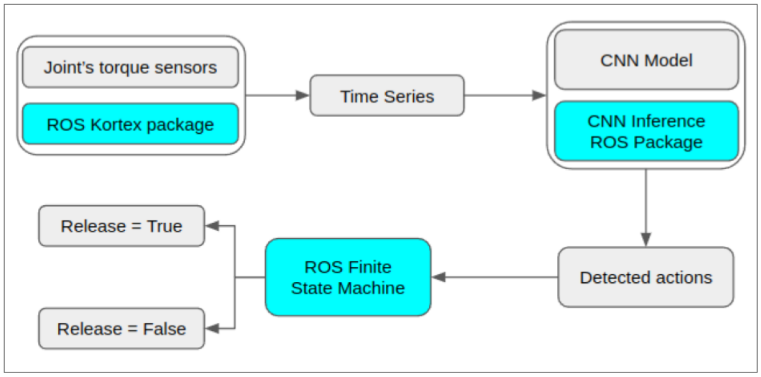


Figure 3.19: Block diagram of torque-based handover [46].

3.3 Vision-Based Object Handover

We tend to detect the human's intention of grasping or manipulating the object with RGB-D camera. The presence of the RGB-D camera plays a critical role when it comes to detecting whether the fingers of a human hand are wrapped completely around the object for grasping. It is non-trivial to determine what a human is intending to do based on the information collected from our RGB-D camera. In order to carry out an object handover that is both trustworthy and accurate, it is necessary to use a real-time inference method and accurate detection[46].

3.3.1 Dataset and Annotation

Our vision dataset consists of a comprehensive set of RGB images that are collected by the eye-in-hand RGB-D camera mounted on Kinova. Figure 3.20 illustrates the camera's setup. In order to maintain the data diversity and to increase the number of data points, we asked 10 volunteers (male: 5, female: 5) to participate in the robot-to-human object handover tasks. We recorded the rosbag files including RGB-D data from the subject's fingers while holding the object. Afterwards, we used the labelImg Python package to annotate the fingertips based on the VOC standard and prepare a fingertip dataset for feeding to SSD network[46]. The VOC standard establishes a fixed XML-based structure for the annotation of picture objects. For object detection tasks, this format is frequently used to provide ground-truth annotations. An XML file accompanies each annotated image and describes the features of the image's subjects [61].

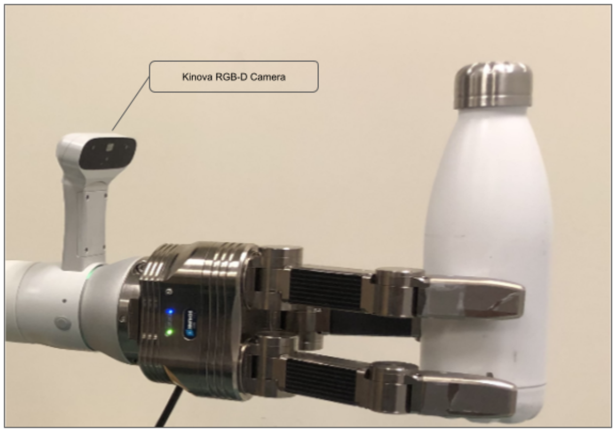


Figure 3.20: Kinova RGB-D camera is used for data collection and experiments [46].

3.3.2 Training Pipeline

Our vision algorithm is based on a Single Shot Multibox Detector (SSD) [41], which is able to function in real-time and recognize several items, in spite of the fact that the camera may partially obscure some of them. This technology is based on a deep learning architecture that enables it to rapidly identify objects and accurately detect their location in an image. SSD uses a convolutional neural network (CNN) to identify objects, predict their bounding boxes and classify them. The model is trained using large datasets with accurate annotations and optimized for accuracy and speed. The algorithm is capable of detecting objects of different sizes and shapes, as well as in varying conditions such as rotation, illumination, and occlusion. This technology can be used in a wide range of applications, including autonomous vehicles, video surveillance, and medical imaging. In addition, SSD is also capable of localizing objects in three dimensions, which makes it an ideal tool for robotics and augmented reality applications [8] [1].

SSD is a deep neural network specifically created for detecting multiple objects in RGB images. It is based on a feed-forward convolutional network which takes advantage of multiple-scale feature maps to detect objects. This base network is connected to convolutional feature layers to enable the network to recognize objects of various sizes. Fig 3.21 shows the architecture of the SSD network.

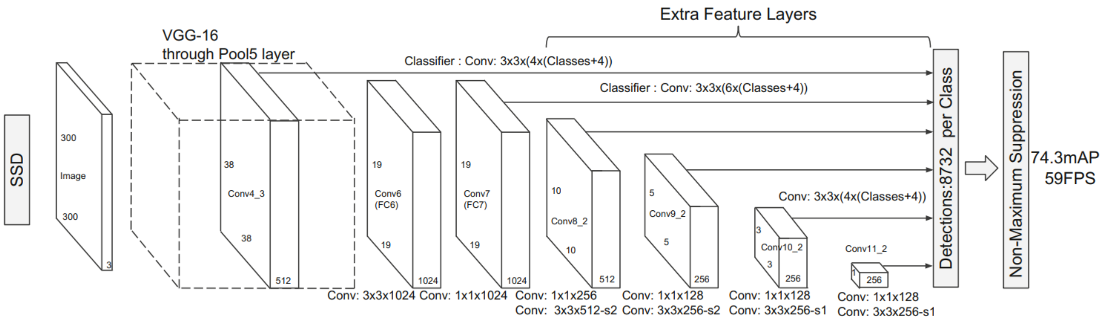


Figure 3.21: SSD network architecture [41].

The SSD model extends a base network by the addition of feature layers, which gives it the additional capability of predicting the offset among bounding boxes, aspect ratios, and the confidence scores associated with them. In each feature layer, a group of convolutional filters is positioned in order to identify a predetermined quantity of bounding boxes within the image. The ground truth information is assigned to the preset outputs, which is something that other object detectors don't do. This is one of the ways that SSD stands apart from other object detectors. From the standpoint of training, SSD was developed in such a way that it is based on the multi-box objective, which gives it the ability to recognize objects that fall into many categories [71]. Table 3.5 shows the hyperparameters used in our SSD network used for vision-based finger-tip detection. These parameters are manually tuned [41].

Table 3.5: Hyperparameters used in the SSD model [41].

Learning Rate	0.001
Weight decay	0.0005
Momentum	0.9
Batch size	32 images per iteration
Optimizer	SGD (Stochastic Gradient Descent)
Data augmentation	Random cropping, Random flipping
Loss function	Multibox loss shown in Eqn 3.22
Training iterations	24k
Intersection of Union (IOU) threshold	0.5 , 0.7
Network architecture	VGG-16 as the base architecture

The overall objective loss will be the sum of two components: the first is the localization loss (loc), and the second is the confidence loss (conf) as shown in Eq 3.2.

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \tag{3.2}$$

The value of "N" reflects the total number of pre-populated boxes. ″l″ indicates the bounding box label that was predicted, while "g" is the ground truth parameter. Moreover, a combination of the softmax-loss statistic over the classes of confidence "c" and the weight "α" is used in the calculation of the confidence loss. Throughout the course of developing our training pipeline, we carried out a number of experiments, all of which are outlined in Table 3.3. We discovered during the training tests that data augmentation led to a lower loss value and a larger mAP. This was the case despite the fact that the mAP was still increased. As a result, we relied on this model to carry out our vision-based object handover. Fig 3.22 indicates the overall loss value is SSD network training. In order to evaluate the network on the validation dataset, we calculate the Mean Average Precision (mAP) at different Intersections of Union (IOU). The results in 3.23 show the mAP@0.5/0.7 IOU [46].

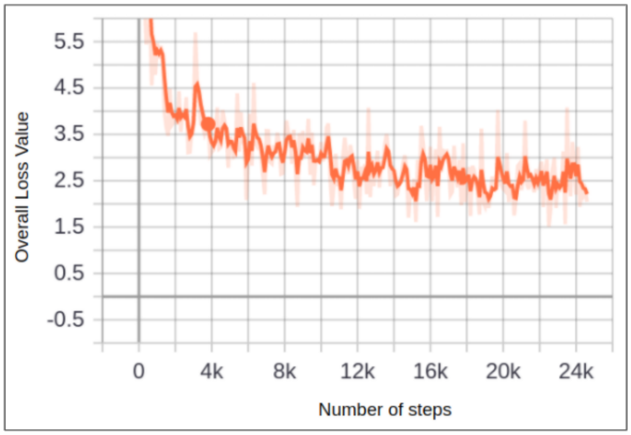


Figure 3.22: Overall training loss in the SSD training process [46].

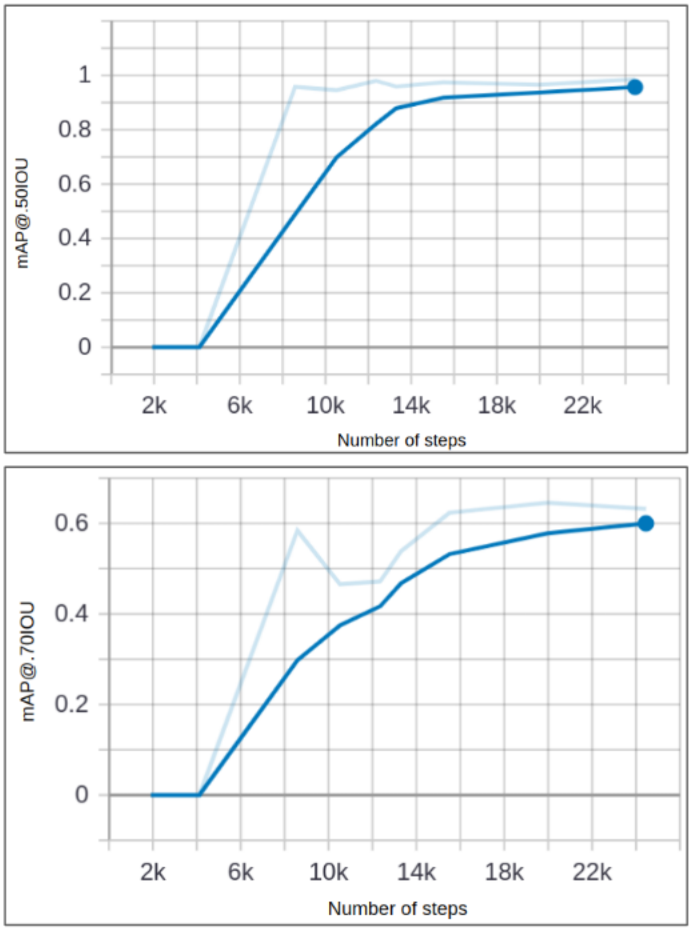


Figure 3.23: Our SSD model is evaluated on a validation dataset to extract mAP. The top plot represents the mAP@0.5 and its final value is 0.96. The bottom plot belongs to mAP@0.7 in which the final value is 0.61 [46].

3.3.3 Vision-based Object Handover Block Diagram

As shown in fig 3.24, the ROS Kortex package, along with developed algorithms, manage the data processing and analysis. The vision data, which is represented as RGB-D data, will be fed into the SSD inference package and the bounding boxes along with grasping, will be detected. ROS Finite State Machine then decides whether or not to release the object. Figure 3.25 shows the inference image of the ROS-Kortex-Vision package.

