ROBUST AND ADAPTIVE DEXTEROUS MANIPULATION WITH VISION-BASED LEARNING FROM MULTI-DEMONSTRATIONS

by

Nuo Chen

Submitted in partial fulfillment of the requirements
for the degree of Master of Applied Science

at

Dalhousie University
Halifax, Nova Scotia
August 2024

Table of Contents

List of Tables

List of Figures

Abstract

Robotic manipulators perform tasks as humans with the potential to greatly assist people in industry, health care, and general society services. In this thesis, a vision-based learn from demonstration framework for a 7-degree-of-freedom robotic manipulator has been proposed. This framework enables learning from multiple human hand demonstrations to execute dexterous pick-and-place tasks. Conventional methods for collecting demonstration data involve manually and physically moving the robot. These methods can be cumbersome, lack dexterity, and be physically straining. The proposed contactless and markerless approach leverages MediaPipe software, dynamic time warping, Gaussian mixture model, and Gaussian mixture regression to capture and regress multiple dexterous hand motions. The proposed approach results in a more comprehensive motion representation, simplifying multiple demonstrations, and mitigating the non-smoothness inherent in a single demonstration. Dynamic movement primitives (DMP) with a force coupling term are employed to adaptively assimilate human actions into trajectories executable in dynamic environments. By considering the estimated variance from demonstration data, the path planning parameters are automatically fine-tuned and associated with the linear and nonlinear terms to adapt the trajectories. To compensate for unknown external disturbances, a non-singular terminal sliding mode controller (NTSMC) is applied to the Franka Emika robot for precise trajectory tracking. Experimental studies demonstrated the effectiveness and robustness of the proposed framework in executing human hand demonstrations, motion planning, and control for pick-and-place tasks.

List of Abbreviations and Symbols Used

Abbreviations

LfD	learn from demonstration
DTW	dynamic time warping
GMM	Gaussian mixture model
GMR	Gaussian mixture regression
DMP	dynamic movement primitives
NTSM	non-singular terminal sliding mode
2D	two-dimensional
3D	three-dimensional
HRI	human-robot interaction
PD	proportional-derivative
SMC	sliding mode control
RL	reinforcement learning
DoF	degree of freedom
DH	Denavit-Hartenberg
RGB	red-green-blue
FPS	frames per second
ROS	robot operating system
PDF	probability density function
EM	expectation maximization
SM	sliding mode
RMSE	root mean square error

Symbols

$\forall$	for all
$\in$	belongs to

$\mathbb{R}$	set of real numbers
$\mathbb{R}^n$	set of $n \times 1$ real vectors
$\mathbb{R}^{7\times 7}$	set of $n \times n$ real matrices
A^{-1}	inverse of matrix A
A^T	transpose of matrix A
$\mathcal{I}$	identity matrix
Σ	summation
$\mathcal{P}$	probability
$\mathcal{N}$	probability density function

Acknowledgements

First of all, I would like to appreciate my supervisor, Dr. Ya-Jun Pan, for her attentive guidance and support in all aspects. It is my great honor to be able to conduct research under her guidance. I would like to thank Dr. Jason Gu and Dr. Mohammad Saeedi for their assistance as my supervisory committee members.

Secondly, I would also like to thank my friends in Halifax along with my colleagues in the ACM Lab. Thank you for making my work and life more interesting and helping me a lot. In particular, I would like to thank Lucas Wan, Qiguang Chen, and Scott Buchanan for their help with the experiments.

Finally, I would like to express my gratitude to my family for funding my studies and providing me with emotional support.

Chapter 1

Introduction

This chapter describes the importance of manipulators for practical applications in broad areas such as manufacturing, agriculture, and healthcare. An overview of the motivation, contribution, and outline of this thesis is presented in this chapter.

1.1 Background

As the robotics development continues to expand, so will the range of applications for robots in everyday life. This section introduces some industrial applications of manipulators in manufacturing, agriculture, and healthcare.

1.1.1 Manufacturing

In modern manufacturing, robotic manipulator systems have become an integral part of automated production. The introduction of robotic arms has not only increased productivity but also significantly improved product quality and consistency.

Manipulators can perform a variety of complex assembly tasks on the assembly line. Traditional manual assembly is not only time-consuming but also prone to human error. Robotic arms are programmed to accurately take operations such as gripping, aligning, inserting, and fixing parts. Especially in electronics manufacturing, robots can quickly and accurately assemble tiny and precise components such as circuit boards and micro motors. This not only improves the production speed but also ensures the quality consistency of each product.

Welding robots are widely used in the automotive manufacturing and metal processing industries. Their main advantage lies in their ability to perform high-precision welding operations such as spot and arc welding. Equipped with advanced sensors and vision systems, welding robots can monitor the welding process in real-time and adjust the welding parameters to ensure the quality of the weld. Compared with manual welding, welding robots can work in hazardous environments such as high temperatures and toxic gases, which reduces health risks for human workers.

Manipulators are also applied in warehousing and logistics for handling boxes and bags. Traditional manual handling leads to worker fatigue and injury, while robotic arms can work 24/7, significantly improving handling efficiency. The robotic arm equipped with an intelligent gripping system can automatically recognize the shape and weight of the items and adjust the gripping strength to ensure that the items are not damaged during the handling process.

In the packaging industry, manipulators are used to automate packaging, sealing, labeling, and other operations. Its high-speed and precise operation capability significantly improves packaging speed and reduces labor costs. In the food and pharmaceutical industries, robotic arms are also able to work in aseptic environments to ensure product hygiene and safety.

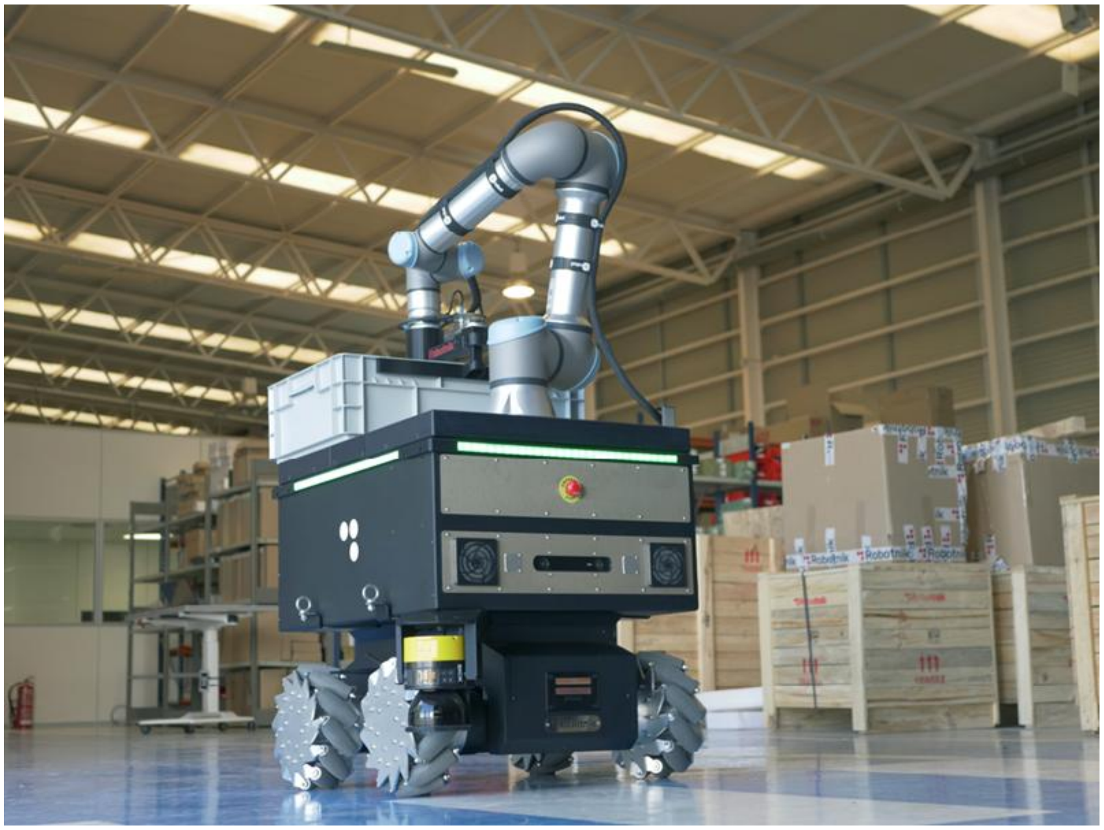


Figure 1.1: A mobile manipulator used for transportation [1]

The role of manipulator systems in smart manufacturing and Industry 4.0 is particularly important. By combining with the Internet of Things and big data analytics, robots can achieve self-monitoring and self-optimization. This intelligent production method greatly improves productivity and equipment utilization. Fig. 1.1 shows an example of a mobile manipulator used for transportation.

1.1.2 Agriculture

In the field of agriculture, the application of manipulator systems is becoming more and more widespread, significantly improving the efficiency and precision of agricultural production. The main applications of robotic arms in agriculture include picking, planting, spraying, sorting, packaging, and testing of agricultural products.

Manipulators play an important role in fruit and vegetable picking. Conventional manual picking is not only time-consuming and laborious but also prone to damage the fruits. Robotic arms can accurately recognize and pick fruits and vegetables by equipping high-precision vision systems and flexible gripping devices, reducing fruit damage and waste. As shown in Fig. 1.2, the robotic system with a gripper can pick apples and adjust the gripping force and angle to ensure that the fruits are picked intact.

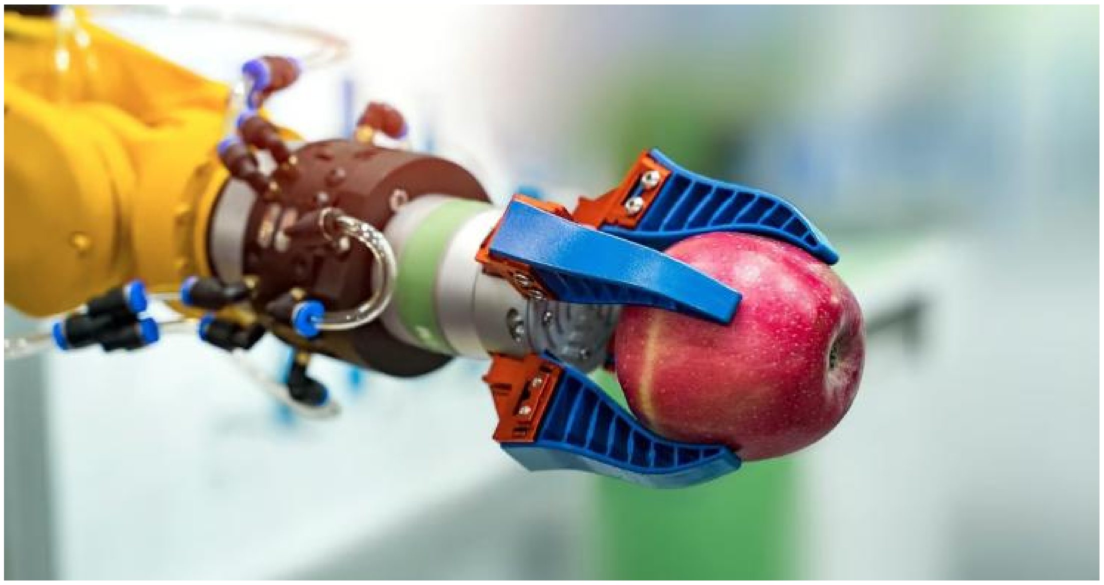


Figure 1.2: Agricultural robots pick apples, gather strawberries, harvest lettuce and strip away weeds [2].

Manipulator systems can be used for precision sowing and transplanting in planting operations. It is easy for the robot to accurately plant seeds or seedlings in a given location, ensuring uniformity of planting and consistency of depth. For example, in the greenhouse, the robot can accurately insert the seedlings into the soil according to the predetermined planting diagram and adjust the depth and spacing at the right time, which improves the planting efficiency and survival rate.

Robotic systems are widely employed in pesticide spraying and fertilizer application. Equipped with intelligent sensors and spraying systems, the robotic arm can accurately control the spraying amount and range of pesticides and fertilizers, avoiding overspray and waste. For example, the manipulator systems can accurately spray pesticides and fertilizers according to the growth status of plants and pests and diseases, which improves the effect of application and resource utilization, and at the same time reduces environmental pollution.

1.1.3 Healthcare

The applications of manipulator systems in the medical field are wide and varied, from surgery to rehabilitation, to laboratory automation and drug development, robotic arms play an important role in improving the quality of healthcare services, therapeutic effects, and reducing medical costs.

Manipulators are most widely and maturely used in surgery, especially in minimally invasive surgery and complex surgery. The robotic system performs precise cuts, sutures, and other operations through fine manipulation by the surgeon at the console. The high precision and stability of the manipulators greatly lower the risk of post-operative infection and shorten patient recovery time. As illustrated in Fig. 1.3, the surgeon demonstrates the surgical operation, and the manipulator will eliminate the shaking of the human hands.

In cardiovascular intervention, neurological intervention, and other minimally invasive interventions, the robotic arm can help the doctor carry out fine operations, such as catheter navigation and stent placement. The manipulator can provide real-time feedback on the position of the catheter and the situation of the blood vessels, assisting the doctor in carrying out accurate operations, reducing operational errors, and improving the success rate of the operation.