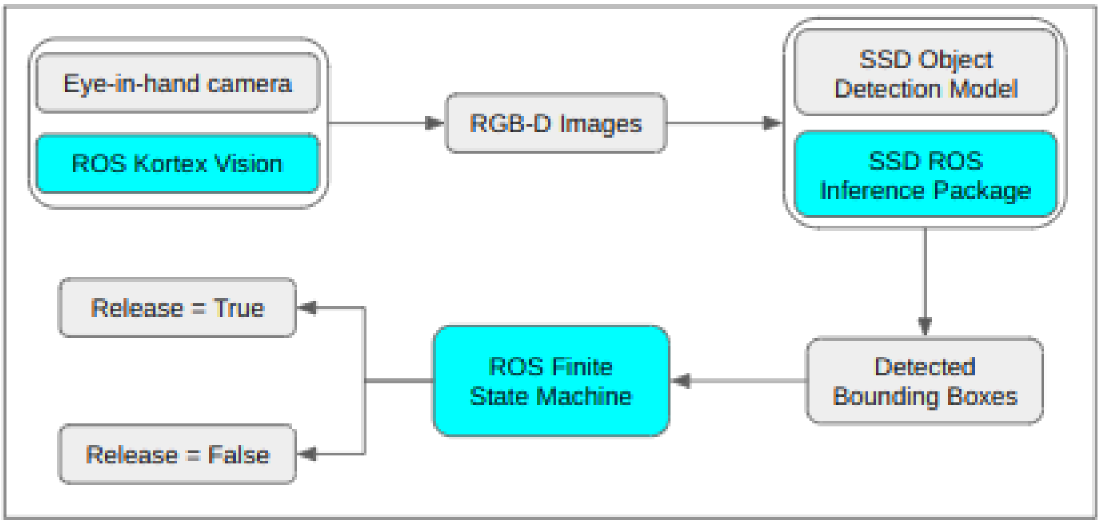


Figure 3.24: Block diagram of vision-based handover [46].

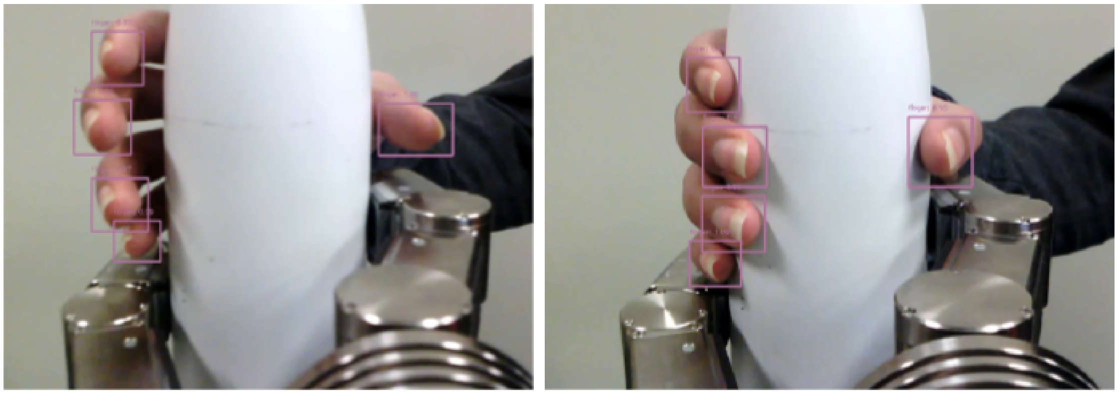


Figure 3.25: Real-time fingertips detection while the human receiver is trying to receive the object. On the left, the object is not held by the human receiver. On the right, the object is held by the human receiver [46].

It is mandatory to detect at least three fingers, including the thumb on the side and the other two, such as index and middle fingers on the other side of the object, for the SSD fingertips detector to determine a proper grasp and send out the release signal. Additionally, the positions of the fingertips must fall between the front and back planes of the object. Only then will the SSD fingertips detector function properly. Fig 3.26 shows the defined front and backplane of the object where the fingers will be placed. The SSD fingers detector makes use of the point cloud data in order to calculate an estimate of the position of each fingertip with respect to the camera frame. This information includes the x, y, and z coordinates of the bounding box around each fingertip. Then the rough position of the fingers are measured wrt the object border (front plane, back plane, side planes) to determine if the fingers are touching the object on the sides.

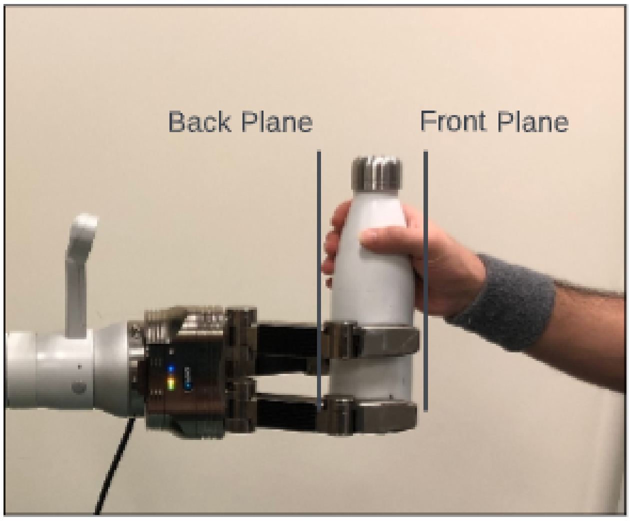


Figure 3.26: Vision-based release detection - detected fingers lie in between the front and back planes of the object [46].

3.4 Robot to Human Object Handover Using Vision and Joint Torque Sensor Modalities

In the previous two sections, CNN classifier and SSD object detection algorithms were explored, and the dataset collection, dataset annotation, and training pipeline were explained in detail. In this section, we are planning to use a fusion algorithm to fuse the results of the two detection algorithms and use them towards a safe and robust handover. By fusing the two networks' results, we tend to detect a proper grasp, detected by the vision-based algorithm, and a proper action which is precepted using the torque-based method. Fig 3.27 shows the block diagram through which the fusion algorithm is developed.

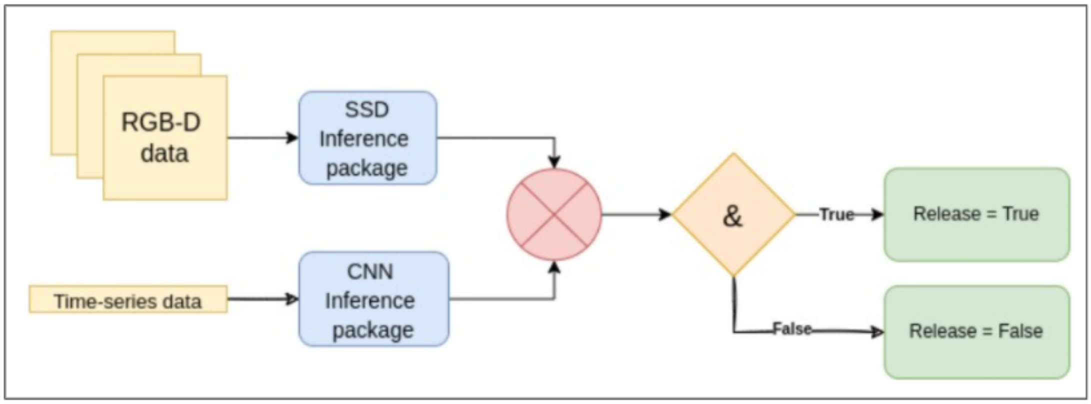


Figure 3.27: Algorithmic fusion of torque and vision data in our object handover mission.

The Vision-based SSD fingertip detection determines if the grasp has happened and the torque-based CNN classifier is applied in order to detect both the actions that have been taken as well as the contact that has been made between the hand and the item. For the purpose of determining whether or not the gripper should release, a straightforward "AND" function is applied to the outputs of both the SSD fingertip detector and the torque-based CNN classifier. The steps followed by fusion are outlined below:

- A launch file in our ROS package named "torque-vision function" will be launched to start the fusion-based algorithm handover.
- This launch file will start with 3 ROS nodes , 2 of them containing the SSD and CNN algorithm implementation. The other ROS node will be launched to subscribe to the topics published by the other 2 nodes and makes a decision for SDH action.
- A finite state machine will receive the subscribed topics from nodes that out put the results of SSD and CNN and will send the decision to the SDH laucnh file for executing the proper action (release or not release)

Since we are using two different sensors in the fusion algorithm, we need to synchronize both topics to receive them at the same rate. We have used topic-callback-synchronization functionality [63], which ensures that the torque-vision-function node can perform its tasks in a manner that is consistent with real-time.

Chapter 4

A Study of Object Handover from Robot to Human

In the previous chapter, we presented algorithms and implementation of our release controller that combines torque and visual sensing for releasing (or not) the grasped object during the physical exchange phase of object handover. We trained our system with the torque and vision dataset collected during the robot-to-human object handover. This chapter mainly describes the experiments and results of a robot-to-human object handover study that we carried out with human subjects [1]. The experiments were divided into two parts. Part A aims to compare different sensor-modality-based algorithms (Torque-based, Vision-based, Fusion-based) for success rates and failure detection, i.e., detecting different actions applied on the object by a human receiver. It shows how the combined use of joint torque and vision sensors improves the success rate in object handover and failure prevention. Part B focuses on the overall quality of the handover and evaluates different handover algorithms in terms of smoothness, ease, and similarity to human-human handover. We also compare our own methods with a baseline algorithm[12, 54]. Our conclusions are validated via statistical methods. To summarize:

- Part A tests the action detection algorithm using success rate as a validation metric
- Part B tests the object handover quality using a survey given to participants to fill out

4.1 Part A experiments

In this section, we first define the meaning of **success** and **failure** in the object handover study. An object handover is **successful** if the object is released when the human receiver applies proper action on the object and intends to receive the object. Moreover, the robot

[1] Appropriate ethical approvals have been obtained and attached to the thesis appendix B

should resist releasing if the receiver's action is, e.g., an inadvertent bump and the intention is not for receiving. As mentioned in earlier chapters, such actions could include bumping, pushing, or pulling down, either inadvertently or tricking the robot.

A **failure** in handover happens when the robot does not act as expected, i.e., it releases the object when the receiver's intention was not to receive the object, and conversely, it does not release the object, i.e., it maintains its grasp when the receiver's intention was to receive the object. In both cases, the robot misreads the cues.

This definition was then used in our experimental study to compare successful and failed handovers and hence determine the respective success rates for i) torque-based release controller, ii) vision-based release controller and iii) fusion (combining torque and vision) based release controller. We also compared our results with previous works which have used success rate as their metric for robot-to-human Object handover [46].

4.1.1 Research Questions: Controllers' Performances

The main goal of the Part A set of experiments was to compare the three different controllers in terms of accuracy in action detection and being able to differentiate between the actions. More precisely, we explored the following themes:

1. **Is the torque-based controller able to distinguish different actions?:** Our expectation is that using the CNN classifier trained with a time-series (torque) dataset, the controller will be able to detect various actions and hence provide a proper release (or not) action on the gripper.

2. **Is the vision-based controller able to determine whether or not the object has been grasped?:** Using the SSD network trained with the collected RGB-D dataset, fingertips and their location around the object are expected to be detected. We assume that if three different fingers are detected on the sides of the object, it has been grasped by those fingers, and it will be later shown in the results that vision-based release controller will rely on this information to release the object.

3. **Does combining torque and vision algorithms' outputs result in a better success rate for the release controller?:** Using the CNN classifier output to determine the action applied on the object, plus having the location of the fingertips surrounding the object, will probably provide us with a better result. Please note that since the torque-based release controller detects actions as well, it will also give us information about the direction range of the action.

4.1.2 Method and Experimental Setup

The robot, consisting of the Kinova Gen3 arm+SDH gripper, set on one of three different modes, including torque-based, vision-based, and fusion-based, held a white bottle with its

Table 4.1: Each action in the object handover mission is evaluated using the following predefined success criteria [46].

actions	success criteria
no action	Do not release the object
bump	Do not release the object
push	Do not release the object
hold	Release the object smoothly
pull	Release the object smoothly
pull-up	Release the object smoothly

gripper. Each human subject, one of the 30 volunteers from Simon Fraser University who participated in this study, was asked to apply the six predefined actions on the object intentionally, as stated in a table 4.1. These actions were demonstrated live by the experimenter in the lab, and subjects learnt how to apply the actions to the object. They were not informed about the type of controller being used. The total duration of the experiments (for all the controllers) for each subject was approximately 15-20 minutes.

During the experiments, each subject stood in front of the robot, and a prompt appeared on the monitor instructing them to apply a specified action on the object. The release controller analyzed the data and determined whether the SDH should release or hold the object. Depending on the action taken, the result was categorized as a success or failure, as defined in Section 4.1 above.

If the controller detected that the actions were intended for receiving, it would command the SDH controller to open the fingers with a constant speed. The experimenter (researcher) recorded the success or failure of each experiment within each mode just after the subject performed the action.

4.1.3 Subject Recruitment Email

For viewing the subject recruitment email, please refer to the Appendix B. Recruitment advertisements were announced orally to SFU engineering students in various locations on the campus and were also posted on the social media website Facebook.

4.1.4 Results

The results are shown in the two tables 4.2 and 4.3. Table 4.2 shows how well different actions are detected by the three controllers. The results show that the fusion-based algorithm is more reliable in terms of success rate.

The success rate in correctly detecting the actions in the torque-based algorithm is 90 %. It shows a good performance in detecting different actions based on torque data. However, if the receiver tries to manipulate the object, i.e. pulling the object with one finger, it fails to detect whether or not the object being pulled is grasped fully or manipulated. We note

Table 4.2: Success rate in detecting various actions by the three controllers: torque-based, vision-based, and fusion-based. "s" and "f" indicate "success" and "failure", respectively [46].

action type	number of trials	torque-based		vision-based		fusion-based	
		s	f	s	f	s	f
no action	30	30	0	30	0	30	0
bump	30	25	5	28	2	30	0
push	30	27	3	0	30	29	1
hold	30	24	6	28	2	29	1
pull	30	28	2	30	0	30	0
pull-up	30	28	2	26	4	29	1

that the controller has a reduced success rate in detecting bump and hold. One plausible explanation is that detecting hold is based on small vibrations from the human's hand, which are likely difficult for the network to differentiate hold from bump. The results are derived from Table 4.2

The vision-based controller, on the other hand, has a lower success rate which is 79 %. In the vision-based object handover experiment, the algorithm has zero success in detecting push but happens to detect rest of the actions. This is because the controller is only able to detect fingers, so it confuses between them and detects all the actions, which includes fingers on the two sides of the object, as hold and releases the object. A plausible explanation is that since the controller does not detect applied force (magnitude/direction), it relies on the fingers' positions with respect to the object to send the decision for release. The results are shown in 4.2 and 4.3.

The fusion-based controller combined torque-based information with vision-based information and they complement each other and it results in a higher success rate for each action. As shown in Table 4.2, while the torque-based handover fails to detect bump vs hold with a good success rate, the vision sensor is better able to distinguish bump vs hold, because in the former, all three fingers are not involved, whereas in the latter, they are. On the other hand, when vision-based handover fails to detect the difference between push and pull, the torque-based algorithm comes to play and detects the action. Hence overall, we get much better results with the fusion-based controller.

Table 4.3 shows the overall success rates (cumulative over all actions) for the three controllers. Indeed, as expected, the fusion-based controller that combines torque and vision data has the highest success rate.

4.2 Part B experiments

This set of experiments was carried out to compare different controllers from a receiver's perspective, i.e., smoothness, ease of taking the object, and the similarity to human-to-

Table 4.3: Overall comparison of success rate among the three different controllers in R2H object handover [46].

system	Num of trials	Num of successful handovers	s rate
Torque	180	162	90%
vision	180	114	79%
fusion	180	177	98%

human handover. Besides our three controllers, we also incorporate a fourth controller that calculates the force applied at the end effector (via Jacobian) and is a proxy for a wrist-mounted force-torque sensor, which has been used in some previous works in the literature.

4.2.1 Research Questions and Hypothesis

More specifically, we were interested in answering the following research questions regarding the overall quality of R2H transfer with respect to the different release controllers:

1. **Does the release controller result in a stable handover (i.e. without dropping the object)?** Using the release controller, the object should be safely, i.e., without dropping, transferred from the robot's gripper to the human's hand. The metric used is the likelihood (or probability) of the object being dropped before the human has achieved a stable grasp on the object. The subjects (human receivers) in the experiments were asked to rate the chances of the object being dropped during the handover process.

2. **How easy is it to receive the object from the robot?** The release controller should provide the human receiver with an easy handover in which the receiver won't need to apply an excessive force (beyond the necessary force for a usual human to human handover) to get the full possession of the object. Subjects were asked to rate the ease of the transfer process.

3. **How smooth is the handover?** The human receiver should be able to get the object without waiting for an unduly long time for the robot's response. If the robot delays the process of handover, it may confuse the human receiver and result in various unknown consequences like the receiver may change his/her mind about receiving the object, or they may put too much pressure on the object. In these cases, the robot might break or the object may be dropped.

4. **Does the handover system provide a natural human-like handover?** It's supposed to evaluate the overall quality of the R2H handover compared to H2H handover.

4.2.2 Method and Experimental Setup

30 healthy adults (graduate and undergraduate students from Simon Fraser University (15 males and 15 females) participated in a handover study with the same experimental setup as was used in the earlier experiment, i.e., the same water bottle was the object to be handed over and the system consisted of the same 7 DOF kinova arm with the same SDH 3-finger gripper mounted on it.(Please refer to Appendix Bfor the consent form). Human subjects were asked to receive the object five times in each different mode. While the sensor data were measured through the ROS platform, the kinova+SDH robot handed over a white water bottle to the subjects and they will rate each controller separately in the survey to express their evaluation of each controller. Then these survey responses are analyzed through hypothesis testing, and the results are explained. The survey analysis is then conducted in pairs (torque vs force, torque+vision vs force+vision, and etc.) to show a pairwise comparison.

4.2.3 Survey Design

After each experiment, the participants were asked to fill out a questionnaire as provided below. Each question is asking the subject to evaluate a feature in the resulting handover as compared to human-to-human handover on a five-point Likert scale [28]. The questions are as follows:

- Rate the likelihood that the object could have been dropped during the handover (1 – Likely to be dropped, 5- Not likely to be dropped).
- Rate the resistance you felt in the handover (1 – Too resistant, 5 – Very compliant).
- Rate the smoothness in the handover (1- not smooth at all, 5- very smooth)
- Rate the overall quality of the handover (1- very different, 5- very similar)

4.2.4 Subject Recruitment Email

For viewing the subject recruitment email, please refer to the appendix B. Recruitment advertisements were announced orally to SFU engineering students in various locations on the campus and on the social networking website Facebook to recruit volunteers.

4.2.5 A Brief Synopsis of Relevant Data Analysis Methods

Based on the research questions we had in the previous section, we have proposed some hypotheses to explore during the R2H object handover.

${H_1}^R$ = Users prefer the fusion algorithm, i.e., vision+torque controller as compared to the other 2 controllers, i.e., solely torque based or solely vision based.

${H_2}^R$ = The vision-based controller will provide the smoothest handover among all the controllers

${H_3}^R$ = The fusion algorithm controller will be least likely to drop the object among all the 3 controllers

${H_4}^R$ = The torque-based handover will work smoother than force-based object handover

In order to test and validate the hypotheses mentioned above, we analyzed survey responses for each paired controller. The subjects rated the release controller on a Likert scale data. The Sign Test, the Wilcoxon Signed-Rank Test and t-test are three appropriate methods for analyzing data obtained from the Likert scale [44] [45]. Since we plan to perform paired comparisons for the data that fall into the interval category, i.e. the participants have to choose between 1 to 5 to rate each controller, then these tests are appropriate.

The Sign Test and Wilcoxon Signed-Rank Test are used when it is not known if the data distribution is normal. Both these tests perform a non-parametric test that does not require any assumptions about the distribution of the data. However, the t-test is a parametric test that assumes the data are normally distributed and the variances of the two groups being compared are equal (in our case it's paired t-test). The Sign Test is a quick and easy way to examine paired data, but it lacks the precision of more advanced methods like the Wilcoxon Signed-Rank Test

The Sign Test is used when comparing two related samples or paired observations. It determines whether the positive and negative differences between paired observations are equally likely to occur. It compares the number of positive and negative signs to a critical value from the binomial distribution. If there are more positive differences than negative differences, then the alternative hypothesis that the medians are different is accepted and the null hypothesis of equal medians is rejected.

Wilcoxon signed-rank test is used to assess whether the medians of paired differences between two related samples are significantly different. It ranks the absolute differences between paired observations and, unlike Sign Test, ignores the signs.

Both Wilcoxon and sign test can be used with ordinal, interval, or ratio data.

On the other hand, T-test is used to compare the means of paired observations but also two independent samples. T-test calculates the t-statistic, which measures the difference between the sample means relative to the variability within the groups. It compares the t-statistic to a critical value from the t-distribution to determine statistical significance.

It is most commonly used with interval or ratio data, but it can also be applied to approximately normally distributed ordinal data [44].

Two main types of t-tests are i) Independent samples t-tests, and ii) paired samples t-tests (In our case, it's paired samples). When comparing two samples that do not come from the same population or reflect separate experiments, the independent samples t-test is appropriate. Otherwise, the paired samples t-test is performed.

In all three methods, the null hypothesis that there is no difference between the samples is rejected in favour of the alternative hypothesis that there is a difference between the samples if and only if the test statistic is larger than a critical value. Tables of critical values, often known as p-values, are used to establish the threshold of statistical significance.

Since the Wilcoxon sign rank test gives the absolute difference between the medians of two samples and is used for data with unknown distribution, we use it for our testing. Even though our data distribution is likely not normal, we nevertheless performed paired t-test and the results we got were the same.

4.2.6 Results

Fig 4.1 shows the boxplot of the survey responses for Q1, i.e, liklihood of dropping the object. The same data is shown in tabular form in Table 4.4. Pairwise comparison of the four release controllers via respective t-tests is shown in Table 4.5 and 4.6, shows that there is no statistically significant difference between controllers for likelihood of dropping the object, as rated by the subjects. It is important to note that the v-value and the p-value are the measures of how similar the values are to one another. The greater the distance that v-value is from zero, the lower the p-value, which indicates that the medians are more different. [45].

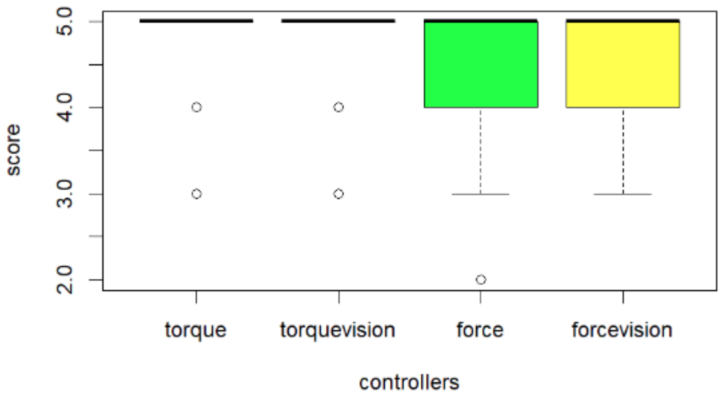


Figure 4.1: Boxplot generated based on survey responses to Q1.