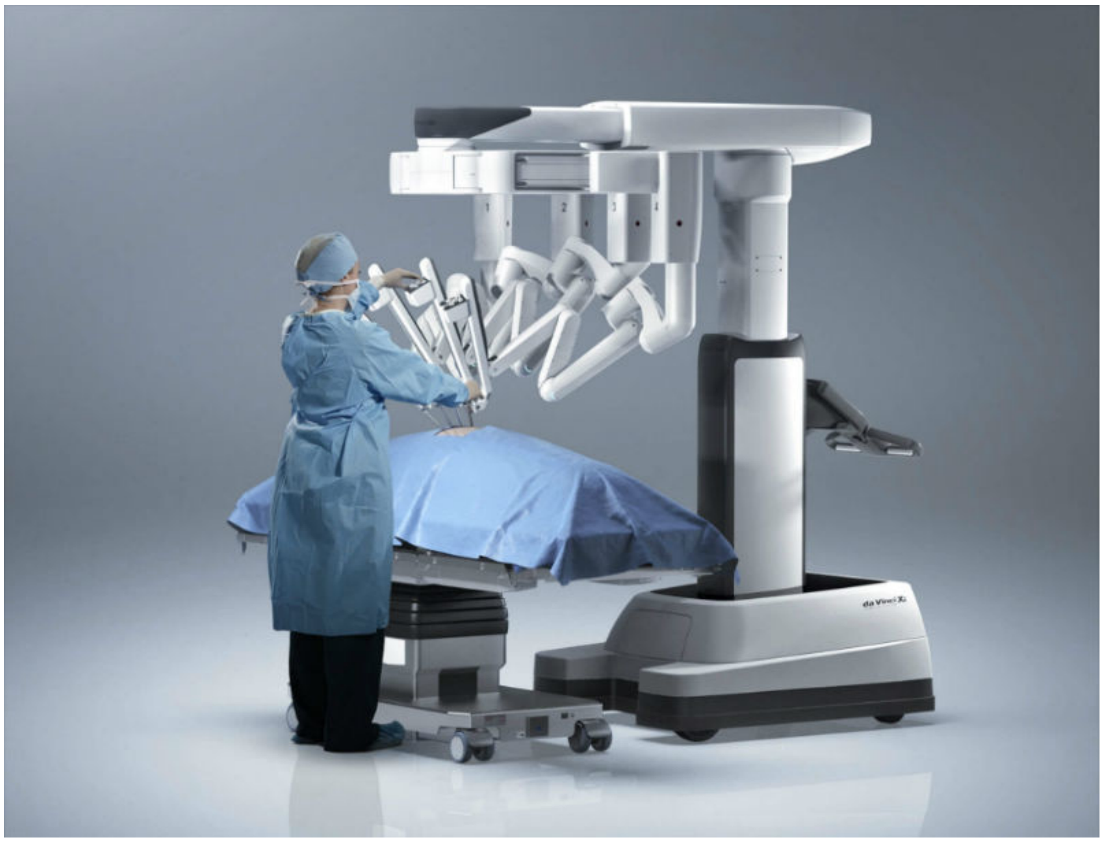


Figure 1.3: The surgeon is demonstrating the surgical operation of a specific task with manipulators [3]

Robotic arms are also used in rehabilitation therapy to help patients with physical rehabilitation training. Rehabilitation robots can provide personalized training programs according to the patient's rehabilitation needs to help patients restore motor function. For instance, the robotic arm can simulate actions in daily life, such as grasping and holding, to help stroke patients with hand rehabilitation training. Through sensors and intelligent control systems, the robotic arm can adjust the training intensity and mode in real-time to ensure the rehabilitation effect.

For telemedicine, the manipulator systems can realize off-site surgery and examination through remote control. For example, in remote areas, doctors can perform surgeries or examinations by remote control of robots, solving the problem of uneven distribution of medical resources. Through the high-speed network connection, doctors can monitor the operation of the manipulator in real-time, which improves the feasibility and safety of telemedicine.

Manipulators are utilized in assisted nursing to help patients complete some basic activities in their daily lives, such as eating, dressing, and washing. The nursing robotic arm can help patients with limited mobility to live autonomously through voice or gesture commands, which improves their life quality.

1.2 Research Motivation

With the continued expansion of the robotics industry, the scope of robots in everyday life is poised to increase, thereby placing higher demands on the intelligent evolution of robots. Conventional methodologies for robot learning detect the environment through sensors, coupled with extensive computational processes executed within simulated environments, all in the pursuit of developing logical motion planning strategies for robots during task execution. This approach, however, requires substantial time and has high requirements on hardware performance. In stark contrast, human execution of analogous tasks is simple and intuitive. Therefore, one promising way to enhance robot intelligence involves learning from human demonstration, wherein humans assume the role of instructors. Within this framework, robots imitate and learn from demonstration (LfD), thereby elevating their behavioral dexterity.

For capturing human demonstration data, one approach is to use physical sensors. However, these sensors can be cumbersome, requiring individuals to wear various devices. They are often expensive, require calibration, and may inhibit certain motions, which can be especially problematic in scenarios involving multiple demonstrations. Another common method involves dragging the robot end-effector. This approach, while effective in capturing motions, offers limited flexibility and often struggles with recording the rotational aspects of the end-effector, which are crucial for dexterous tasks.

This thesis aims to propose a vision-based LfD framework to capture and process the human demonstration data conveniently. To execute more tasks, path planning methods and controllers are developed to make the system more adaptive and robust.

1.3 Thesis Contributions and Outline

Compared to existing literature, this thesis has the following novelties and contributions:

1. **A new framework**: The proposed framework offers a new dexterous manipulation solution with human-like behaviors through multiple demonstrations and executions of the demonstrated human motions for robotic manipulators.

2. **Robust and dexterous demonstrations**: Multiple demonstrations were acquired and processed using the dexterous motion framework. By employing dynamic time warping (DTW), Gaussian mixture model (GMM), and Gaussian mixture regression (GMR) for data processing, our approach effectively extracts common features from multiple demonstrations with their variances. This process improves the smoothness of the movements and reduces uncertainties associated with a single demonstration.

3. **Adaptive trajectory planning with dynamic movement primitives (DMP)**: Incorporating variances from multiple demonstrations into the DMP allows for adjustments in the influence of virtual force term on the trajectories. The feasibility of integrating virtual force term into the DMP was verified in the experimental testing with obstacle avoidance.

4. **Non-singular terminal sliding mode (NTSM) controller for unknown payload**: The implementation of the controller reduces the impact of unknown external disturbances. This controller plays an important role in tracking error minimization, thereby enhancing the system's overall robustness against disturbances and enabling the system to effectively complete a broader range of tasks.

The vision-based human hand dexterous motion detection system created in this study has been utilized for teleoperation and published in the following conference proceeding: N. Chen, L. Wan, Q.G. Chen and Y.-J. Pan, "Real Time Vision-based Human Hand Motion Tracking and Grasping for a Robotic Manipulator with Soft Hand", in Proceedings of the 2023 CSME International Congress of Canadian Mechanical Engineering, vol. 6, 2023. Furthermore, the DMP algorithm is applied to realize path with different start and goal points. The related work has been published in ISIE 2024: N. Chen and Y.-J. Pan, "Vision-Based Dexterous Motion Planning by Dynamic Movement Primitives with Human Hand Demonstration", in Proceedings of the 33rd IEEE International Symposium on Industrial Electronics (ISIE), Ulsan, South Korea, 2024. The overall work of this thesis, including dexterous motion capture, multiple demonstration processing, adaptive path planning, and robust controller design, was submitted to the IEEE Transactions on Industrial Electronics (TIE) in April 2024. Please read the Appendix A for my publication list.

The contents of this thesis are organized as follows. Chapter 1 introduces the applications of manipulator systems in industries. Chapter 2 presents state-of-the-art literature that proposes works in the same domain for this research. Chapter 3 describes system devices and the overview framework of this thesis. Chapter 4 proposes the vision-based human hand motion detection framework and how to learn from multiple demonstrations. Chapter 5 introduces some modified DMP methods as the path planning methods and some controllers applied in the experiments. Chapter 6 concludes the preceding work and shows the future research directions.

Chapter 2

Literature Review

This chapter introduces an overview of recent literature in the field of learning from demonstrations, vision detection methods, path planning theory, and controller design to interact with the environment. The objective of this chapter is to provide an overview of relevant research and highlight the unique contributions of this thesis.

2.1 Learn from Demonstration

Learning from demonstration (LfD) serves as a foundational framework facilitating communication and comprehension between humans and robots, thereby enhancing their ability for effective collaboration. Robots building upon this framework are intelligent robotic systems capable of learning from humans and interacting with humans. LfD emerges as a highly promising avenue for the seamless integration of robots into the operational fabric of factories, healthcare facilities, and broader societal contexts.

In allowing robots to learn from human demonstrations, they can acquire a better representation of human intentions and preferences, thus enabling more effective collaboration on various tasks. For example, within manufacturing settings, collaborative robots can learn proper part assembly and specific production tasks through learning from human demonstrations. Deng et al. [5] proposed a method wherein manipulators imitate assembly operations by learning from human demonstration videos. This method entails mapping from two-dimensional (2D) hand poses to the three-dimensional (3D) assembly space based on a neural network. In the medical domain, collaborative robots can replicate demonstrations from a surgeon's movements, thereby assisting in surgical procedures and rehabilitation processes. Kim et al. [6] developed a deep network to navigate a surgical tool inside the eye with reliability, achieved through learning from human demonstrations. Their results evidenced high accuracy in both physical experimentation and simulation environments.

In certain tasks, learning human motion through robotic systems proves to be more effective. One approach involves capturing human motion data, such as joint angles or key point coordinates, and translating these data into inputs for the robot, enabling it to replicate human movements. Some studies employed depth cameras to gather human body information, which is subsequently processed through filtering techniques and robotic kinematics, allowing the robot to follow human trajectories accurately [7,8]. Another approach leverages the properties of the human body to control robotic motion. For instance, Matthew et al. [9] presented a new humanoid robot design, an actuator-aware kino-dynamic motion planner, and a landing controller as part of a practical system design to optimize trajectories and realize a spinning jump in the simulation.

To capture human demonstration data, one approach is to utilize physical sensors such as wearable inertial sensors. Li et al. [10] employed a new type of motion capture technology to capture human walking motion and establish the human body kinematics model, to realize the reconstruction of 3D human motion. However, these sensors are inconvenient because they require individuals to wear various devices. They are often expensive, demand calibration, and may constrain certain motions, which can be especially problematic in scenarios involving multiple demonstrations. For the manipulator demonstration, another commonly employed method involves physically dragging the robot end-effector. Ti et al. [11] physically guided the end-effector to execute a peg-in-hole task, enabling the manipulator to record the end-effector's Cartesian positions along the trajectory. Xing et al. [12] guided the Franka robot's end-effector to do carving tasks. A mechanism with handle separated measurement of the user-applied interaction force from the interaction with the environment. In 2023, Chen et al. [13] detected a human hand's 3D coordinate with two webcams, enabling the robot to pick up an object and place it in a human hand. Trajectories learning was applied to set trajectories of the end-effector, learned from the Franka Emika robot's dragging movements. Despite its effectiveness in capturing motions, this approach exhibits limited adaptability and often encounters difficulties in accurately capturing the rotational motions of the end-effector, a critical aspect for accomplishing dexterous tasks.

2.2 Vision Detection

Recent advancements in computer vision have enabled researchers to adopt vision-based methods for capturing human motion. A notable advantage of employing cameras lies in obviating the necessity for individuals to sensors, thereby offering a more expeditious and streamlined alternative to traditional data collection methods. In [14], Cai et al. used stereo cameras to track the position of human-driven objects, facilitating the subsequent emulation of these trajectories by robotic systems. In 2020, Lin et al. [15] presented a vision-based approach to avoid moving obstacles in a dynamic environment. A depth camera was utilized to estimate the position, velocity, and size of the obstacles, and robust collision avoidance was achieved by a model predictive controller. Spaa et al. [16] utilized a human body motion capture system to capture and analyze human body posture data, enabling predictive adjustments in the controller to the robotic arm's posture. This adaptive approach ensures that the human operator can work in an optimal ergonomic position, thus reducing effort and enhancing comfort during tasks.

In the research of human-robot interaction (HRI), the demand for human coordinate detection has widely appeared, which emerged a proliferation of camera-based skeletal detection tools. Cao et al. introduced OpenPose, a tool for the real-time extraction of multiple human skeletal structures from webcam videos [17]. It enables the real-time extraction of human skeletal structures from a webcam and is amenable to multi-person scenarios, although it demands relatively high hardware requirements. OpenPose is capable of detecting a total of 135 key points so it can be applied to extract information about the key points of the human body and analyze movement trends. Chen et al. controlled the movement of the walking-aid cane robot by determining whether there is a tendency for the human body to fall through changes in the coordinates of key points at the head, waist, and hips [18]. In 2023, Cai et al. detected multiple hand demonstrations using a depth camera and OpenPose to obtain a comprehensive translational trajectory and predict the endpoint by DMP [19].

Similarly, MediaPipe, another vision-based tool, focused on extracting human hand skeletal key points [20]. In comparison to OpenPose, MediaPipe holds the advantage of accurately and efficiently capturing 21 2D key points of the human hand, thus facilitating precise hand gesture acquisition. In [21], Chunduru et al. applied

Mediapipe and detected hand gestures to manipulate a virtual globe. However, these works did not explicitly consider the hand's quaternions in motion planning. To the best of the author's knowledge, there has been no application of DMP with both translational and rotational demonstrations captured by cameras. Incorporating quaternions in motion planning adds a layer of dexterity, and exploring this aspect could be a potential avenue for future research in enhancing manipulator control.

2.3 Path Planning

Once the human demonstration data is collected, the next step is to model the discrete demonstration trajectory for modifications. Dynamic movement primitives is a method used to generate and control movements based on demonstrations [22]. The goal of DMP is to simulate the dynamic properties and flexibility of human movements, enabling the generation of appropriate movements in new environments while considering human dynamics. In [23], the integration of DMP with adaptive control for path planning and force control was executed on curved surfaces. Additionally, Gams et al. enhanced DMP by incorporating virtual force terms, enabling it to adapt to new environments and realize obstacle avoidance [24]. Enhancing the robustness of DMP can involve the introduction of probability to mitigate the influence of noise, as shown in [25] and [19]. The adaptability of the DMP is verified by its ability to generate multiple trajectories with different starting and ending points based on a single training demonstration [26]. To embed the confidence level of the human demonstrator in the task execution and improve the system's robustness, we have implemented the variance from multiple demonstrations into the DMP trajectory generation. To the best of the author's knowledge, DMP has not been applied with variance learned from multiple demonstrations.