Table 4.4: Summary of survey responses to the likelihood that the object could have been dropped based on 5 receivings per each controller (question 1 of the survey).

controller method	Min	1st Qu.	Median	Mean	3rd Qu.	Max
Torque-based	3	5	5	4.792	5	5
Torque+vision	3	5	5	4.708	5	5
Force-based	2	4	5	4.5	5	5
Force+vision	3	4	5	4.5	5	5

Table 4.5: Pairwise (controllers) comparison of different controllers for receiver rated likelihood of dropping the object with paired t-test.

controller method	Null Hypothesis	Conf - level	p - value	result
T , T+V	$M_0 = M_1$	0.95	0.4912	Fail to reject
F , F+V	$M_0 = M_1$	0.95	1	Fail to reject
F , T	$M_0 = M_1$	0.95	0.1097	Fail to reject
F+V , T+V	$M_0 = M_1$	0.95	0.096	Fail to reject

Table 4.6: Pairwise (controllers) comparison of different controllers for receiver rated likelihood of dropping the object with Wilcoxon test.

controller method	Null Hypothesis	Conf - level	v-value	p - value	result
T , T+V	$M_0 = M_1$	0.95	10	0.5716	Fail to reject
F , F+V	$M_0 = M_1$	0.95	14	1	Fail to reject
F , T	$M_0 = M_1$	0.95	13	0.1373	Fail to reject
F+V , T+V	$M_0 = M_1$	0.95	0	0.1736	Fail to reject

Fig 4.2 shows the boxplot of the survey responses for Q2, i.e., receiver rated ranking of smoothness of object transfer. The same information is shown in tabular form in

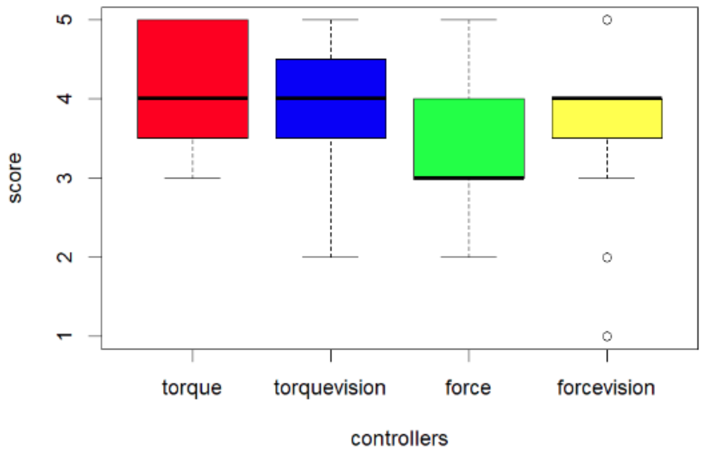


Figure 4.2: Boxplot generated based on survey responses to Q2.

Table 4.7. Table 4.8 and Table 4.9 shows the pairwise comparison of human receiver perceived ranking of smoothness of object handover via t-test analysis. It shows that there is no difference between controllers in terms of smoothness of the handover except between force-based and torque-based controllers. It shows that the torque-based controller performs significantly better than the force-based controller in terms of receiver perceived smoothness of object handover. A plausible explanation is that since the torque-based controller

is trained with lots of training data in different directions and various magnitudes, its performance in detecting the action and releasing the object is quicker than the force-based method.

Table 4.7: Summary of survey responses to rank smoothness in the handover based on 5 receiving per each controller (question 2 of the survey).

controller method	Min	1st Qu.	Median	Mean	3rd Qu.	Max
Torque-based	3	3.75	4	4.208	5	5
Torque+vision	2	3.75	4	3.958	4.25	5
Force-based	2	3	3	3.33	4	5
Force+vision	1	3.75	4	3.708	4	5

Table 4.8: Pairwise (controllers) comparison of different controllers for receiver rated the smoothness in the handover with paired t-test.

controller method	Null Hypothesis	Conf - level	p - value	result
T , T+V	$M_0 = M_1$	0.95	0.2656	Fail to reject
F , F+V	$M_0 = M_1$	0.95	0.1535	Fail to reject
F , T	$M_0 = M_1$	0.95	0.004495	Reject
F+V , T+V	$M_0 = M_1$	0.95	0.2826	Fail to reject

Table 4.9: Pairwise (controllers) comparison of different controllers for receiver rated the smoothness in the handover with Wilcoxon test.

controller method	Null Hypothesis	Conf - level	v-value	p - value	result
T , T+V	$M_0 = M_1$	0.95	68.5	0.3027	Fail to reject
F , F+V	$M_0 = M_1$	0.95	64.5	0.1069	Fail to reject
F , T	$M_0 = M_1$	0.95	54.5	0.009515	Reject
F+V , T+V	$M_0 = M_1$	0.95	32	0.3447	Fail to reject

Fig 4.3 shows the boxplot of the survey responses for Q3, i.e., receiver rated ranking of ease of taking the object handover.

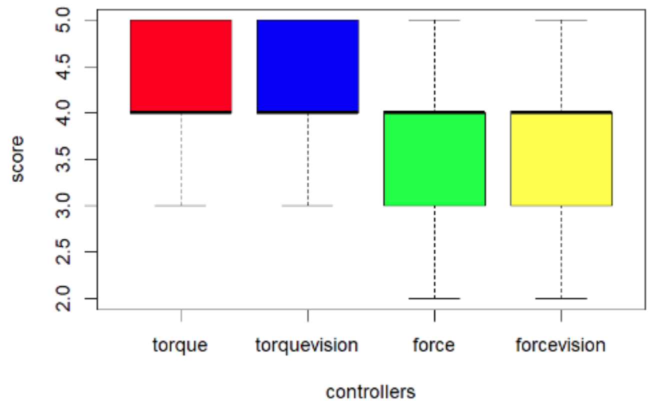


Figure 4.3: Boxplot generated based on survey responses to Q3.

The same information is shown in tabular form in Table 4.10. Table 4.11 and 4.12shows the pairwise comparison of human receiver perceived ranking of ease taking of object handover via paired t-test analysis. It shows that there is no difference between controllers in terms of ease of taking the object in the handover process except between force-based and torque-based controllers. It shows that the torque-based controller performs significantly better than the force-based controller in terms of receiver perceived ease of taking in object handover.

Table 4.10: Summary of survey responses to the rate the ease of taking in the handover based on 5 actions per each controller (question 3 of the survey).

controller method	Min	1st Qu.	Median	Mean	3rd Qu.	Max
Torque-based	3	4	4	4.25	5	5
Torque+vision	3	4	4	4.25	5	5
Force-based	2	3	4	3.625	4	5
Force+vision	2	3	4	3.792	4	5

Table 4.11: Pairwise (controllers) comparison of different controllers for receiver rated the ease of taking the object in handover with paired t-test.

controller method	Null Hypothesis	Conf - level	p - value	result
T , T+V	$M_0 = M_1$	0.95	1	Fail to reject
F , F+V	$M_0 = M_1$	0.95	0.3824	Fail to reject
F , T	$M_0 = M_1$	0.95	0.006081	Reject
F+V , T+V	$M_0 = M_1$	0.95	0.4883	Reject

Table 4.12: Pairwise (controllers) comparison of different controllers for receiver rated the ease of taking the object in handover with Wilcoxon test.

controller method	Null Hypothesis	Conf - level	v-value	p - value	result
T , T+V	$M_0 = M_1$	0.95	33	1	Fail to reject
F , F+V	$M_0 = M_1$	0.95	30	0.4644	Fail to reject
F , T	$M_0 = M_1$	0.95	20.5	0.01055	Reject
F+V , T+V	$M_0 = M_1$	0.95	14	0.008457	Reject

Fig 4.4 shows the boxplot of the survey responses for Q4, i.e., the receiver rated the whole similarity to human-to-human handover.

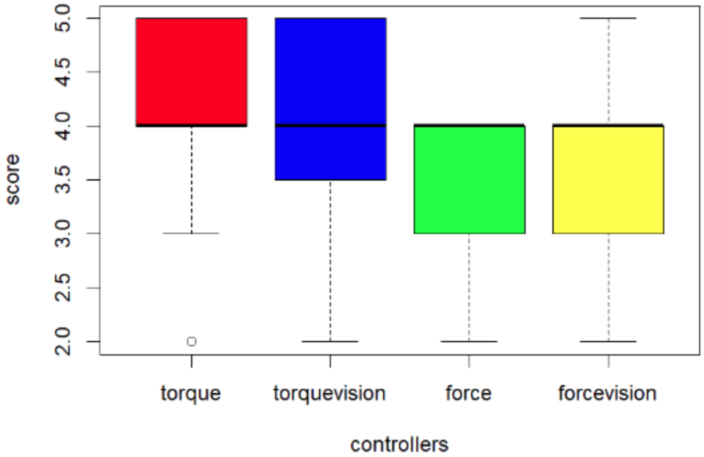


Figure 4.4: Boxplot generated based on survey responses to Q4.

The same information is shown in tabular form in Ta- ble 4.13. Table 4.14 and 4.15shows the pairwise comparison of human receiver perceived ranking of similarity to human-to-human handover via paired t-test analysis. It shows that the torque-based controller performs better than the force-based controller in terms of similarity to human-to-human handover.

Table 4.13: Summary of survey responses to the rate the whole similarity to human-to-human handover based on 5 actions per each controller (question 4 of the survey).

controller method	Min	1st Qu.	Median	Mean	3rd Qu.	Max
Torque-based	2	4	4	4.125	5	5
Torque+vision	2	3.75	4	4.042	5	5
Force-based	2	3	4	3.5	4	4
Force+vision	2	3	4	3.792	4	5

Table 4.14: Pairwise (controllers) comparison of different controllers for receiver rated the whole similarity to human-human handover with paired t-test.

controller method	Null Hypothesis	Conf - level	p - value	result
T , T+V	$M_0 = M_1$	0.95	0.5385	Fail to reject
F , F+V	$M_0 = M_1$	0.95	0.03156	Reject
F , T	$M_0 = M_1$	0.95	0.0002621	Reject
F+V , T+V	$M_0 = M_1$	0.95	0.1617	Fail to reject

Table 4.15: Pairwise (controllers) comparison of different controllers for receiver rated the whole similarity to human-human handover with Wilcoxon test.

controller method	Null Hypothesis	Conf - level	v-value	p - value	result
T , T+V	$M_0 = M_1$	0.95	17.5	0.5877	Fail to reject
F , F+V	$M_0 = M_1$	0.95	4	0.04183	Reject
F , T	$M_0 = M_1$	0.95	18	0.001158	Reject
F+V , T+V	$M_0 = M_1$	0.95	22.5	0.179	Fail to reject

4.3 Discussion

In summary, based on the two human receiver studies, reported in part A and part B, respectively, we may draw the following conclusions:

- Fusion-based controller that combines torque sensor data with wrist-mounted RGB camera sensor (vision sensor) gives the best success rate for object transfer.
- The torque-based controller trained with a large dataset of different directions and values is rated higher than (simulated) force-based controller in terms of all receiver perceived qualities - the likelihood of dropping the object, smoothness, ease of transfer and similar to human-to-human transfer. We believe that the torque-based controller's effectiveness in detecting the action and releasing the object is superior to the force-based method since it is trained with a large amount of training data in a variety of orientations and magnitudes.
- Fusing two different sensors for performing a handover yields better results by exploiting the complementarity of the sensed information of the force/torque and vision sensors. Fusion-based controllers (T+V, and F+V) are perceived by receivers as closer to human-to-human handover.