In the context of HRI, effective path planning is essential for both trajectory planning and obstacle avoidance algorithms. This is particularly crucial in complex environments, where the uncertainties of both the robot and the HRI must be taken into account. One study addressed this challenge by generating a tree-shaped geometrically feasible plan [27]. Certain tasks require the manipulator to perform repetitive actions, necessitating path planning that emphasizes repeatability to reduce computational complexity and enhance the robot's efficiency [28]. Additionally, path planning can be integrated with motion prediction of target objects. For example, some researchers employed pedestrian prediction algorithms using convolution neural networks to forecast object paths, thereby aiding manipulator navigation [29, 30].

When humans interact with robots, ensuring the safety and collision-free operation of the robot within its environment is significant, particularly in complex settings where robots should adapt their movements in real-time to accommodate environmental changes. In 2022, Zhang et al. [31] devised a predictor to estimate the 3D bounding boxes of obstacles and target objects using data from a depth camera. These predictions are integrated into a policy network alongside feature extraction, enabling the efficient learning of obstacle avoidance strategies. Furthermore, robots must navigate around humans safely. In 2021, Nascimento et al. [32] proposed the concept of a limited safety contour around obstacles to dynamically estimate the real-time distance between the robot and humans, even when the robot's visibility is blocked by obstacles. Chen et al. [33] addressed both sub-tasks by utilizing a depth camera to capture environmental information, extracting the point cloud of the target via a segmentation algorithm, and employing a potential field for path planning to enable the manipulator to avoid obstacles and environmental constraints such as tabletops.

2.4 Controller Design

The previous three sections describe the high-level behavioral design of the manipulator system control, that is how to obtain the input reference trajectory of the robot. For the manipulator itself, we need to design the controller to adapt to different working environments.

The most common control methods for manipulators are proportional-derivative (PD) controller and impedance controller, due to their ease of implementation and effectiveness. The impedance controller in cartesian space is the default controller for the Franka Emika robot. In [13], Chen et al. applied an adaptive PD controller to achieve trajectory tracking with the Franka manipulator. In [34], Heshmati et al. proposed a cooperative impedance controller for multiple underwater vehicle manipulator systems to realize object transportation. However, these two control methods may face challenges in accurately following desired trajectories in the presence of external disturbances, such as the weight of a grasped object. To address the limitations of

PD and impedance controllers and enhance robustness against external disturbances, sliding mode control (SMC) has been applied to linear and nonlinear systems [35]. In 2021, Ke et al. [36] presented an adaptive sliding mode control scheme that incorporates a super twisting algorithm, a low-pass filter, and an adaptive law to achieve free chattering. Notably, non-singular terminal sliding mode control has been proposed to achieve sliding surfaces and zero error in finite time while avoiding singularities [37], whose control input does not contain the error term in the denominator, which eliminates the risk of zero division and encountering a singularity. In [38], Shen et al. designed an adaptive NTSM controller to synchronize the position of nonlinear multiple manipulator systems. A group of Phantom Omni robotic devices were used to carry out numerical simulations. In 2023, Wan et al. [39] demonstrated the effectiveness of the NTSM controller in a multi-robot system, successfully reducing errors induced by internal and external disturbances.

Similarly, reinforcement learning (RL) is also applied to sliding mode control aims to suppress the vibration. In [40], Long et al. used sliding mode control to track the desired trajectory and simultaneously generated a compensating torque to decrease the chattering in the tip of a hybrid manipulator. The compensate torque was calculated by an actor-critic network. Dimeas et al. [41] applied RL to impedance controllers, facilitating adaptation to operator skills. By formulating the RL objective to minimize jerk, the proposed controller autonomously learns suitable damping characteristics for effective collaboration without prior knowledge of target positions. In [42], Sangiovanni et al. applied both centralized and decentralized integral sliding mode controllers and designed a deep RL decision maker to generate rewards to switch the controller. In 2023, Gong et al. [43] proposed RL-based sliding mode control and designed the reward function to optimize the value function. The output of the controller consists of both the sliding mode part and the weighting part trained by RL.

Chapter 3

System Description

This chapter introduces the experimental equipment used for this thesis, containing hardware facilities as well as software systems and software toolkits. The hardware facilities are located in the Advanced Control and Mechatronics (ACM) Laboratory at Dalhousie University.

3.1 Hardware

This section shows the hardware to collect human demonstration and to execute the robot movement.

3.1.1 Franka Emika Robot

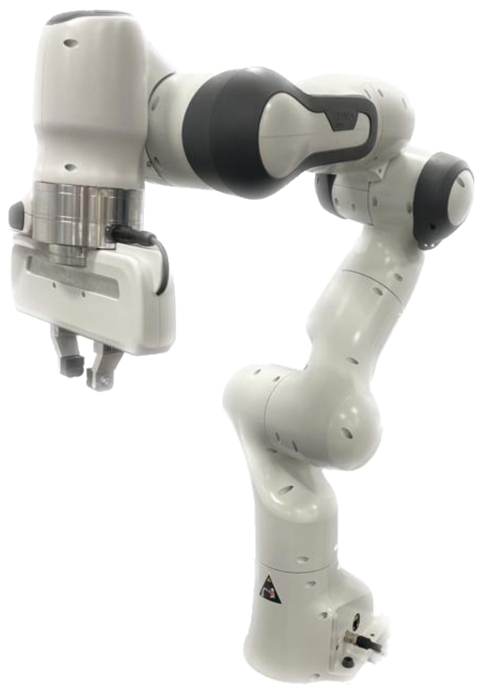


Figure 3.1: Franka Emika Robot

As shown in Fig. 3.1, the manipulator used in this thesis is the 7-degree-of-freedom (DoF) Franka Emika robot, which offers a maximum 3 kg payload. The repeatability of the Franka Emika robot is 0.1 mm so the interface for joint torque control enables dexterous control. The joints provide a 1 kHz signal transmission frequency to ensure a smooth control process. The above performances make it good platform at force and torque control and human-robot interaction.

Denavit-Hartenberg (DH) parameters are the four parameters associated with a particular convention for attaching reference frames to the links of a robot manipulator. d represents offset along previous z-axis to the common normal. θ represents angle about previous z-axis from old x-axis to new x-axis. a represents the length of the common normal. Assuming a revolute joint, this is the radius about the previous z-axis. α represents angle about common normal, from old z-axis to new z-axis. The DH parameters for the manipulator's kinematic chain are shown in Table 3.1 [44].

Table 3.1: Denavit-Hartenberg parameters of Franka Emika Robot

Joint	a (m)	d (m)	α (rad)	θ (rad)
Joint 1	0	0.333	0	θ_1
Joint 2	0	0	$-\pi/2$	θ_2
Joint 3	0	0.316	$\pi/2$	θ_3
Joint 4	0.0825	0	$\pi/2$	θ_4
Joint 5	-0.0825	0.384	$-\pi/2$	θ_5
Joint 6	0	0	$\pi/2$	θ_6
Joint 7	0.088	0	$\pi/2$	θ_7
Flange	0	0.107	0	0

The dynamics of the manipulator in joint space are:

$$\mathcal{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} + \mathcal{G}(\boldsymbol{q}) = \boldsymbol{\tau} + \boldsymbol{\tau_d}, \tag{3.1}$$

where $\mathcal{M}(\boldsymbol{q}) \in \mathbb{R}^{7\times 7}$ represents the inertial matrix, $\mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q}) \in \mathbb{R}^{7\times 7}$ represents the Coriolis and centripetal matrix, $\mathcal{G}(\boldsymbol{q}) \in \mathbb{R}^7$ is the gravity vector, $\boldsymbol{\tau}\in \mathbb{R}^7$ is the control torque input vector, and $\boldsymbol{\tau_d}\in \mathbb{R}^7$ represents the accumulation of disturbances from internal friction, unmodelled dynamics, and external disturbances. $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}\in \mathbb{R}^7$ are the joint angle, velocity, and acceleration vectors, respectively.

Since the human demonstration data exists in the Cartesian space, the trajectory planning should also be in the Cartesian space. The dynamic equation Eq. (3.1) can be transformed to Cartesian space. In Cartesian space, the end-effector pose is denoted as $\boldsymbol{x_m} = \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{m}}^T, & \boldsymbol{H}_{\boldsymbol{m}}^T \end{bmatrix}^T \in \mathbb{R}^7$ where $\boldsymbol{p_m} = \begin{bmatrix} x_m, & y_m, & z_m \end{bmatrix}^T \in \mathbb{R}^3$ is the translational position and $\boldsymbol{H_m} = \begin{bmatrix} \eta_m & \boldsymbol{\epsilon}_{\boldsymbol{m}}^T \end{bmatrix}^T \in \mathbb{R}^4$ is the rotational quaternion. The quaternion is made up of $\eta_m = q_{m_w}$, a scalar denoting the real part, and $\boldsymbol{\epsilon_m} = \begin{bmatrix} q_{m_x}, & q_{m_y}, & q_{m_z} \end{bmatrix}^T$, a vector denoting the imaginary part. The control input is $\boldsymbol{u} \in \mathbb{R}^6$.

The transformation from the joint space to the Cartesian space is

$$\begin{bmatrix} \dot{\boldsymbol{x}}_m \\ \ddot{\boldsymbol{x}}_m \end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\dot{\boldsymbol{q}} \\ J(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \end{bmatrix}, \tag{3.2}$$

where $J(\boldsymbol{q}) \in \mathbb{R}^{6\times 7}$ is the Jacobian matrix, $\dot{\boldsymbol{x}}_m = \begin{bmatrix} \dot{\boldsymbol{p}}_m^T, & \boldsymbol{\omega}_m^T \end{bmatrix}^T, \ddot{\boldsymbol{x}}_m = \begin{bmatrix} \ddot{\boldsymbol{p}}_m^T, & \dot{\boldsymbol{\omega}}_m^T \end{bmatrix}^T \in \mathbb{R}^6$, where $\boldsymbol{\omega}_m \in \mathbb{R}^3$, $\dot{\boldsymbol{\omega}}_m \in \mathbb{R}^3$ are the angular velocity and acceleration of the end-effector, respectively.

With Eq. (3.1) and Eq. (3.2), the dynamic equation of the manipulator in Cartesian space can be represented as

$$\bar{\mathcal{M}}(\boldsymbol{q})\ddot{\boldsymbol{x}}_m + \bar{\mathcal{C}}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{x}}_m + \bar{\mathcal{G}}(\boldsymbol{q}) = \boldsymbol{u} + \boldsymbol{u_d}, \tag{3.3}$$

where

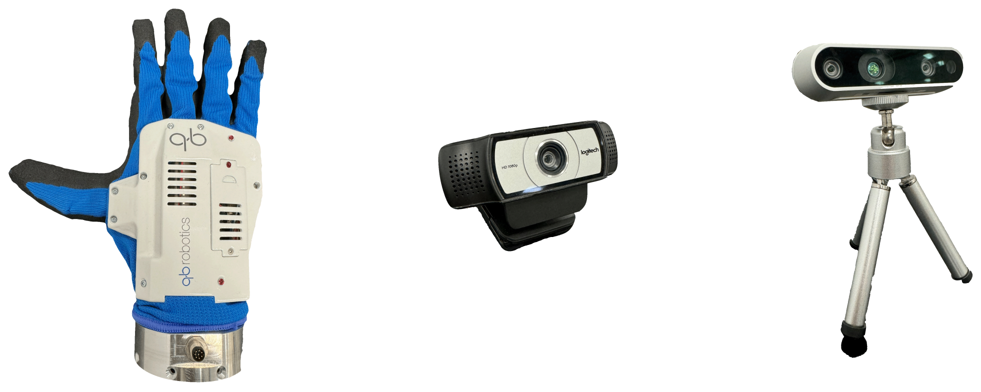
$$\begin{aligned} \bar{\mathcal{M}}(\boldsymbol{q}) &= J(\boldsymbol{q})^{-T}\mathcal{M}(\boldsymbol{q})J(\boldsymbol{q})^{-1}, \\ \bar{\mathcal{C}}(\dot{\boldsymbol{q}}, \boldsymbol{q}) &= J(\boldsymbol{q})^{-T}(\mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q}) - \mathcal{M}(\boldsymbol{q})J(\boldsymbol{q})^{-1}\dot{J}(\boldsymbol{q}))J(\boldsymbol{q})^{-1}, \\ \bar{\mathcal{G}}(\boldsymbol{q}) &= J(\boldsymbol{q})^{-T}\mathcal{G}(\boldsymbol{q}), \quad \text{and} \quad \boldsymbol{u} = J(\boldsymbol{q})^{-T}\boldsymbol{\tau}. \end{aligned}$$

The Jacobian and dynamics in Cartesian space may be separated into translational and rotational components as

$$J(\boldsymbol{q}) = \begin{bmatrix} J_p(\boldsymbol{q}) & J_H(\boldsymbol{q}) \end{bmatrix}^T, \quad \bar{\mathcal{M}}(\boldsymbol{q}) = \begin{bmatrix} \bar{\mathcal{M}}_p(\boldsymbol{q}) & \bar{\mathcal{M}}_H(\boldsymbol{q}) \end{bmatrix}^T.$$

3.1.2 qb-Softhand

The qb-Softhand is a soft-robotics-based anthropomorphic robotic hand designed for safe interaction with its environment, objects, and humans. This innovative technology reduces the risk of harm to operators, product damage, and robot wear. It boasts adaptability, effortlessly gripping various objects without requiring control adjustments. With its single-motor actuation, qb-Softhand is not only affordable but also easy to use and integrate, making it a versatile and efficient grasping solution.