Chapter 5

Conclusions and Future Work

Robot-to-human object handover, R2H, is normally divided into 3 phases - pre-handover, physical exchange, and post-handover. At this stage, we focused on the physical exchange phase and tried to improve the success rate by fusing the results of 2 different sensors' data analysis and also by preventing the dropping of the object in failure situations. Many works have aimed at proposing R2H handover to improve the smoothness, ease of taking, and similarity to human-to-human handover. In order to achieve this, some have characterized their release controllers mostly inspired by human-human handover experiments and how the giver and receiver gradually change their force to perform the physical exchange of the object. Although some works have explored failure detection after the physical exchange, none of the works explicitly addresses inadvertent actions that may occur as the exchange takes place - actions such as inadvertent bumping, by the receiver's hand. Such actions clearly should not be interpreted as grasping actions by the giver, and the release controller should not release the object. In extreme cases, such actions may even be voluntary on the part of the receiver to deceive the robot and manipulate the object. Also, if the giver only relies on haptic sensing (torque and force) for object handover, its ability to differentiate between manipulation or proper grasp by the receiver reduces. We carried out some experiments to explore the importance of vision in detecting manipulation and proper grasp. However, we relied on the same set of experiments which were done in [55]. To address these two issues, we used two different sensor modalities, torque and vision, and introduced a fusion method for real-time R2H object handover on a robotic system, including a 3-finger Schunk Dexterous gripper and 7-degrees-of-freedom Kinova arm.

We explored the following issues:

- We used torque sensors embedded in the Kinova Gen 3 arm to differentiate between different actions performed on the object by the human receiver. We collected a large dataset of torque data while the robot is in the home position holding the object, and the human receiver tries to apply different actions on the object. The torque dataset is then fed into a CNN to be classified and used for our release controller. Then for the experiments, We brought 30 subjects to validate the controller. The results are promising in terms of accuracy, with more than a 90 % success rate in action detection. However, as expected, torque sensing fails to detect the manipulation, which in our case means whether the object is being fully grasped or not.

- Using a wrist-mounted RGBD camera, we detected the receiver's fingertips around the object, which was deemed to the receiver having grasped the object. By incorporating depth and RGB information from the vision sensor, we collected a big dataset of different fingertips captured around the object while being grasped by the SDH gripper during robot-to-human object handover and performed an object detection algorithm developed by SSD network. To evaluate the controller, we asked the same participants to apply the same actions on the object. The results emphasize that the vision-based algorithm, although able to detect the grasp around the object, cannot differentiate between different directions of the forces applied to the object. success rate of 76 % shows the same conclusion.

- Finally, the torque-based information and the vision-based information were fused together using a ROS Finite State Machine. We combined the output results of both classifiers and fed them into our release controller. When applying our proposed fusion algorithm to the R2H handover missions, we see a significant increase in action detection and success rate. This has been shown in our result, showing a 98 % success rate in robot-to-human object handover.

5.1 Future work

While we have shown how the fusion of vision and torque sensor data in the release controller helps improve the success of object handover, it's worth mentioning that our work has some limitations that should be addressed in future works.

- Specific handover configuration: The Kinova gen3 consists of seven DOF and accordingly seven joint-level torque sensors. So for a given hand pose, there are many possible join configurations for R2H handover. Our training data and test data was limited to one specific configuration (shown in Figure 1.3) mainly due to time limitations.

- Our proposed method makes use of joint-level torque sensors, which may not be available in many robotic manipulators. In such cases, one would use a wrist mounted force/torque sensor.

- Using time-series data from torque sensors, we recorded 1 second of data for the CNN network, and it takes 1 second (not considering the computational burden of the running programs) for the software platform to detect the actions from the receiver, which is not real-time.

- Although SDH is endowed with tactile sensors on the fingers, our proposed algorithm has not used tactile sensors for action detection.
- Our proposed vision-based controller is only tested with the cylindrical white bottle. We have not evaluated our algorithm with other objects that are usually used in R2H handover.

Considering the limitations mentioned above, following directions for future work are suggested.

We believe that our method, in principle, is generalizable to multiple set of joint configurations, however, this will need a substantially larger set of training data for many different configurations. It should also be explored how the 1-second latency in action detection can be reduced to make the algorithm more real-time. Another key future work direction would be to use tactile data (in addition to torque data) for action detection. Moreover, tactile data can provide additional useful information such as the location of contact between the fingers and the object and potentially detect if the object slips in the hand. One could combine the previous work from our lab [**?**] that detects actions based on tactile data with our work to achieve more robust results. Although the white bottle has been utilized as a canonical object (a picture of a bottle is presented in Figure 1.2 in Chapter 3), it is important to emphasize that our method is not object-specific and can be extended to a wide range of objects. Our SSD model can be trained for identifying graps on a variety of objects and the torque-based CNN classifier relies on changes in joint torques and therefore it will likely be not too sensitive to object geometry. In any event, it can also be trained for a variety of objects. In future, multi-object experiments will be a fruitful direction to pursue [46].

The fusion algorithm could definitely be improved upon. For instance, we could consolidate the two separate networks into a single network using deep sensing fusion techniques [23][30][19]. Also, the results from the two networks could be weighted and then make the release decision. The failures that happened during our experiments could be further refined and explored by considering false positive and false negative situations. In many situations, it seems the consequences of a false positive (release the object when it should not be) such as dropped object are more costly than a false negative (not release the object when it should be). Additionally, we could also incorporate the depth data captured by the RGB-D camera and use semantic segmentation to decide whether or not the fingers are touching the item, and, if they are, where they are making contact. There is current work going on in our lab exploring this issue.

In terms of the pre-handover phase, The RGB-D data can also aid in estimating the receiver's hand velocity and detecting its gesture, which can be taken as further evidence of the receiver's purpose. Another improvement that can be made in future studies is to use a compliant mode with respect to the human receiver, which makes the handover even smoother and safer. Coupling this physical exchange stage with pre-handover and post-handover stages to build a fully autonomous handover is our longer-term goal. Some of these aspects are being pursued and studied by current graduate students in the RAMP lab.

Bibliography

[1] Ssd-object-detection-tfod-training-pipeline. https://github.com/Bmoradi93/SSD-Object-Detection-TFOD-Training-Pipeline, 2021.

[2] Acin. Schunk-SDH-2. `https://www.acin.tuwien.ac.at/en/industrial-robotics/robots/`.

[3] Arash Ajoudani, Andrea Maria Zanchettin, Serena Ivaldi, Alin Albu-Schäffer, Kazuhiro Kosuge, and Oussama Khatib. Progress and prospects of the human–robot collaboration. *Autonomous Robots*, 42(5):957–975, 2018.

[4] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31(3):606–660, 2017.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[6] Patrizia Basili, Markus Huber, Thomas Brandt, Sandra Hirche, and Stefan Glasauer. Investigating human-human approach and hand-over. In *Human centered robot systems*, pages 151–160. Springer, 2009.

[7] Mustafa Gokce Baydogan, George Runger, and Eugene Tuv. A bag-of-features framework to classify time series. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2796–2802, 2013.

[8] Behnam Moradi. SSD-Object-Detection-ROS2. `https://github.com/Bmoradi93/librealsense`, 2021. Online; Github repository.

[9] Philippe C Besse, Brendan Guillouet, Jean-Michel Loubes, and François Royer. Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 17(11):3306–3317, 2016.

[10] Aude Billard and Danica Kragic. Trends and challenges in robot manipulation. *Science*, 364(6446):eaat8414, 2019.

[11] Jonathan Bohren, Radu Bogdan Rusu, E Gil Jones, Eitan Marder-Eppstein, Caroline Pantofaru, Melonee Wise, Lorenz Mösenlechner, Wim Meeussen, and Stefan Holzer. Towards autonomous robotic butlers: Lessons learned with the pr2. In *2011 IEEE International Conference on Robotics and Automation*, pages 5568–5575. IEEE, 2011.

[12] Wesley P Chan, Chris AC Parker, HF Machiel Van der Loos, and Elizabeth A Croft. A human-inspired object handover controller. *The International Journal of Robotics Research*, 32(8):971–983, 2013.

[13] Marco Costanzo, Giuseppe De Maria, and Ciro Natale. Handover control for human-robot and robot-robot collaboration. *Frontiers in Robotics and AI*, 8:672995, 2021.

[14] John J Craig. *Introduction to robotics.* Pearson Educacion, 2006.

[15] YL Cun, L Bottou, G Orr, and K Muller. Efficient backprop, neural networks: Tricks of the trade. *Lecture notes in computer sciences*, 1524:5–50, 1998.

[16] Mohammad-Javad Davari, Michael Hegedus, Kamal Gupta, and Mehran Mehrandezh. Identifying multiple interaction events from tactile data during robot-human object transfer. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–6. IEEE, 2019.

[17] Travis Deyle, Hai Nguyen, Matthew Reynolds, and Charlie Kemp. Rfid-guided robots for pervasive automation. *IEEE Pervasive Computing*, 9(2):37–45, 2010.

[18] A Gomez Eguiluz, Iñaki Rañó, Sonya A Coleman, and T Martin McGinnity. Reliable object handover through tactile force sensing and effort control in the shadow robot hand. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 372–377. IEEE, 2017.

[19] Wilfried Elmenreich. An introduction to sensor fusion. *Vienna University of Technology, Austria*, 502:1–28, 2002.

[20] John Cristian Borges Gamboa. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017.

[21] Mamoun Gharbi, Pierre-Vincent Paubel, Aurélie Clodic, Ophélie Carreras, Rachid Alami, and Jean-Marie Cellier. Toward a better understanding of the communication cues involved in a human-robot object transfer. In *2015 24th IEEE international symposium on robot and human interactive communication (RO-MAN)*, pages 319–324. IEEE, 2015.

[22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT press, 2016.

[23] Raffaele Gravina, Parastoo Alinia, Hassan Ghasemzadeh, and Giancarlo Fortino. Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges. *Information Fusion*, 35:68–80, 2017.

[24] Michael Hegedus, Kamal Gupta, and Mehran Mehrandezh. Towards an integrated autonomous data-driven grasping system with a mobile manipulator. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1601–1607. IEEE, 2019.

[25] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[26] Irjensen. Multivariate Time-series Classification. `https://github.com/irjensen/multivariate-timeseries-classification`.

[27] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[28] Ankur Joshi, Saket Kale, Satish Chandel, and D Kumar Pal. Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396–403, 2015.

[29] keras. SparseCategoricalAccuracy. `https://github.com/keras-team/keras/blob/v2.13.1/keras/metrics/accuracy_metrics.py#L177`, 2023.

[30] Jihun Kim, Dong Seog Han, and Benaoumeur Senouci. Radar and vision sensor fusion for object detection in autonomous vehicle surroundings. In *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 76–78. IEEE, 2018.

[31] kinovarobotics. https://www.kinovarobotics.com/, 2023.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012 alexnet. *Adv. Neural Inf. Process. Syst*, pages 1–9, 2012.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[34] James Large, Jason Lines, and Anthony Bagnall. The heterogeneous ensembles of standard classification algorithms (hesca): the whole is greater than the sum of its parts. *arXiv preprint arXiv:1710.09220*, 2017.

[35] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[36] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[37] Yann LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–50. Springer, 2002.

[38] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[39] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[40] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[41] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[42] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[43] Andrea H Mason and Christine L MacKenzie. Grip forces when passing an object to a partner. *Experimental brain research*, 163(2):173–187, 2005.

[44] John H McDonald. *Handbook of biological statistics*, volume 2. sparky house publishing Baltimore, MD, 2009.

[45] John H McDonald. *Handbook of biolological statistics.* New York • , 2014.

[46] Mohammadhadi Mohandes, Behnam Moradi, Kamal Gupta, and Mehran Mehrandezh. Robot to human object handover using vision and joint torque sensor modalities. In *Robot Intelligence Technology and Applications 7: Results from the 10th International Conference on Robot Intelligence Technology and Applications*, pages 109–124. Springer, 2023.