(a) qb Softhand (b) Logitech Web Camera (c) RealSense Depth Camera

Figure 3.2: Hardware used in this work

3.1.3 Camera

Two types of cameras are used in the thesis: a webcam that can only acquire red-green-blue (RGB) images and a depth camera that can acquire both RGB and depth images.

In the web camera based detection section, two Logitech C930E HD cameras used for the MediaPipe detection have a video resolution of 1080p and 30 frames per second (FPS). For the depth camera based detection, the experimental setup uses the RealSense D435i depth camera, featuring a dual-camera system capable of capturing both light and depth information. The RGB sensor has a field of view of $69° \times 42°$. The depth sensor has a maximum rate of 90 FPS, and the RGB sensor has a maximum rate of 30 FPS. In the conducted experiments, the chosen frame rate was 20 FPS, resulting in a sampling time of 0.05 s. This choice balances capturing sufficient data and managing computational resources. The depth accuracy is specified to be smaller than 2% at a distance of 2 m, and all experimental data falls within this specified range. This ensures the reliability and precision of the depth information captured by the camera throughout the experiment.

3.2 Software

This section presents the software toolkits to collect human hand key points and the software system to communicate with each device.

3.2.1 Robot Operating System

Robot operating system (ROS) is an open-source robotics software platform designed to simplify and speed up the robotics development process. It provides a range of tool libraries to enable developers to create complex robotics applications more easily. ROS integrates well with various simulation environments, such as Gazebo, enabling developers to test their robotics applications in a virtual environment.

In this thesis, we have mainly used the communication function of ROS to send motion commands as well as motion coordinates to the robot. In order to realize this functionality, we use nodes and topics in ROS. Nodes are the basic units in ROS, and each node performs a specific task. For example, one node may be responsible for sensor reading and another for motion control. Topics, on the other hand, are a way for nodes to communicate with each other, and nodes can publish or subscribe to topics. For example, in this thesis, we use a node in a computer to publish the desired trajectory to a topic, and the robot's control node subscribes to the topic to get the data of the desired trajectory to realize the robot's control. ROS enables data transfer between different devices on the same network.

3.2.2 MediaPipe

This thesis uses MediaPipe to collect the human hand's information, and each palm collects 21 points, including four joint points of each finger and wrist joint points. MediaPipe is a machine learning framework developed by Google for video and image processing tasks, including hand recognition [20]. MediaPipe hand recognition can track the position of human hands in the view of the camera in real-time. The framework uses machine learning based techniques for hand pose estimation and tracking, enabling hand recognition in different contexts. Compared with other skeleton recognition tools, it does not require high computer performance, and it also has the functions of human skeleton recognition and face recognition. The function of MediaPipe's hand recognition has broad application prospects in fields such as virtual reality, games, smart homes, and medical treatment.

3.3 System Overview

The overview of this thesis is shown in Fig. 3.3. A comprehensive framework for vision-based learning from demonstrations to generate dexterous robot motions is proposed, which is one of the main contributions of the thesis. Vision-based detection methods are applied to extract the 3D coordinates of human hand key points. To define the robot motion, translational Cartesian position, rotational quaternion motion, and grasping distance of the gripper are defined. Mean filters are applied to pre-process the single demonstration so as to ignore the noise. On the other hand, DTW, GMM, and GMR are employed to synthesize a comprehensive trajectory from multiple datasets. The variance derived from these demonstrations, along with force coupling terms, is implemented into the DMP framework, thereby enhancing the adaptability and robustness of motion planning in dynamic environments. Impedance controller as well as NTSM-PD controller are developed to ensure the robustness of the manipulator tracking during different pick-and-place tasks.

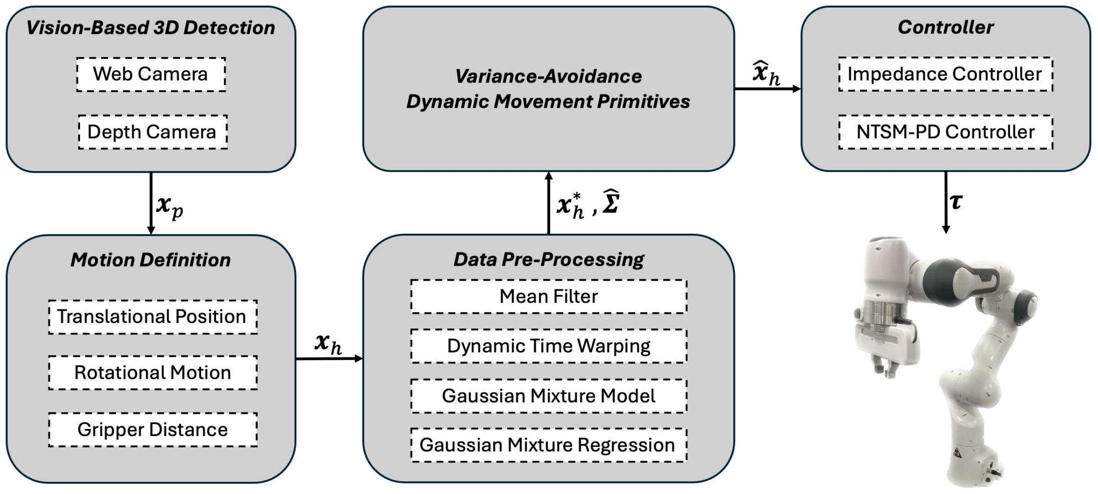


Figure 3.3: Schematic diagram of the work in the thesis

Chapter 4

Vision-Based Learning from Multiple Demonstrations

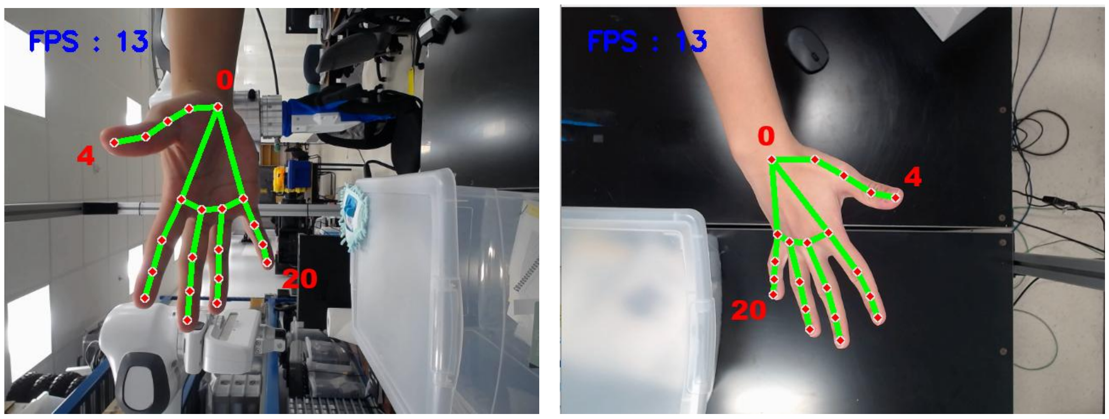
This chapter introduces the framework to learn from multiple demonstrations with vision-based detection. 3D coordinate detection methods are designed to capture the human hand key points conveniently. With the 3D coordinates, dexterous motions can be defined as the desired input of the robot. To improve the robustness of the system, data processing methods with multiple demonstrations are utilized in this chapter.

4.1 3D Coordinate Detection

This section presents two different methods to detect the 3D coordinate of the human hand key points.

4.1.1 Web Camera Based Detection

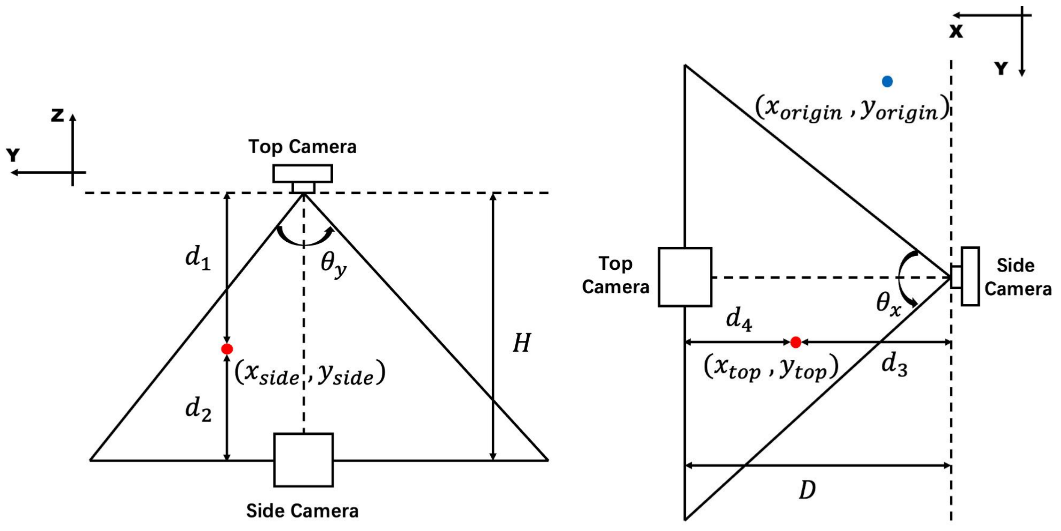
Since webcams can only acquire RGB images, one single camera cannot acquire enough coordinate data, so two webcams are used to provide the robot with x, y, and z-axis position information of the hand for the manipulator to track.



(a) Side Camera Frame (b) Top Camera Frame

Figure 4.1: Mediapipe view in different camera frames

First, MediaPipe is utilized to obtain the 2D pixel coordinates of the hand in the camera frame. In this thesis, each camera captures a different angle of a single hand. As shown in Fig. 4.1, each view of the hand can be extracted with red points and green lines. There are 21 red points on each hand to represent the key points and green lines form the skeleton to connect the key points. Red numbers are shown to identify the index of key points. Number 0 denotes the wrist position, Number 4 is the position of the thumb fingertip, and Number 20 is the little finger fingertip. Blue numbers in the top left side represent the FPS, which is affected by the computer's performance.



(a) Side View (b) Top View

Figure 4.2: Top and side views to calculate distance

Second, a method of 3D localization of the key points with two cameras is developed. The two cameras are set perpendicularly and the side camera should be on the edge of the top camera's frame. As shown in Fig. 4.2, the depths of two cameras d_1, d_3 can be concluded as Eq. (4.1) and Eq. (4.2).

$$d_1 = H - d_2, \tag{4.1}$$

$$d_3 = D - d_4, \tag{4.2}$$

where H and D are the distances between two cameras in x and z directions, which are measured in advance with $H = 0.60m$, $D = 0.35m$.

In the pixel coordinate system, the actual distance l can be calculated by Eq. (4.3).

$$l = d\ tan(\frac{n}{N}\theta), \tag{4.3}$$

where d is the depth, n is pixel distance, N is the pixel value of width or height of the camera image and θ is the view angle of the camera. Term $\frac{1}{N}\theta$ is one pixel's angle in the picture, and $d\ tan(\frac{1}{N}\theta)$ is the real length of one pixel so actual distance l with n pixels can be concluded as Eq. (4.3). So d_2 and d_4 can be calculated by Eq. (4.4) and Eq. (4.5).

$$d_2 = d_3 tan(\frac{(\frac{Y}{2} - y_{side})}{Y}\theta_y), \tag{4.4}$$

$$d_4 = d_1 tan(\frac{(x_{top} - \frac{X}{2})}{X}\theta_x), \tag{4.5}$$

where θ_x and θ_y are the view angles of the cameras, X and Y are the pixel values of the width and height of the camera image. θ_x, θ_y, X, and Y are constant parameters related to the camera. $(x_{top},\ y_{top})$ and $(x_{origin},\ y_{origin})$ represents the pixel coordinates of the hand's point and origin point in the top camera image, $(x_{side},\ y_{side})$ represents that in the side camera. In this case, $X = 640,\ Y = 480,\ \theta_x = 66^o,\ \theta_y = 57^o,\ x_{origin} = 520,\ y_{origin} = 130$.

The depths, d_1 and d_3 can be calculated by Eqs. (4.1)-(4.5) as

$$d_1 = \frac{H - Dtan(\frac{(\frac{Y}{2} - y_{side})}{Y}\theta_y)}{1 - tan(\frac{(x_{top} - \frac{X}{2})}{X}\theta_x)tan(\frac{(\frac{Y}{2} - y_{side})}{Y}\theta_y)}, \tag{4.6}$$

$$d_3 = \frac{D - Htan(\frac{(x_{top} - \frac{X}{2})}{X}\theta_x)}{1 - tan(\frac{(x_{top} - \frac{X}{2})}{X}\theta_x)tan(\frac{(\frac{Y}{2} - y_{side})}{Y}\theta_y)}. \tag{4.7}$$

With these two depths, we can calculate the 3D coordinate of each point by

$$\boldsymbol{p}_h = \begin{bmatrix} x_h, y_h, z_h \end{bmatrix}^T = \begin{bmatrix} d_1 tan(\frac{(x_{origin} - x_{top})}{X}\theta_x) \\ d_1 tan(\frac{(y_{origin} - y_{top})}{Y}\theta_y) \\ H_{top} - d_1 \end{bmatrix}, \tag{4.8}$$

where H_{top} is the height of top camera. In this thesis, $H_{top} = 1.0m$ [45].

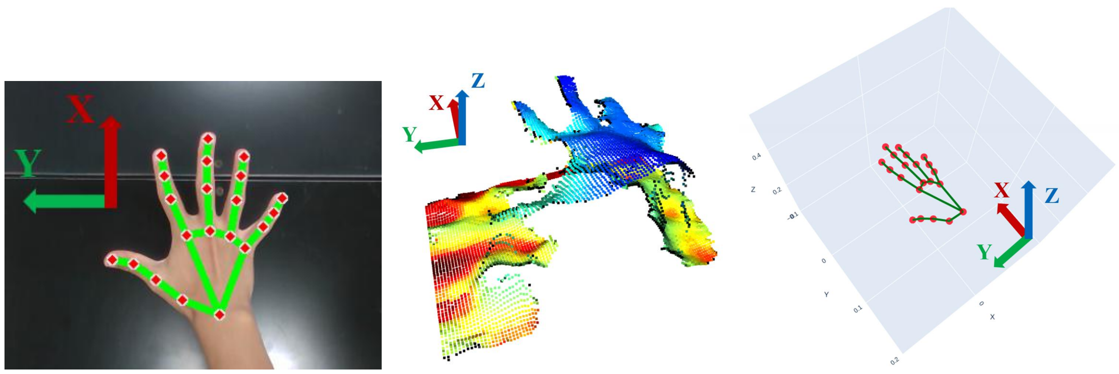
4.1.2 Depth Camera Based Detection

The depth camera is capable of capturing both RGB and depth images simultaneously while maintaining their alignment. To ensure consistency and facilitate precise correspondence, both RGB and depth images are uniformly set to a resolution of 640×480. This standardization enables precise correspondence between points in two images. As shown in Fig. 4.3 (a), the first step employs MediaPipe, which identifies the hand's key points within the RGB image. The 2D pixel coordinates corresponding to these key points are extracted. By associating the index of these pixels with the depth image, the depth information for each pixel is obtained, as illustrated in Fig. 4.3 (b). The pixel coordinates do not represent real-world coordinates and therefore, a coordinate transformation from the pixel coordinates, x_p, y_p to real-world spatial coordinates, $\boldsymbol{x}_h = \left[\boldsymbol{p}_h^T, \quad \boldsymbol{H}_h^T\right]^T$, is required.

Using Eq. (4.8), the 3D coordinates of the hand are represented as

$$\boldsymbol{p}_h = \left[x_h, y_h, z_h\right]^T = \begin{bmatrix} d\tan(\frac{(x_p - \frac{X}{2})}{X}\theta_x) \\ d\tan(\frac{(y_p - \frac{Y}{2})}{Y}\theta_y) \\ H - d \end{bmatrix}, \tag{4.9}$$

where H is the height of camera, θ_x and θ_y are the view angles of camera, X and Y are the resolution. In this thesis, $H = 1.0$ m, $X = 640$, $Y = 480$, $\theta_x = 69°$, $\theta_y = 42°$. The final 3D hand is shown in Fig. 4.3 (c) with the python toolbox Plotly.



(a) RGB image with MediaPipe (b) Depth image (c) 3D hand

Figure 4.3: The proposed 3D hand coordinate generation

4.2 Motion Definition

This section introduces the definition of translational motion, rotational motion, and gripper distance from 3D hand coordinates. These motions define the manipulator's end-effector movement.

4.2.1 Translational Motion

Translational motion is the most common motion when controlling the end-effector of a manipulator. In the previous part, we can get 3D coordinates of 21 key points of hand. So we set the trajectory of the wrist as the translational motion as Eq. (4.10).

$$\boldsymbol{p}_h = \begin{bmatrix} x_w, y_w, z_w \end{bmatrix}^T, \tag{4.10}$$

where (x_w, y_w, z_w) denotes the position of wrist.

4.2.2 Rotational Motion

In addition to controlling the 3D translational coordinates of the end-effector, equal significance is attributed to managing the orientation of the end-effector and the grasping of the gripper, which are easily overlooked elements of the current research, as these movements are not conveniently collected in the demonstration. As shown in Fig. 4.4, these motions can be calculated and corresponded through the 3D coordinates of the wrist, thumb tip, and index fingertip, which correspond to points 0, 4, and 8 in Mediapipe, respectively.

To represent the orientation of the end-effector, the Euler angles of yaw, pitch, and roll orientations are as in Eq. (4.11).

$$\begin{bmatrix} \psi_h \\ \theta_h \\ \phi_h \end{bmatrix} = \begin{bmatrix} \arctan(\frac{x_i - x_t}{y_t - y_i}) \\ \arctan(\frac{z_w - z_t}{x_t - x_w}) \\ \arctan(\frac{z_i - z_t}{y_t - y_i}) \end{bmatrix}, \tag{4.11}$$

where the yaw angle ψ_h is the rotation about the z-axis, the pitch angle θ_h is the rotation about the y-axis and the roll angle ϕ_h is the rotation about the x-axis. (x_i, y_i, z_i) and (x_t, y_t, z_t) denote the positions of index fingertip and thumb tip, respectively.

While Euler angles offer a straightforward and intuitive method for the orientation representation, the Franka Emika robot uses quaternions $\boldsymbol{H}$ as its chosen representation for orientation, so the conversion from Euler angles to quaternions is needed.

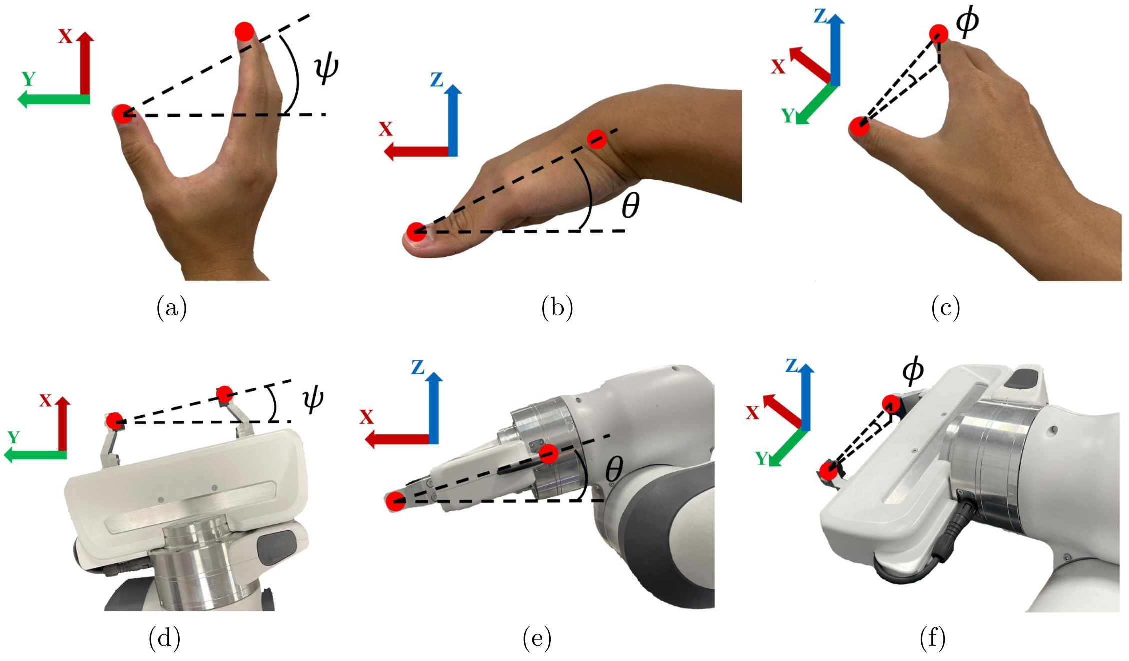


Figure 4.4: Euler angles generation. (a)-(c) are the yaw, pitch, and roll of hand and (d)-(f) are the yaw, pitch, and roll of gripper

Prior to executing this transformation, it is imperative to understand the quaternion multiplication operation. Assume $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ are quaternions, which can be represented by Eq. (4.12).

$$\begin{cases} \boldsymbol{H_1} = \eta_1 + \boldsymbol{\epsilon_1} = q_{w1} + q_{x1}\boldsymbol{i} + q_{y1}\boldsymbol{j} + q_{z1}\boldsymbol{k}, \\ \boldsymbol{H_2} = \eta_2 + \boldsymbol{\epsilon_2} = q_{w2} + q_{x2}\boldsymbol{i} + q_{y2}\boldsymbol{j} + q_{z2}\boldsymbol{k}. \end{cases} \tag{4.12}$$

Then the multiplication of $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ can be obtained as by Eq. (4.13), which is also called Hamilton product [46].

$$\begin{aligned} \boldsymbol{H_1}\boldsymbol{H_2} &= \eta_1\eta_2 - \boldsymbol{\epsilon_1} \cdot \boldsymbol{\epsilon_2} + \eta_2\boldsymbol{\epsilon_1} + \eta_1\boldsymbol{\epsilon_2} + \boldsymbol{\epsilon_1} \times \boldsymbol{\epsilon_2} \\ &= \begin{bmatrix} q_{x1}q_{w2} + q_{w1}q_{x2} + q_{y1}q_{z2} - q_{z1}q_{y2} \\ q_{y1}q_{w2} + q_{w1}q_{y2} + q_{z1}q_{x2} - q_{x1}q_{z2} \\ q_{z1}q_{w2} + q_{w1}q_{z2} + q_{x1}q_{y2} - q_{y1}q_{x2} \\ q_{w1}q_{w2} - q_{x1}q_{x2} - q_{y1}q_{y2} - q_{z1}q_{z2} \end{bmatrix}. \end{aligned} \tag{4.13}$$

Through the multiplication of three axes, the transformation equation between quaternions and Euler angles is as:

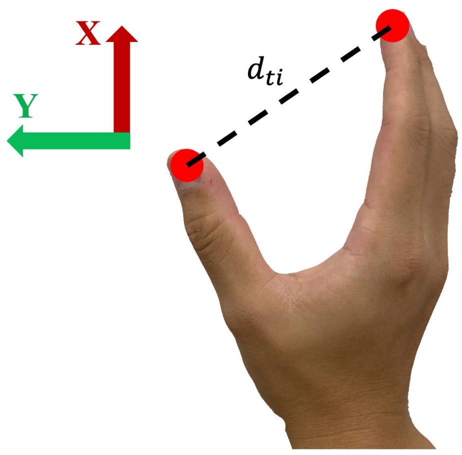
$$\boldsymbol{H}_h = \boldsymbol{H}_x(\phi_h)\boldsymbol{H}_y(\theta_h)\boldsymbol{H}_z(\psi_h) = \begin{bmatrix} C_{\frac{\phi_h}{2}} \\ S_{\frac{\phi_h}{2}} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} C_{\frac{\theta_h}{2}} \\ 0 \\ S_{\frac{\theta_h}{2}} \\ 0 \end{bmatrix} \begin{bmatrix} C_{\frac{\psi_h}{2}} \\ 0 \\ 0 \\ S_{\frac{\psi_h}{2}} \end{bmatrix}$$

$$= \begin{bmatrix} C_{\frac{\phi_h}{2}} C_{\frac{\theta_h}{2}} C_{\frac{\psi_h}{2}} + S_{\frac{\phi_h}{2}} S_{\frac{\theta_h}{2}} S_{\frac{\psi_h}{2}} \\ S_{\frac{\phi_h}{2}} C_{\frac{\theta_h}{2}} C_{\frac{\psi_h}{2}} - C_{\frac{\phi_h}{2}} S_{\frac{\theta_h}{2}} S_{\frac{\psi_h}{2}} \\ C_{\frac{\phi_h}{2}} S_{\frac{\theta_h}{2}} C_{\frac{\psi_h}{2}} + S_{\frac{\phi_h}{2}} C_{\frac{\theta_h}{2}} S_{\frac{\psi_h}{2}} \\ C_{\frac{\phi_h}{2}} C_{\frac{\theta_h}{2}} S_{\frac{\psi_h}{2}} - S_{\frac{\phi_h}{2}} S_{\frac{\theta_h}{2}} C_{\frac{\psi_h}{2}} \end{bmatrix},$$

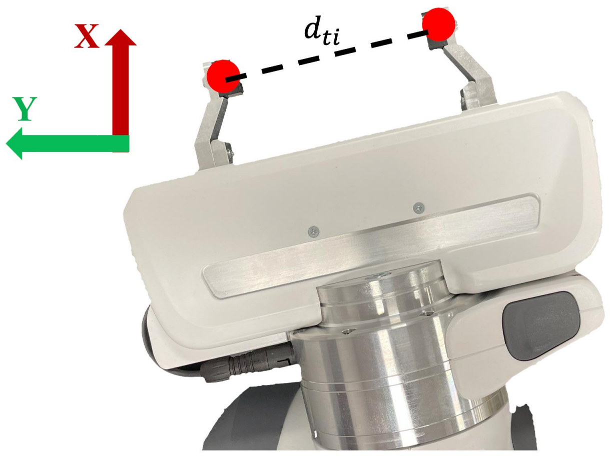
where $C_x = \cos(x)$ and $S_x = \sin(x)$ with $x = \frac{\theta_h}{2}, \frac{\psi_h}{2}, \frac{\phi_h}{2}$.

Given that the default configuration of the robot's end-effector is perpendicular to the ground, while the default hand posture in the human demonstration aligns parallel to the ground, an essential adjustment is mandated. We rotated the end-effector 90° around the y-axis, so the desired quaternion should be calculated as the demonstration quaternion multiplying the quaternion rotated 90° around the y-axis so that the end-effector will perform like the humans.

4.2.3 Gripper Distance



(a)



(b)

Figure 4.5: Gripper distance generation

For grasping, the distance between the thumb tip and the index fingertip d_{ti} can be defined as the gripper distance in Fig. 4.5, which can be calculated as

$$d_{ti} = \sqrt{(x_t - x_i)^2 + (y_t - y_i)^2 + (z_t - z_i)^2}. \tag{4.14}$$

If $d_{ti} < \bar{d}_{ti}$, the robot considers it as a grasping motion learned. Subsequently, the gripper will execute the grasping action, which reduces the gripper distance and stops when the contact force of the gripper reaches 3 N. $\bar{d}_{ti}$ may vary from different tasks. In this thesis, $\bar{d}_{ti} = 0.1$ m.