[47] AJung Moon, Daniel M Troniak, Brian Gleeson, Matthew KXJ Pan, Minhua Zheng, Benjamin A Blumer, Karon MacLean, and Elizabeth A Croft. Meet me where i'm gazing: how shared attention gaze affects human-robot handover timing. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pages 334–341, 2014.

[48] Bilge Mutlu. *Designing gaze behavior for humanlike robots.* PhD thesis, Carnegie Mellon University, 2009.

[49] Open-CV. Depth Map from Stereo Images. `https://docs.opencv.org/3.4/dd/d53/tutorial_py_depthmap.html`.

[50] Valerio Ortenzi, Akansel Cosgun, Tommaso Pardi, Wesley P Chan, Elizabeth Croft, and Dana Kulić. Object handovers: a review for robotics. *IEEE Transactions on Robotics*, 37(6):1855–1873, 2021.

[51] Matthew KXJ Pan, Vidar Skjervøy, Wesley P Chan, Masayuki Inaba, and Elizabeth A Croft. Automated detection of handovers using kinematic features. *The International Journal of Robotics Research*, 36(5-7):721–738, 2017.

[52] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[53] Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

[54] Sina Parastegari, Ehsan Noohi, Bahareh Abbasi, and Miloš Žefran. A fail-safe object handover controller. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2003–2008. IEEE, 2016.

[55] Sina Parastegari, Ehsan Noohi, Bahareh Abbasi, and Miloš Žefran. Failure recovery in robot–human object handover. *IEEE Transactions on Robotics*, 34(3):660–673, 2018.

[56] Vinay Pilania and Kamal Gupta. Mobile manipulator planning under uncertainty in unknown environments. *The International Journal of Robotics Research*, 37(2-3):316–339, 2018.

[57] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[58] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[59] Juan José Rodríguez and Carlos J Alonso. Support vector machines of interval-based features for time series classification. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 244–257. Springer, 2004.

[60] ROSwiki. rosbag. `http://wiki.ros.org/rosbag`, 2020.

[61] ROSwiki. Read And Write Pascal Voc XML Annotations In Python. `https://mlhive.com/2022/02/read-and-write-pascal-voc-xml-annotations-in-python#:~:text=Pascal%20VOC(Visual%20Object%20Classes,and%20train%20Machine%20Learning%20models.`, 2022.

[62] ROSwiki. sensormsgs JointState Message. `http://docs.ros.org/en/noetic/api/sensor_msgs/html/index-msg.html`, 2022.

[63] ROSwiki. ApproximateTime Policy. `https://wiki.ros.org/message_filters#ApproximateTime_Policy`, 2023.

[64] ROSwiki. ROS Finite State Machine. `http://wiki.ros.org/decision_making/Tutorials/FSM`, 2023.

[65] Schunk. SCHUNK Dextrous Hand 2.0 (SDH 2.0). `https://www.nist.gov/document/9020173r1pdf`, 2008. Online; National Institute of Standards and Technology.

[66] Satoru Shibata, Kanya Tanaka, and Akira Shimizu. Experimental analysis of handing over. In *Proceedings 4th IEEE international workshop on robot and human communication*, pages 53–58. IEEE, 1995.

[67] Monica Sileo, Michelangelo Nigro, Domenico D Bloisi, and Francesco Pierri. Vision based robot-to-robot object handover. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 664–669. IEEE, 2021.

[68] Maria Staudte and Matthew Crocker. The utility of gaze in spoken human-robot interaction. In *Proceedings of Workshop on Metrics for Human-Robot Interaction 2008, March 12th*, pages 53–59, 2008.

[69] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[70] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[71] Christian Szegedy, Scott Reed, Dumitru Erhan, Dragomir Anguelov, and Sergey Ioffe. Scalable, high-quality object detection. *arXiv preprint arXiv:1412.1441*, 2014.

[72] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

[73] Weitian Wang, Rui Li, Zachary Max Diekel, Yi Chen, Zhujun Zhang, and Yunyi Jia. Controlling object hand-over in human–robot collaboration via natural wearable sensing. *IEEE Transactions on Human-Machine Systems*, 49(1):59–71, 2018.

[74] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[75] Jorg D Wichard and Maciej Ogorzalek. Time series prediction with ensemble models. In *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)*, volume 2, pages 1625–1630. IEEE, 2004.

[76] Wei Yang, Chris Paxton, Maya Cakmak, and Dieter Fox. Human grasp classification for reactive human-to-robot handovers. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11123–11130. IEEE, 2020.

[77] Yezhou Yang, Cornelia Fermuller, Yi Li, and Yiannis Aloimonos. Grasp type revisited: A modern perspective on a classical feature for vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 400–408, 2015.

[78] Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956, 2009.

[79] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International journal of machine learning and cybernetics*, 1:43–52, 2010.

Appendix A

Specification

A.1 RGB-D Registeration

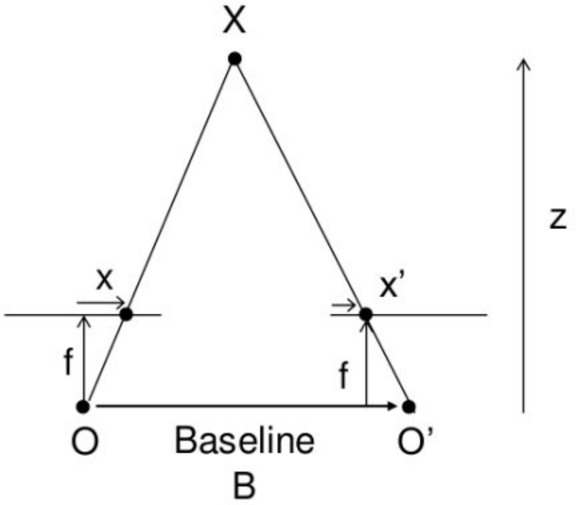


Figure A.1: A sample figure of two stereo cameras and an object with its images [49]

The above diagram contains equivalent triangles. Writing their equivalent equations will yield us following result:

$disparity = x - x' = Bf/Z$ [49]

The coordinates x and x denote the angular separation between the camera's optical center and the 3D scene point in the picture plane. We have the focal length of the camera (f) and the distance between the cameras (B). The above equation states, in a nutshell, that the distance between two points in an image and the camera's center is directly proportional to the depth of that point in the picture. Therefore, we can calculate the depth of every pixel in a picture with this data [49].

A.2 Jacobian

All the equations in this section have been adopted from [14].

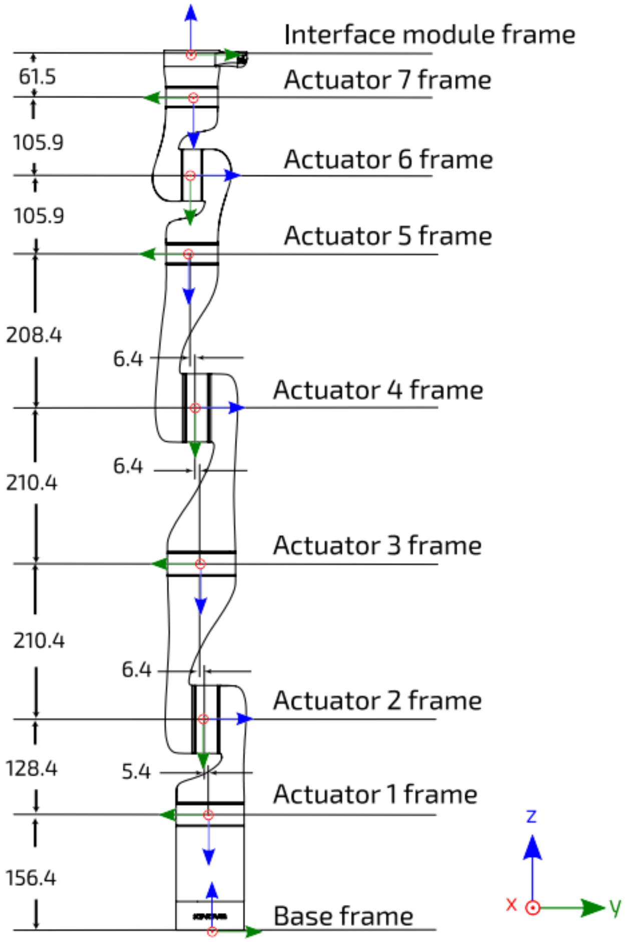


Figure A.2: 7 DoF robot frame definitions and dimensions (all joints at 0 position, dimensions in mm) [31].

$$^{Base}T^{*}{}_{Tool} = {}^{Base}T^{*}{}_{1}{}^{1}T^{*}{}_{2}{}^{2}T^{*}{}_{3}{}^{3}T^{*}{}_{4}{}^{4}T^{*}{}_{5}{}^{5}T^{*}{}_{6}{}^{6}T^{*}{}_{7}{}^{7}T^{*}{}_{Tool}$$

Here:

$$^{i-1}T^{*}{}_{i} = {}^{i-1}T_{i}R_{z}(q_{i})$$

Where $^{i-1}T^{*}{}_{i}$ is the general transformation matrix from frame [i-1] to frame [i], and when the rotation angle between 2 frames is 0, the transformation matrix will be $^{i-1}T_{i}$. $R_z(q_i)$ represents the rotation of q_i around joint i [14].

To Start with calculation of static forces in a manipulator, we pay attention to the typical feature of a manipulator. Typically, the robot is going under some external forces and loads at the end-effoctor that could be the result of keeping a load at the gripper or moving something in the environment with the end-effector. To keep the system in static equilibrium, we must find a solution for the joint torques generated as a result of the external force. We first lock all the joints to turn the manipulator into a structure before we can consider static forces in a manipulator. Then, taking into account every link in the structure, we formulate a force-moment balance relationship using the link frames. Finally, we determine the static torque that the manipulator needs to be in static equilibrium with respect to the joint axis. Then the genrerated set of joint torques required to support a static load acting at the end-effector will be solved through the proposed equation [14].