4.3 Data Processing

This section presents methods to ignore the noise for single demonstration and multiple demonstrations to elevate the robustness of the system.

4.3.1 Mean Filter

After obtaining the single motion trajectory and posture from the human demonstration, a mean filter is applied to smooth the raw data. A mean filter is a linear filter that takes the data in a window and calculates the mean and then returns the mean value calculated in the window. The advantages of this algorithm are its efficiency and simplicity of thought. Similarly, the disadvantages are obvious, when the length of the window is too large, the calculation of the mean value becomes fuzzy and many features are lost. The mean filter requires a one-dimensional vector of length N, denoted as $\boldsymbol{x_h}$, and the output vector $\boldsymbol{x_h^*}$ after applying an average smoothing filter with a window size of k given by Eq. (4.15).

$$x_{hi}^* = \frac{1}{k} \sum_{j=i-\frac{k-1}{2}}^{i+\frac{k-1}{2}} x_{hj}, \tag{4.15}$$

where x_{hi}^* represents the ith element in the output vector, x_{hj} corresponds to the jth element in the input vector and i ranges from $\frac{k-1}{2}$ to $N - \frac{k-1}{2}$.

4.3.2 Dynamic Time Warping

The raw demonstration data, limited by the inherently non-smooth motion of the human body and potential errors in camera calibration, requires processing. Using multiple demonstrations can mitigate errors and uncertainties associated with single demonstrations.

When individuals perform various demonstrations for the same task, the movements may exhibit similarities, but the time sequences are often not identical. For instance, in a pick-and-place experiment, the duration for lifting and lowering the object may not be exactly the same across different demonstrations. Therefore, the initial step in learning from multiple demonstrations is regularizing the time series associated with these diverse movements.

DTW is a method that gauges the similarity between two time series and aligns them effectively. DTW corresponds two series in different time steps, but not by Euclidean distance [4]. The fundamental concept is to identify the optimal match between two series by aligning them along the time axis. Suppose time series A as $\{a_1, a_2, ..., a_n\}$, B as $\{b_1, b_2, ..., b_m\}$, where n and m are their respective lengths. Define a Euclidean matrix E, where $\mathsf{E}(i, j)$ denotes the Euclidean distance of the ith element of A and the jth element of B. Next, define a cumulative distance matrix C, where $\mathsf{C}(i, j)$ denotes the minimum cumulative cost to reach the point (i, j) from the starting point, which is calculated as

$$\mathsf{C}(i,j) = \mathsf{E}(i,j) + min\{\mathsf{C}(i-1,j), \mathsf{C}(i,j-1), \mathsf{C}(i-1,j-1)\}. \tag{4.16}$$

After the computation of C, the optimal matching can be determined by backtracking along the path. The path is selected by initiating the backtrack from the endpoint in C and moving towards the starting point along the path with the minimum accumulated cost. Fig. 4.10 shows the correspondence between two time series. By selecting one time series as the reference, all other series can be aligned with the reference series using DTW. This alignment process generates new data with consistent time steps for all series, ensuring temporal homogeneity. After applying DTW, the multiple demonstrations, $\boldsymbol{X}_h = \begin{bmatrix} \boldsymbol{x}_{h_1}, & \boldsymbol{x}_{h_2}, & \cdots, & \boldsymbol{x}_{h_N} \end{bmatrix}$, are converted to multiple regularized demonstrations on the same time scale, $\bar{\boldsymbol{X}}_h = \begin{bmatrix} \bar{\boldsymbol{x}}_{h_1}, & \bar{\boldsymbol{x}}_{h_2}, & \cdots, & \bar{\boldsymbol{x}}_{h_N} \end{bmatrix}$, where N is the number of demonstrations.

Suppose we have a series A=$\{0.1, 0.2, 0.5, 0.5, 0.3, 0.1, 0.1, 0.1, 0.2, 0.3\}$, which has 10 elements, and a same length series B=$\{0.1, 0.1, 0.2, 0.3, 0.4, 0.5, 0.3, 0.2, 0.1, 0.2\}$, shown in Fig. 4.6. Firstly, calculate the Euclidean distance matrix of two series, as shown in Fig. 4.7. Each number in grid (i, j) represents the Euclidean distance between b_i and a_j. Darker colored squares represent larger distances and lighter colored squares represent smaller distances. Secondly, find a path from the bottom left corner of the Euclidean matrix to the top right corner such that the sum of the elements on the path is minimized. To realize this goal, it is necessary to calculate the cumulative matrix based on Eq. (4.16). The cumulative matrix is shown in Fig. 4.8. Each number in grid (i, j) represents the minimum cumulative distance from the bottom left corner to the point (i, j). Then, find the minimum cost path starting from the top right corner. Red lines represent the minimum cost path. Finally, corresponding elements in the path are shown in Fig. 4.10. For example, the minimum cost path goes through the point $(b_4,\ a_2)$ so the fourth point of series B corresponds to the second point of series A.

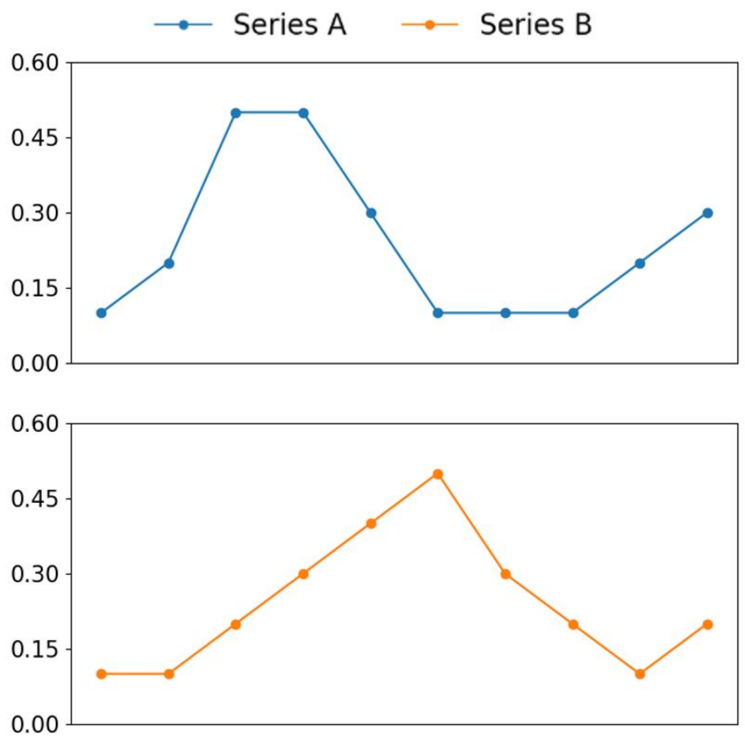


Figure 4.6: DTW example

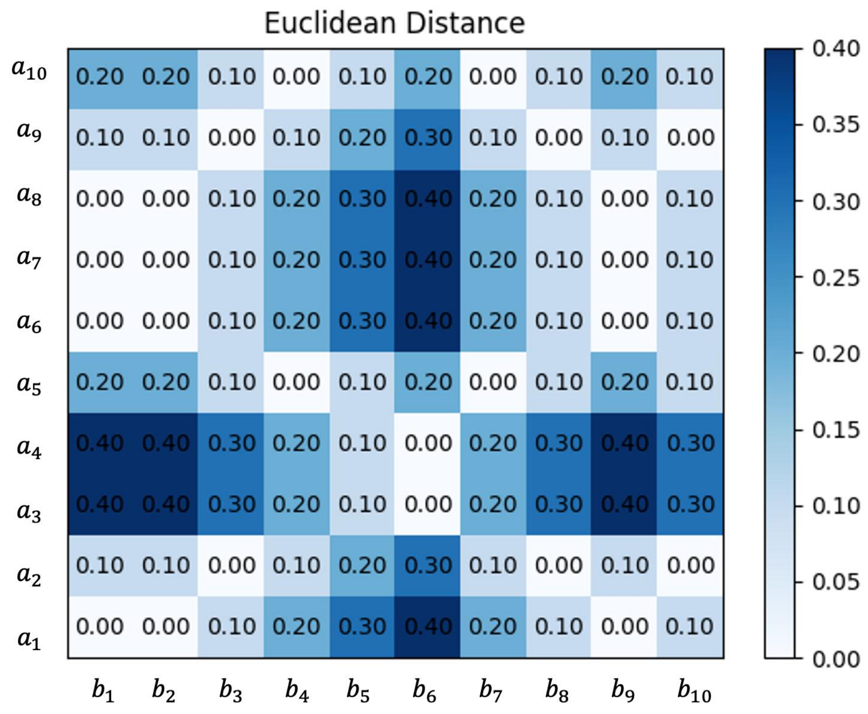


Figure 4.7: DTW Euclidean matrix

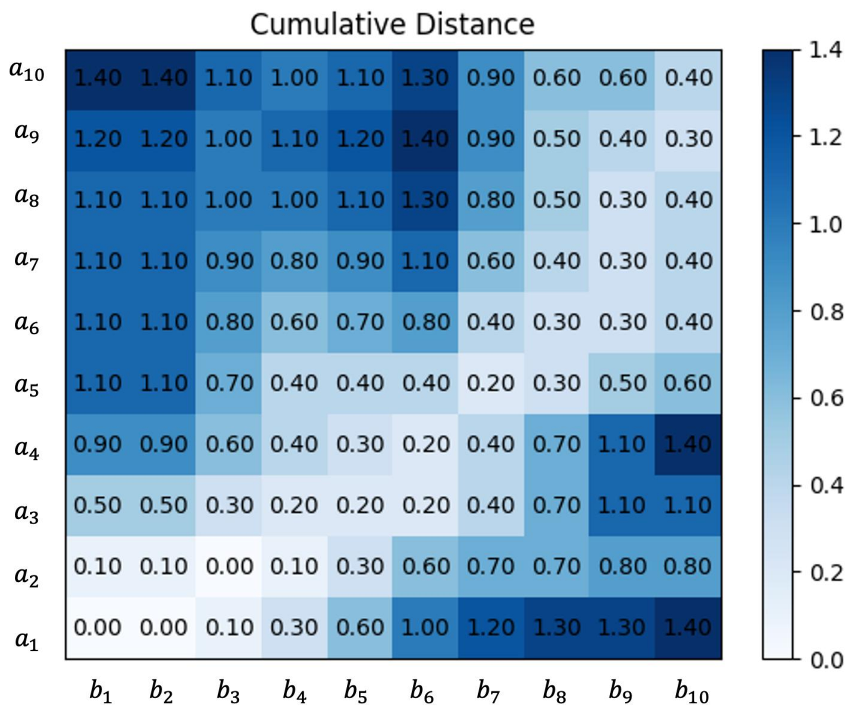


Figure 4.8: DTW cumulative matrix

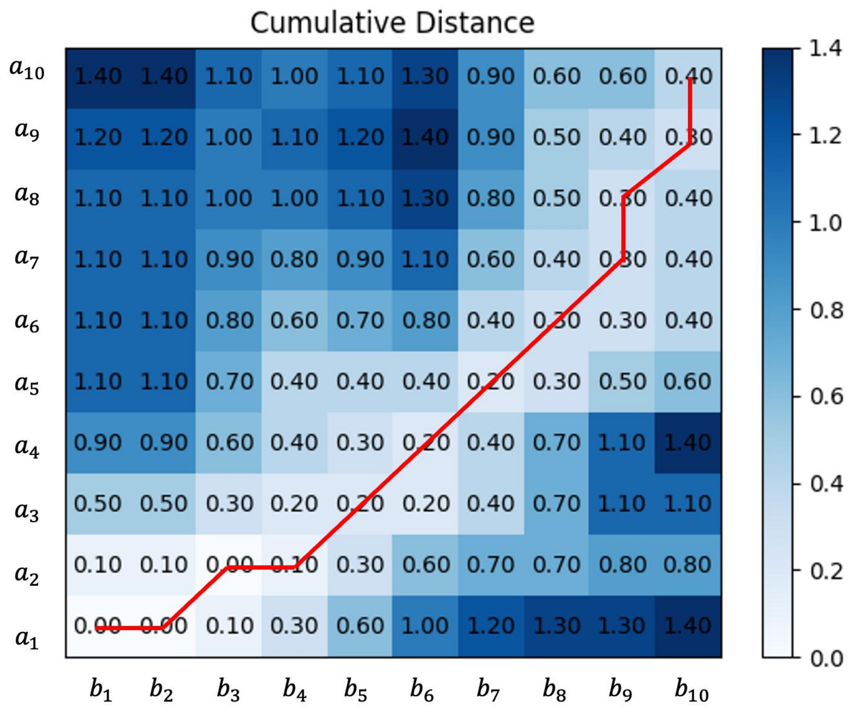


Figure 4.9: DTW cumulative matrix with minimum cost path

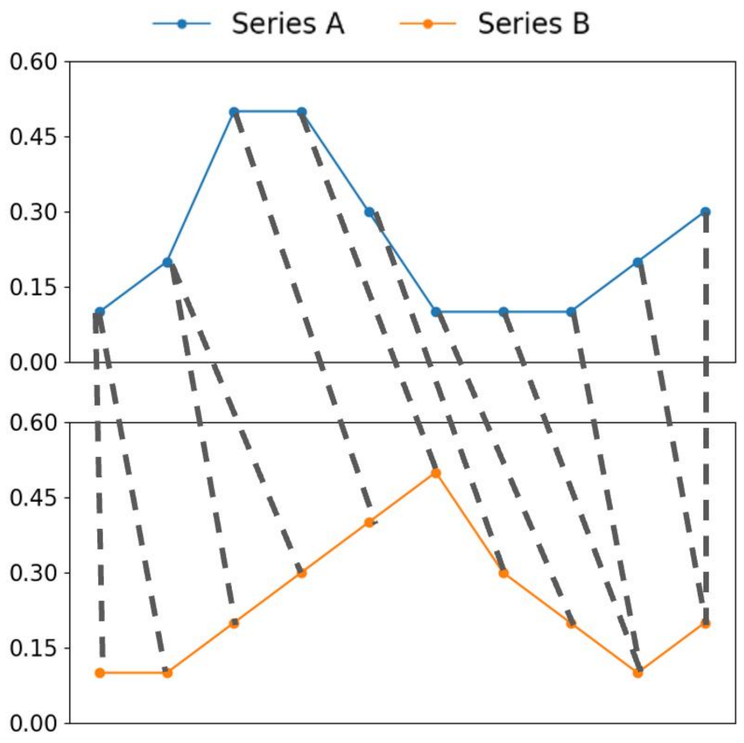


Figure 4.10: DTW correspondence

4.3.3 Gaussian Mixture Model

In scenarios where a single desired trajectory input is required by the robot, it becomes imperative to conclude a single trajectory from multiple demonstrations. Once all time series are regularized, the Gaussian mixture model can be applied to model the entire dataset using multiple Gaussian functions [47].

GMM is a probabilistic model designed for classifying and clustering data, assuming that the observed data is a mixture of multiple Gaussian distributions. Each Gaussian distribution is parameterized by its mean $\boldsymbol{\mu}_k$, covariance matrix $\boldsymbol{\Sigma}_k$, and mixing coefficient π_k as

$$\mathcal{P}(\boldsymbol{t}, \bar{\boldsymbol{X}}_{\boldsymbol{h}}) \sim \sum\nolimits_{k=1}^{K} \pi_k \cdot \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{4.17}$$

where $\boldsymbol{t}$ is time, $\bar{\boldsymbol{X}}_{\boldsymbol{h}}$ is the regularized pose of the hand, and π_k, $\boldsymbol{\mu}_k = \begin{bmatrix} \boldsymbol{\mu}_{t,k} \\ \boldsymbol{\mu}_{\bar{X}_h,k} \end{bmatrix}$, and $\boldsymbol{\Sigma}_k = \begin{bmatrix} \boldsymbol{\Sigma}_{tt,k} & \boldsymbol{\Sigma}_{t\bar{X}_h,k} \\ \boldsymbol{\Sigma}_{\bar{X}_h t,k} & \boldsymbol{\Sigma}_{\bar{X}_h\bar{X}_h,k} \end{bmatrix}$ represent kth prior probability, mean, and covariance, respectively. $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the probability density function (PDF) of multivariate Gaussian distribution and K is the number of Gaussian functions. The parameters of the GMM can be optimized iteratively by the expectation maximization (EM) algorithm [48].

4.3.4 Gaussian Mixture Regression

Gaussian mixture regression (GMR), often used in conjunction with GMM, regresses a single trajectory from the Gaussian models. This method enables the extraction of key features from multiple demonstrations, contributing to a more refined and consolidated representation of the desired motion [49].

After obtaining the GMM parameters, GMR can be applied to predict the conditional probability distribution of the corresponding trajectory $\boldsymbol{x}_{\boldsymbol{h}}^*$ for any new input $\boldsymbol{t}^*$ as in

$$\mathcal{P}(\boldsymbol{x}_{\boldsymbol{h}}^* | \boldsymbol{t}^*) = \sum\nolimits_{k=1}^{K} h_k(\boldsymbol{t}^*) \cdot \mathcal{N}(\boldsymbol{\mu}_k(\boldsymbol{t}^*), \bar{\boldsymbol{\Sigma}}_k), \tag{4.18}$$

where

$$
\begin{aligned}
h_k(\boldsymbol{t}^*) &= \frac{\pi_k \cdot \mathcal{N}(\boldsymbol{t}^*|\boldsymbol{\mu}_{t,k}, \boldsymbol{\Sigma}_{tt,k})}{\sum_{i=1}^{K} \pi_i \cdot \mathcal{N}(\boldsymbol{t}^*|\boldsymbol{\mu}_{t,i}, \boldsymbol{\Sigma}_{tt,i})}, \\
\bar{\boldsymbol{\mu}}_k(\boldsymbol{t}^*) &= \boldsymbol{\mu}_{\bar{x}_h,k} + \boldsymbol{\Sigma}_{\bar{x}_h t,k}\boldsymbol{\Sigma}_{tt,k}^{-1}(\boldsymbol{t}^* - \boldsymbol{\mu}_{t,k}), \\
\bar{\boldsymbol{\Sigma}}_k &= \boldsymbol{\Sigma}_{\bar{x}_h\bar{x}_h,k} - \boldsymbol{\Sigma}_{\bar{x}_h t,k}\boldsymbol{\Sigma}_{tt,k}^{-1}\boldsymbol{\Sigma}_{t\bar{x}_h,k}.
\end{aligned}
$$

Eq. (4.18) can be approximated to

$$
\mathcal{P}(\boldsymbol{x_h^*}|\boldsymbol{t}^*) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}), \tag{4.19}
$$

where

$$
\begin{aligned}
\hat{\boldsymbol{\mu}} &= \sum\nolimits_{k=1}^{K} h_k(\boldsymbol{t}^*)\bar{\boldsymbol{\mu}}_k(\boldsymbol{t}^*), \\
\hat{\boldsymbol{\Sigma}} &= \sum\nolimits_{k=1}^{K} h_k(\boldsymbol{t}^*)(\bar{\boldsymbol{\mu}}_k(\boldsymbol{t}^*)\bar{\boldsymbol{\mu}}_k^T(\boldsymbol{t}^*) + \bar{\boldsymbol{\Sigma}}_k) - \hat{\boldsymbol{\mu}}\hat{\boldsymbol{\mu}}^T.
\end{aligned}
$$

$\hat{\boldsymbol{\mu}}$ is the learned trajectory from multiple demonstrations. $\hat{\boldsymbol{\Sigma}}$ is the variance of the trajectory.

GMM and GMR can effectively learn probabilistic features for multiple demonstrations, including temporal input and multidimensional input situations. However, it is difficult for GMM and GMR to apply their learned trajectories to situations different from the illustrative environment. In order to enhance the adaptability and generalization ability of GMM and GMR, we used DMP to model the trajectories learned by GMM and GMR and combined the environment information to change the trajectories in Chapter 5.

4.4 Experimental Results

This section shows the result of applying previous theories to our experiments, including position measurement with depth camera based detection, mean filter result of a single demonstration, DTW result of two demonstrations, and GMM and GMR result of multiple demonstrations. The pick-and-place task was chosen to validate the effectiveness of the proposed framework. A bottle pick-and-place teleoperation experiment is designed to verify the ability of the manipulator and qb-Softhand to imitate human hand behavior. The video of the experiment can be seen in https://youtu.be/qGP230LvMZ8. In multiple demonstrations, the human demonstration involves picking up a sponge from the workbench with a 40° yaw, moving it over a bottle, and placing the sponge back on the workbench. Six demonstrations were conducted, resulting in six distinct trajectories that encompass position in Cartesian space, Euler angles, and gripping distance. An example of the demonstrations can be seen in https://www.youtube.com/watch?v=nTPQDUEkxKA.

4.4.1 3D Coordinate Measurement

This part of the experiment is dedicated to validating of the accuracy associated with the 3D coordinate generated by MediaPipe and Eq. (4.9). Test points were set up at different heights and in different quadrants to ensure that the method could be tested in all areas. The measured and calculated coordinates are shown in the Table 4.1.

Table 4.1: Measurement of position

Point	Measured (cm)	Calculated (cm)	Absolute Error (cm)
1	(2.0, 8.0, 9.0)	(2.4, 8.0, 7.5)	(0.4, 0.0, 1.5)
2	(-5.0, 0.0, 9.0)	(-6.3, 0.7, 8.1)	(1.3, 0.7, 0.9)
3	(-6.0, -9.0, 34.5)	(-7.5, -8.1, 34.2)	(1.5, 0.9, 0.3)
4	(-19.0, 7.0, 26.5)	(-20.5, 7.4, 26.8)	(1.5, 0.4, 0.3)
5	(20.0, 10.0, 26.5)	(21.3, 10.1, 27.3)	(1.3, 0.1, 0.8)
6	(11.0, -10.0, 12.5)	(10.6, -8.6, 13.9)	(0.4, 1.4, 1.4)
7	(-14.0, -8.0, 12.5)	(-15.5, -6.8, 12.0)	(1.5, 1.2, 0.5)
8	(27.0, -14.0, 34.5)	(28.6, -12.2, 36.1)	(1.6, 1.8, 1.6)
Mean			(1.2, 0.8, 0.9)

As shown in Table 4.1, the maximum error observed along each axis remains confined within the threshold of 2 cm. Some errors may be due to the measurement and the small shaking of the hand during the demonstration.

4.4.2 Mean Filter Result

In light of the inherent noise present in the data collected from the human hand, a pre-processing step is employed. Specifically, we undertake data pre-processing through the application of a mean filter. Fig. 4.11 shows the comparison between raw and filtered Euler angles. The solid line represents the unprocessed data and the dashed line represents the filtered data. As we can see from Fig. 4.11, the unprocessed data is very noisy, and without removing these chattering, a typical controller will have a hard time following such signals and will be harmful to the robot's hardware. The processed signal is not only smooth but also preserves the features of the raw data. If the window size k is set too large, then the processed signal will lose its characteristics, so we have to set the window size k to an appropriate value. In this thesis, the window size k was tuned to 10.

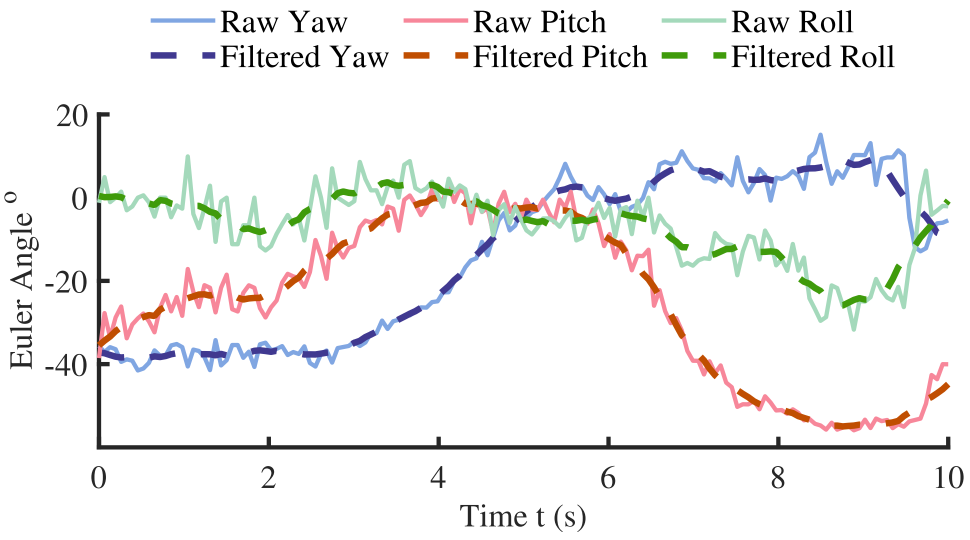


Figure 4.11: Yaw, pitch, and roll with mean filter

4.4.3 Teleoperation Experiment

To validate the proposed framework's ability of teleoperation, a real-time bottle pick-and-place experiment is designed to verify the ability of the manipulator and qb-Softhand to imitate human hand behavior. In the experiment, human operated hand motion in the camera view. Fig. 4.12 (A) shows the initial positions of the human hand and the end-effector. Fig. 4.12 (B) shows the rotation of the end-effector. Fig. 4.12 (C) shows the qb-SoftHand grasping the object. Fig. 4.12 (D) shows the translational movement of the end-effector. Fig. 4.12 (E) shows the qb-SoftHand opening and putting the object down. Fig. 4.12 (F) shows the manipulator finished the task and returned to the initial position with counter rotation.

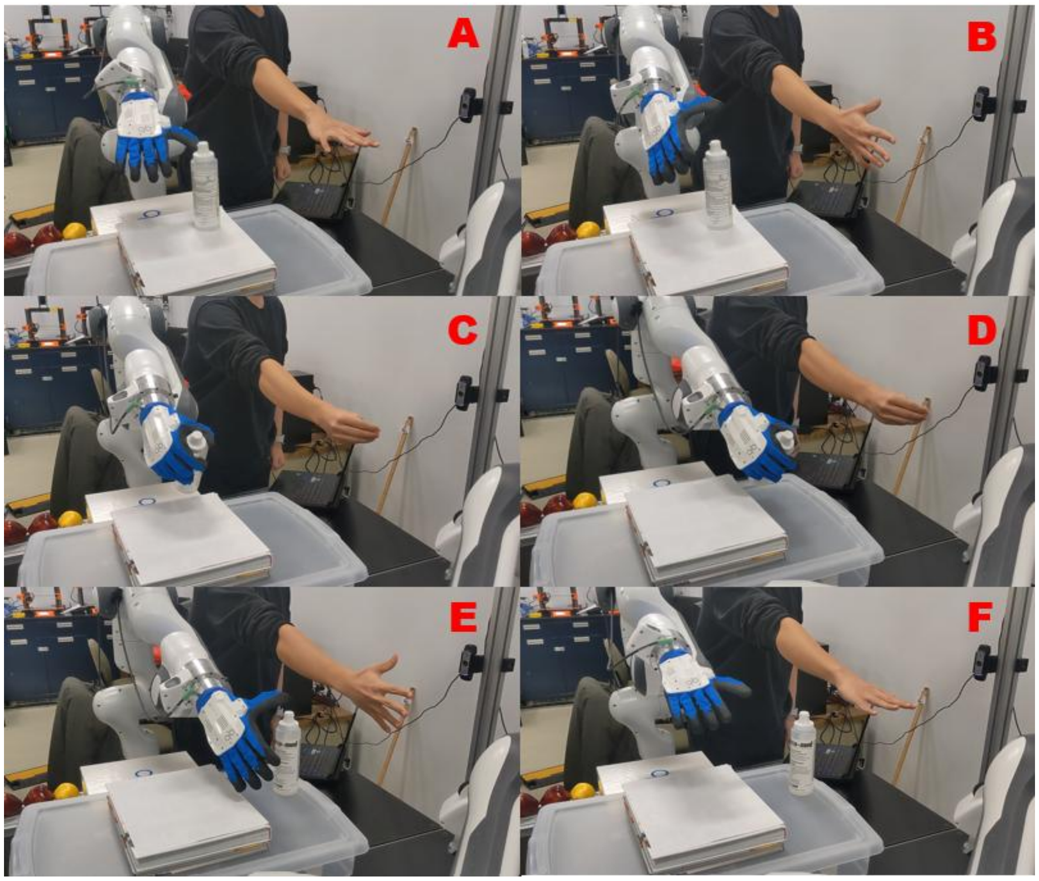


Figure 4.12: Teleoperation experiment

4.4.4 DTW Result

We apply the DTW algorithm introduced in the previous section to the processing of multiple demonstration data. We take the motion trajectory in the z-axis direction as an example, and Figure 4.13 shows the correspondence between two trajectories in different demonstrations. The dotted lines connect the corresponding points in two time series. From the result, we can see that although the two trajectories have different data lengths, the motion trends at different moments are well corresponded. Then we apply the DTW algorithm to all demonstrations. As shown in Fig. 4.14, we have six demonstrations with different lengths. By picking series 5 as the reference series, we can get the regularized series in Fig. 4.15. From Fig. 4.15, we can see that series lengths are regularized to the same with reference. Moreover, they keep the features of the value. If one point corresponds to multiple points, we set the average value as the regularized value.