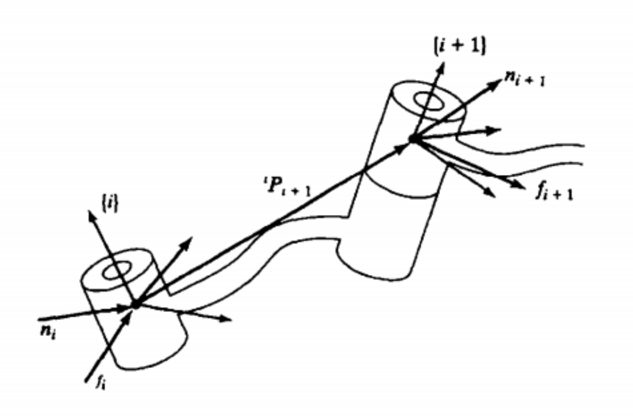


Figure A.3: Static force balance for a single link [14]

Fig A.3 shows the static forces and torques acting on link i. Summing the forces and torques and writing the equilibrium formula we have:

$$^{i}f_i - {^{i}f_{i+1}} = 0$$

$$^{i}n_i - {^{i}n_{i+1}} - {^{i}P_{i+1}} * {^{i}f_{i+1}} = 0$$

If we decide to calculate these forces and torques based on each link's own frame, then we add the usage of rotation matrix which describes frame i+1 to frame i. This will result in the following:

$$^{i}f_i = {^{i}_{i+1}R} * {^{i+1}f_{i+1}}$$

$$^{i}n_i = {^{i}_{i+1}R} * {^{i+1}n_{i+1}} - {^{i}P_{i+1}} * {^{i}f_i}$$

In this part, the forces and torques caused due to gravity will be ignored since that takes a great complexity into account for calculation. The important question here is that in which direction should the torques be calculated. As already known, all the torque components will be resisted by the structure of the whole manipulator (machanism) except the torque with the axis parallel to the axis of the joint's rotation. That's why the dot product of the joint-axis vector and the moment around that link is calculated:

$$\tau_i = {}^{i}_{i}n^T . {}^{i}\hat{Z}_i$$

Moving from end-effector backwards to the base of the manipulator, we calculate the torques generated by the external force on the robot. The resulting matrix multiplication shows that the relation between torque array values and the force components is made with the transpose of the Jacobian matrix written with respect to the end-effector frame. We also can extract this from the following explanation. When force is applied to a mass, i.e. a mechanism, it goes through a displacement, which we basically refer to as work being done. Work is a scalar quantity that has units of energy and is defined as a force acting across a distance. By permitting the quantity of this displacement to go to an infinitesimal, the principle of virtual work enables us to make certain claims about the static case. Since work is measured in energy units, it must have the same value regardless of the generalised coordinate system. We can compare the work done in joint-space terms to the effort done in Cartesian terms specifically. So basically, work is defined as the dot product of a vector force or torque and a vector displacement (could be angular or Cartesian) in the multidimensional situation. Thus we have:

$$F.\delta x = \tau.\delta\theta$$

This expression can be altered as below:

$$F^T.\delta x = \tau^T.\delta\theta$$

and by knowing that jacobian is defined as:

$$\delta X = J\delta\theta$$

we can write:

$$F^T J\delta\theta \ = \tau^T \delta\theta$$

so we can conclude that:

$$F^T J = \tau^T$$

and

$$\tau = J^T F$$

please note that the reference frame for calculating the jacobian is the same frame used for measuring the effective forces acting on the end-effector.

where acting F on the end-effector and τ are respectively a 6×1 Cartesian force-moment vector and a 6×1 vector of joint torques, and δx and $\delta\theta$ are a 6×1 small Cartesian displacement of the end-effector and a 6×1 small joint movements [14].

A.3 Transformation Matrices

Transformation matrices in Kinova gen3 arm [31].

Transformation	${}^{i-1}T_i$	${}^{i-1}{T^*}_i$
Base to frame 1	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0.1564 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q1) & -sin(q1) & 0 & 0 \\ -sin(q1) & -cos(q1) & 0 & 0 \\ 0 & 0 & -1 & 0.1564 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 1 to frame 2	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0.0054 \\ 0 & 1 & 0 & -0.1284 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q2) & -sin(q2) & 0 & 0 \\ 0 & 0 & -1 & 0.0054 \\ sin(q2) & cos(q2) & 0 & -0.1284 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 2 to frame 3	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -0.2104 \\ 0 & -1 & 0 & -0.0064 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q3) & -sin(q3) & 0 & 0 \\ 0 & 0 & 1 & -0.2104 \\ -sin(q3) & -cos(q3) & 0 & -0.0064 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 3 to frame 4	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -0.0064 \\ 0 & 1 & 0 & -0.2104 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q4) & -sin(q4) & 0 & 0 \\ 0 & 0 & -1 & -0.0064 \\ sin(q4) & cos(q4) & 0 & -0.2104 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 4 to frame 5	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -0.2084 \\ 0 & -1 & 0 & -0.0064 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q5) & -sin(q5) & 0 & 0 \\ 0 & 0 & 1 & -0.2084 \\ -sin(q5) & -cos(q5) & 0 & -0.0064 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 5 to frame 6	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & -0.1059 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q6) & -sin(q6) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sin(q6) & cos(q6) & 0 & -0.1059 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 6 to frame 7	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -0.1059 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} cos(q7) & -sin(q7) & 0 & 0 \\ 0 & 0 & 1 & -0.1059 \\ -sin(q7) & -cos(q7) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
frame 7 to Tool	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -0.0615 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -0.0615 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Appendix B

Documents

B.1 Ethics Approval

SFU RESEARCH ETHICS

Minimal Risk Approval – Delegated

Study Number: 30000988
Study Title: Robot to Human Object Handover using Vision and Joint Torque Sensor Modalities

Approval Date: April 29, 2022
Principal Investigator: Kamal Gupta
Faculty/Department: Engineering Science

Expiration Date: April 29, 2023
SFU Position: Faculty

Student Lead: Mohammadhadi Mohandes
SFU Collaborator(s): Mehran Mehrandezh
Research Personnel: N/A
External Collaborator(s): N/A

Funder: NSERC
Funding Title: Mobile Manipulation
Funding Number: 611245

Document(s) Approved in this Application:

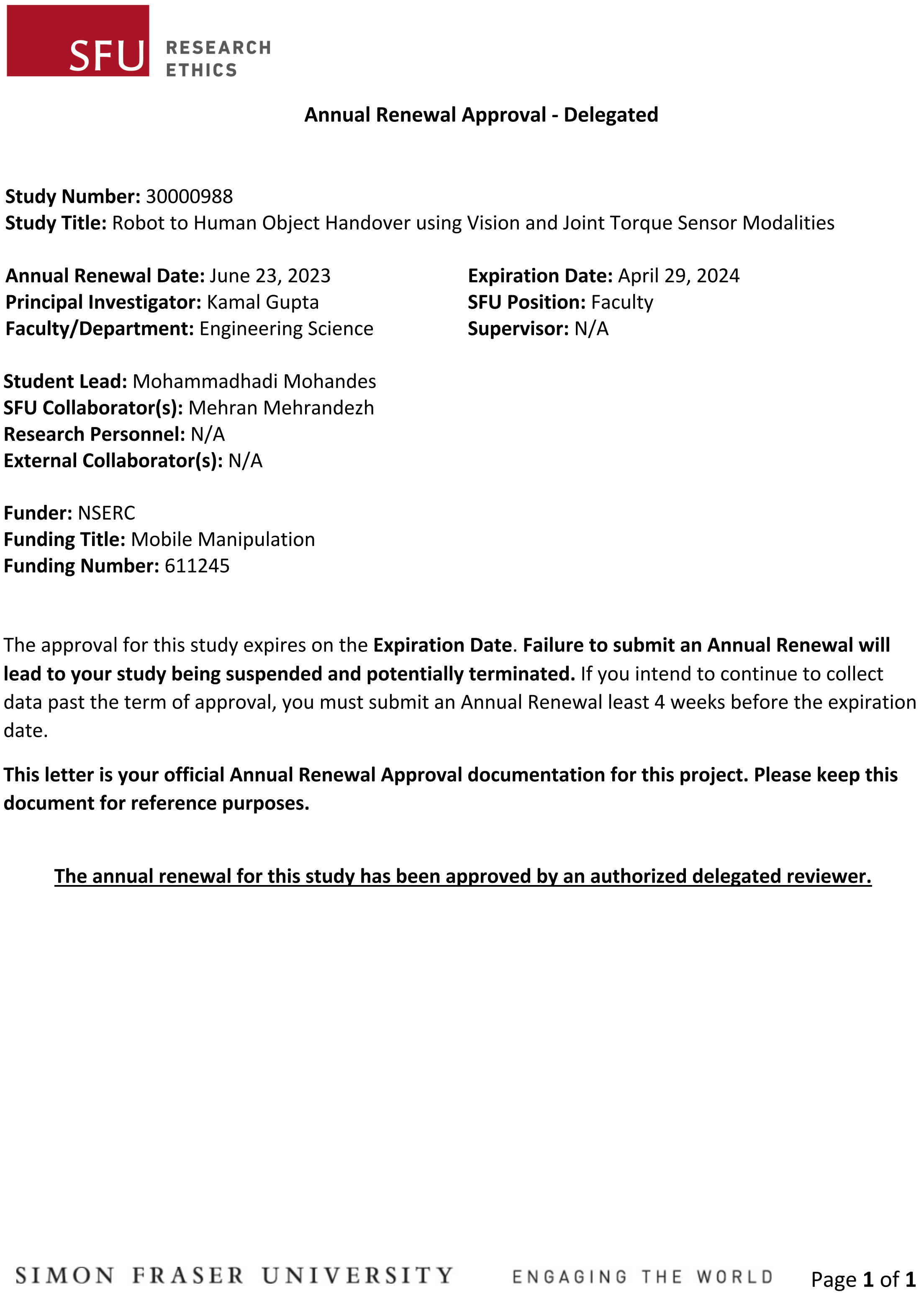
- TCPS 2 CORE Tutorial Certificate for Mohammadhadi Mohandes dated April 25, 2022
- Consent Form, version 1 dated April 26, 2022
- Recruitment Email, version 1 dated April 26, 2022
- Survey Questions, version 1 dated April 26, 2022
- Conduct Research, version 1 dated April 26, 2022
- Confirmation Email, version 1 dated April 26, 2022
- Recruitment Script, version 1 dated April 26, 2022

The application for ethical review and the document(s) listed above have been reviewed and the procedures were found to be acceptable on ethical grounds for research involving human participants.

The approval for this Study expires on the **Expiration Date. An Annual Renewal must be completed every year prior to the Expiration Date. Failure to submit an Annual Renewal will lead to your study being suspended and potentially terminated.** The Board reviews and may amend decisions or subsequent amendments made independently by the authorized delegated reviewer at its regular monthly meeting.

This letter is your official ethics approval documentation for this project. Please keep this document for reference purposes.

This study has been approved by an authorized delegated reviewer.



SFU RESEARCH ETHICS

Annual Renewal Approval - Delegated

Study Number: 30000988
Study Title: Robot to Human Object Handover using Vision and Joint Torque Sensor Modalities

Annual Renewal Date: June 23, 2023	**Expiration Date:** April 29, 2024
Principal Investigator: Kamal Gupta	**SFU Position:** Faculty
Faculty/Department: Engineering Science	**Supervisor:** N/A

Student Lead: Mohammadhadi Mohandes
SFU Collaborator(s): Mehran Mehrandezh
Research Personnel: N/A
External Collaborator(s): N/A

Funder: NSERC
Funding Title: Mobile Manipulation
Funding Number: 611245

The approval for this study expires on the **Expiration Date**. **Failure to submit an Annual Renewal will lead to your study being suspended and potentially terminated.** If you intend to continue to collect data past the term of approval, you must submit an Annual Renewal least 4 weeks before the expiration date.

This letter is your official Annual Renewal Approval documentation for this project. Please keep this document for reference purposes.

The annual renewal for this study has been approved by an authorized delegated reviewer.

SIMON FRASER UNIVERSITY ENGAGING THE WORLD Page **1** of **1**

B.2 Survey Questions

SFU SIMON FRASER UNIVERSITY

Survey Questions

Robot to Human Object Handover – In-Person Human/Robot Interaction and Survey

Hello,

Thank you for dedicating your time to help with this study

Evaluate each feature in the resulting handover as compared to human-to-human handover.

Questions:

- Rate the likelihood that the object could have been dropped during the handover

(1 – Likely to be dropped, 5- Not likely to be dropped):

****

- Rate the resistance you felt in the handover

(1 – Too resistant, 5 – Very compliant):

****

- Rate the smoothness in the handover

(1- not smooth at all, 5- very smooth):

****

- Rate the overall quality of the handover

(1- very different, 5- very similar):

****

B.3 Consent Form

SFU SIMON FRASER UNIVERSITY

Consent Form

Robot to Human Object Handover – In-Person Human/Robot Interaction and Survey

You are being invited to take part in this research study because you are able to see and grasp an object held in a robot's hand and apply actions with a steady hand.

We are conducting this study to learn more about the robustness and success rate of our handover robotic platform and to explore how far the system is from a natural human-human handover.

If you agree to participate the activities should take about 20 minutes to complete.

STUDY TEAM

Mohammadhadi Mohandes, Student Lead

Kamal Gupta, Principal Investigator, Professor, Faculty of Applied Science, Engineering Science Department

All members of the research team are fully vaccinated. The research team will abide by the latest provincial health guidelines in relation to the COVID19 pandemic.

VOLUNTARY PARTICIPATION

Your participation is voluntary. Your instructor/supervisor will not know if you do or don't participate in this study and your decision on whether to participate in the study will not influence your grades in any coursework and will not affect your relationship with your teacher/supervisor in any way. If you decide to not to participate, you may withdraw from the study by informing the Student Lead via email or verbally. Please note that once you submit you will not be able to withdraw. This is because the study is anonymous and the researchers will not be able to know which responses are yours.

You are free to skip any questions you do not wish to answer.

STUDY PROCEDURES

During this study, participants will stand in front of the robot and perform a pre-defined action on the object. The actions you may be asked to perform include grasping, pulling, and pushing the object. Our algorithm will process data from two types of sensors which are torque sensors and a vision sensor on the robot. The torque sensor data will change in response to the actions performed by the human subject and the vision data only captures the object and the fingers of the subject. This primary data is processed real-time to give numerical outputs and will not be stored anywhere. After each action, the robot will decide to release or keep the object and you will be asked to answer a few questions about the process like the smoothness and robustness of the handover. Answers will be anonymous and not associated with your name or any personal identifiers. This study is toward the completion of M.Sc. degree.

If you agree to it, your interactions with the robot may be video recorded. This video will only capture your torso from shoulder to hip so that your arm can be seen. Video file names will be coded and not contain any personal identifiers.

POTENTIAL RISKS AND BENEFITS OF THE STUDY

There are no foreseeable risks or benefits to you in participating in this study. However, in the future, others may benefit from what we learn in this study.

CONFIDENTIALITY AND DATA SECURITY

The results of the study will be anonymous, this means that no one will know who you are from your data. The data from the survey and video files will be stored on SFU managed servers for 4-6 months. Primary data received through the robot's sensors is processed real time and not stored.

CONCERNS ABOUT THIS STUDY

If you have any concerns about your rights as a research participant and/or your experiences while participating in this study, please contact the SFU Office of Research Ethics at dore@sfu.ca or 778-782-6618.

Taking part in this study is entirely up to you. You have the right to refuse to participate in this study. By checking 'I AGREE' below and initialing beside it, you indicate that you consent to participate in this study. You may separately choose to agree to having your interaction video recorded by checking the box and initialing beside it. You do not waive any of your legal rights by participating in this study.

□ I agree to participate in the robot/human interaction and survey
□ I agree to have my robot/human interaction video recorded

B.4 Recruitment Email

SFU SIMON FRASER UNIVERSITY

Recruitment Email

Robot to Human Object Handover – In-Person Human/Robot Interaction and Survey

Greetings,

The Ramp Lab in the Department of Engineering Science at Simon Fraser University is conducting a study on “Robot to Human Object Handover” which is toward the completion of M.Sc. degree. We are seeking SFU graduate/undergraduate students to participate in this study by completing a short series of interactions with the robot and answering a few survey questions. Data collected during this study will be anonymous.

You are eligible to participate in this study if:

1. You are able to clearly see an object 1 metre away from you
2. You are physically able to reach your arm out and grasp a light (50-100 grams) water bottle sized object with a steady hand
3. You are fully vaccinated for COVID19

You are not eligible for this study if:

1. You are with disabilities such as blindness, tremor disease, or the inability to grasp an object

If you decide to participate in this study, you will be prompted to attend our robotic lab and interact with our robotic handover system (10 min) and complete a short questionnaire (~5-10 mins). You may agree to have your interaction video recorded in which case only your torso (shoulder to hip) will be captured.
Please note that your instructor/supervisor will not know if you do or don’t participate in this study and your decision on whether to participate in the study will not influence your grades in any coursework and will not affect your relationship with your teacher/supervisor in any way.

If you are willing to participate in this study, please reply to this email and write “confirmed”. Then we will inform you about the times that we are in the lab to do the experiment in a confirmation email.

If you have any inquiries concerning the study, please feel free to contact the student lead, Mohammadhadi Mohandes

Thank you for your time and support.

Sincerely,

Dr. Kamal Gupta, Principal Investigator (faculty supervisor)
Mohammadhadi Mohandes, Student lead (M.Sc. student)
Engineering Science Department, Simon Fraser University

B.5 Recruitment Script Social Media

SFU SIMON FRASER UNIVERSITY

Recruitment Script

Robot to Human Object Handover – In-Person Human/Robot Interaction and Survey

Greetings,

The Ramp Lab in the Department of Engineering Science at Simon Fraser University is conducting a study on “Robot to Human Object Handover” which is toward the completion of M.Sc. degree. We are seeking SFU graduate/undergraduate students to participate in this study by completing a short series of interactions with the robot and answering a few survey questions. Data collected during this study will be anonymous.

You are eligible to participate in this study if:

1. You are able to clearly see an object 1 metre away from you
2. You are physically able to reach your arm out and grasp a light (50-100 grams) water bottle sized object with a steady hand
3. You are fully vaccinated for COVID19

You are not eligible for this study if:

1. You are with disabilities such as blindness, tremor disease, or the inability to grasp an object

If you decide to participate in this study, you will be prompted to attend our robotic lab and interact with our robotic handover system (10 min) and complete a short questionnaire (~5-10 mins). You may agree to have your interaction video recorded in which case only your torso (shoulder to hip) will be captured.
Please note that your instructor/supervisor will not know if you do or don’t participate in this study and your decision on whether to participate in the study will not influence your grades in any coursework and will not affect your relationship with your teacher/supervisor in any way.

If you are willing to participate in this study, please go to the main sender of this message and write “confirmed” privately. To protect the identity of out participants, we prefer that you don’t react to this message publicly. Then we will inform you about the times that we are in the lab to do the experiment in a confirmation message.

If you have any inquiries concerning the study, please feel free to contact the student lead, Mohammadhadi Mohandes and in Telegram application using @hadimohandes
ID.

Thank you for your time and support.

Sincerely,

Dr. Kamal Gupta, Principal Investigator (faculty supervisor)
Mohammadhadi Mohandes, Student lead (M.Sc. student)
Engineering Science Department, Simon Fraser University

B.6 Conduct Research

SFU SIMON FRAS
UNIVERSITY

Conduct Research

Robot to Human Object Handover – In-Person Human/Robot Interaction and Survey

Welcome to the ramp lab and we want to thank you for your time and dedication. I assume you have read and signed the consent form and you have emailed us back your confirmation about the attendance in this experiment which is “Robot to Human Object Handover”. Please note that the questionnaire is anonymous, and no one will know about your participation in this experiment ad it does not have any effect on your grades or other things.
There will be one researcher and a human subject conducting the experiment in the ramp lab. Both the researcher and the human subject need to be wearing face masks.
The researcher will give instructions about the actions that the subject needs to apply on the object
The researcher will prompt the human subject to stand in front of the robot in the marked area. The researcher then will give instructions on when the subject needs to apply the desired action on the object. After each action the questionnaire which in an open doc file on a laptop needs to be filled out and if the participant needs any help, the researcher will kindly answer their question. Then at the end of the whole experiment (about 20 min long) the researcher will thank the human subject for their time dedication and their participation and they will leave the lab.

B.7 Confirmation Email

Confirmation Email

Robot to Human Object Handover – In-Person Human/Robot Interaction and Survey

Dear …,

Thanks for accepting our invitation to participate in Robot to human Object Handover in the Ramp Lab in the Department of Engineering Science at Simon Fraser University. We will be in conducting research in the lab every day (Mon-Fri) from 9 am to 6 pm. So let us know what time suits you via replying to this email and we will see you in the lab. Also the consent form is attached to this email and you can read it before attending the lab. When you arrive in the lab, you will be provided with the electronic format of the consent form on a laptop and you can decide whether or not you want to take part in the study and survey. Thanks

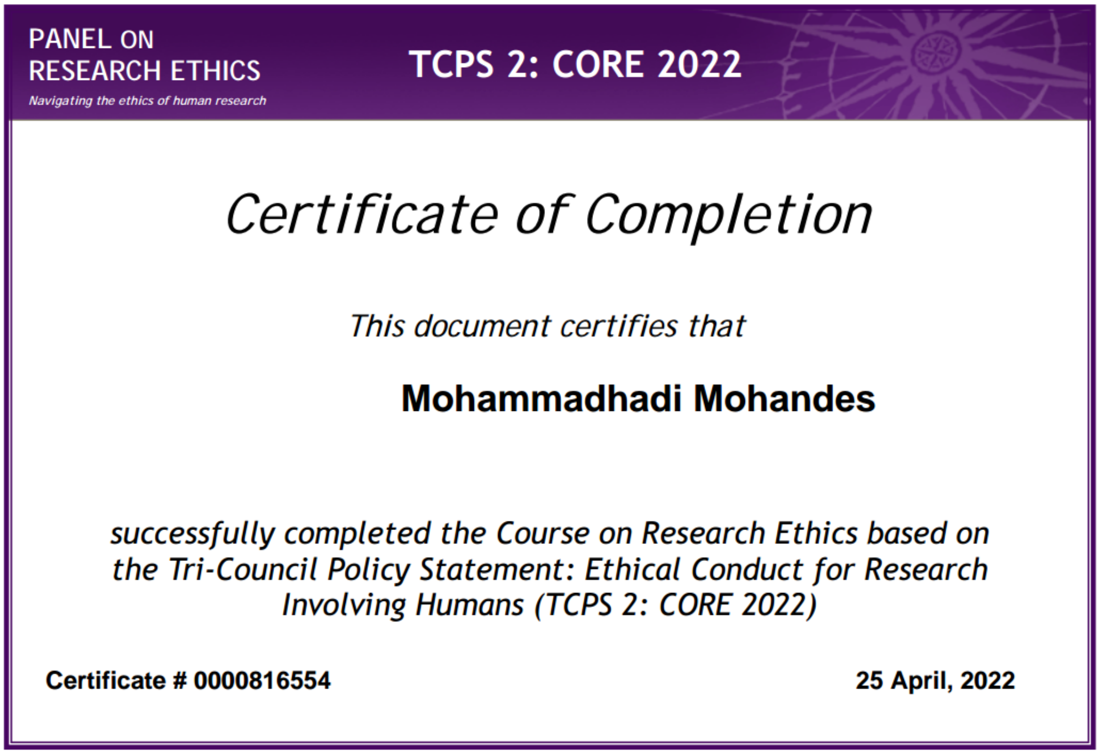
If you have any inquiries concerning the study, please feel free to contact the student lead, Mohammadhadi Mohandes

Thank you for your time and support.

Sincerely,

Dr. Kamal Gupta, Principal Investigator (faculty supervisor)
Mohammadhadi Mohandes, Student lead (M.Sc. student)
Engineering Science Department, Simon Fraser University

B.8 Research Ethics Certificate



PANEL ON
RESEARCH ETHICS
Navigating the ethics of human research

TCPS 2: CORE 2022

Certificate of Completion

This document certifies that

Mohammadhadi Mohandes

successfully completed the Course on Research Ethics based on the Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans (TCPS 2: CORE 2022)

Certificate # 0000816554

25 April, 2022

Figure B.1: Research Ethics certification

Appendix C

Video

This video was presented at RITA 2023 [46]. We have added multiple images with explana-tions along the video for more clarification. See the accompanying supplemental files:

Video for experiments

Filename:
RITA 2022_video.mp4

Finger detection results

Filename:
finger_detection_results.mp4