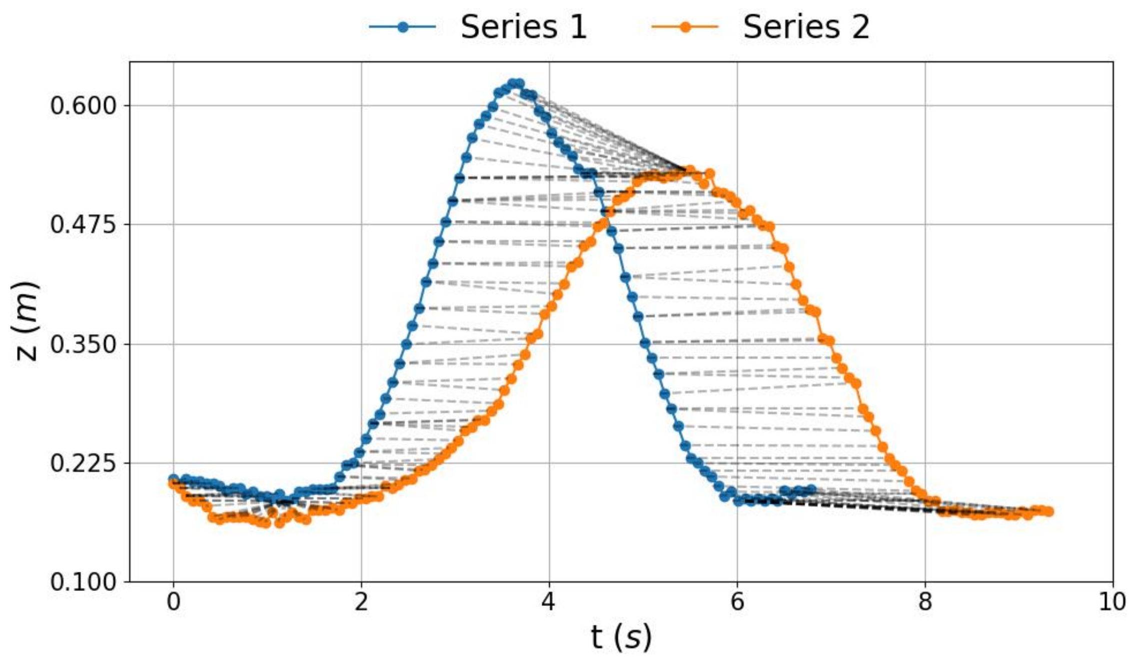


Figure 4.13: DTW result of two series. The dotted lines connect the corresponding points in two time series

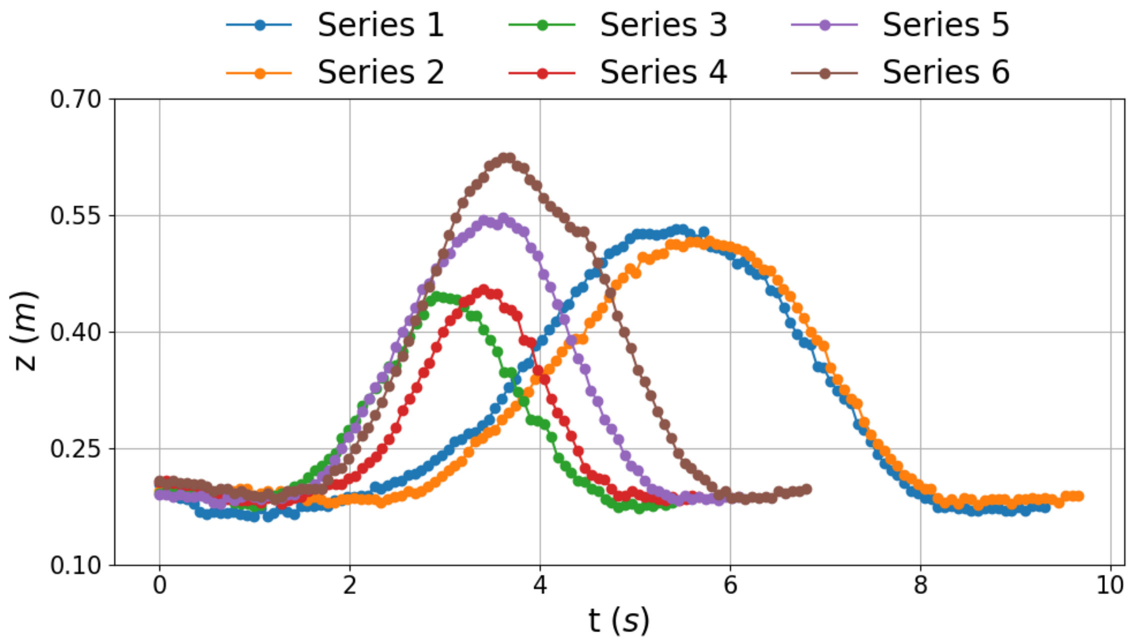


Figure 4.14: Multiple series before DTW

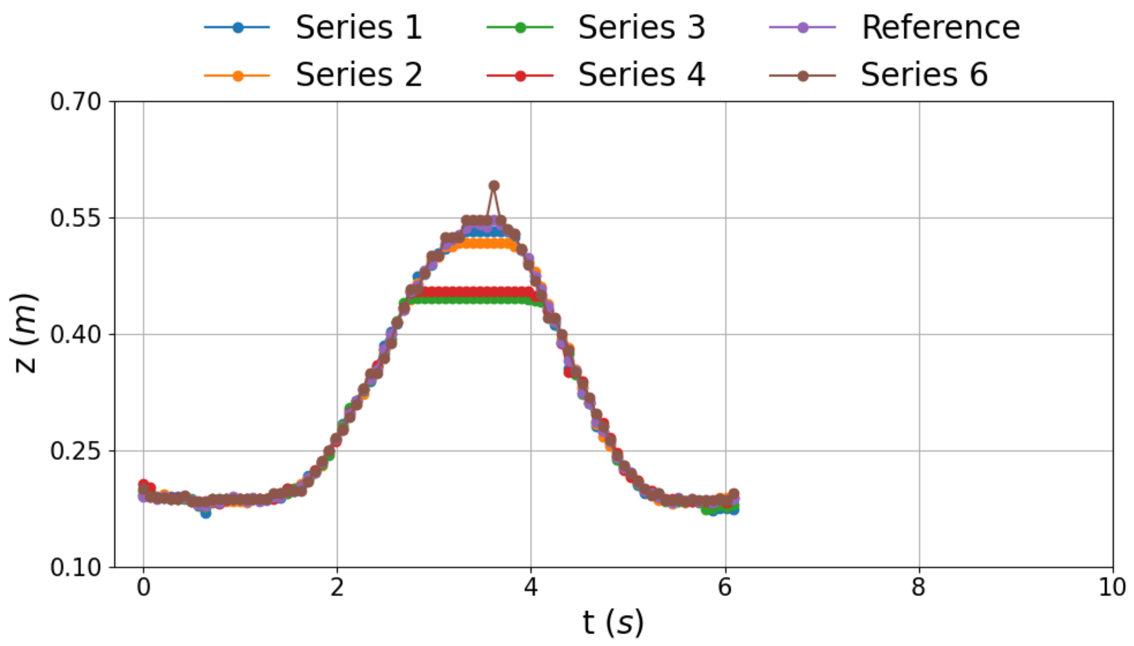


Figure 4.15: Multiple series after DTW by choosing series 5 as the reference series

4.4.5 GMM and GMR Result

Fig. 4.16 and Fig. 4.17 show the result of GMM and GMR, respectively. GMM clusters all points in a dataset in K clusters, which are represented by green ellipses. Here we set $K = 6$. With the iterations of the EM algorithm, the parameters of each Gaussian model can be calculated. After calculating those parameters, GMR will get the expectation of all clusters, which is the blue line in Fig. 4.17. Meanwhile, the variance of each Gaussian model is also calculated, which is the shadow part in Fig. 4.17.

The results obtained through the multiple demonstrations with DTW, GMM, and GMR are illustrated in Fig. 4.18. Dash lines represent the six regularized demonstration datasets and solid lines represent the learned trajectory $\hat{\boldsymbol{\mu}}$. Shadow regions denote variance $\hat{\boldsymbol{\Sigma}}$. The results demonstrate that the synthesized trajectory accurately encapsulates all demonstration trajectories while mitigating the influence of noise. These results validate the capability of the vision-based LfD framework to robustly produce a single trajectory from multiple demonstrations without the need for burdensome sensors or direct physical interactions with the manipulator.

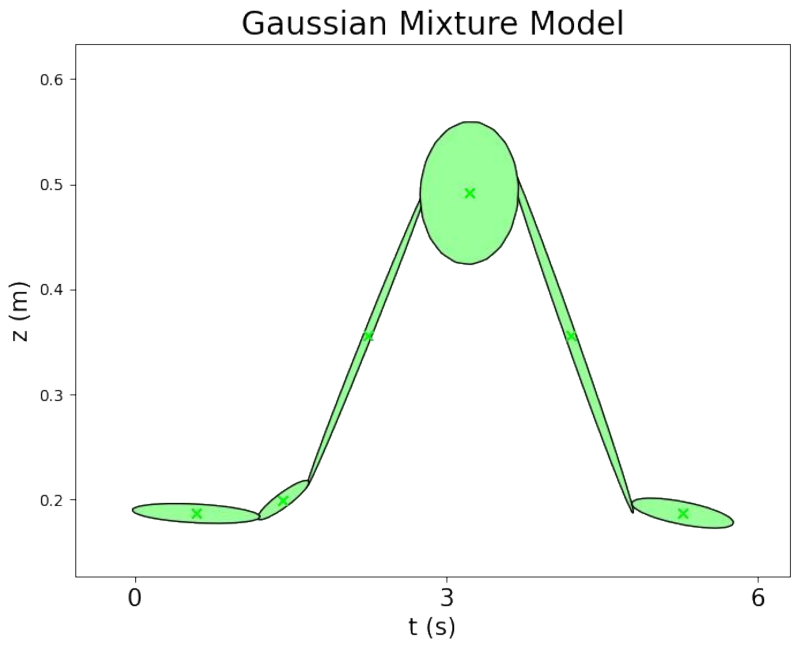


Figure 4.16: Gaussian mixture model result

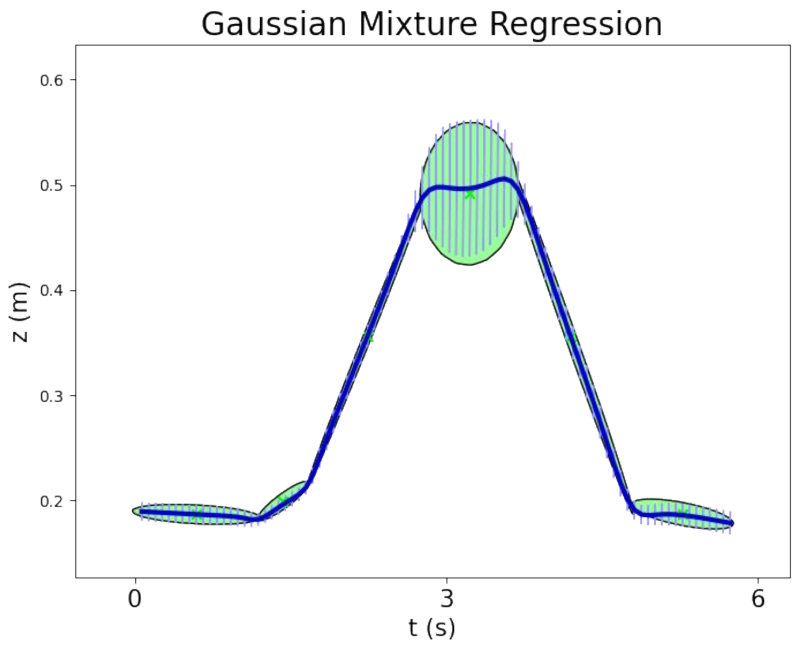


Figure 4.17: Gaussian mixture regression result

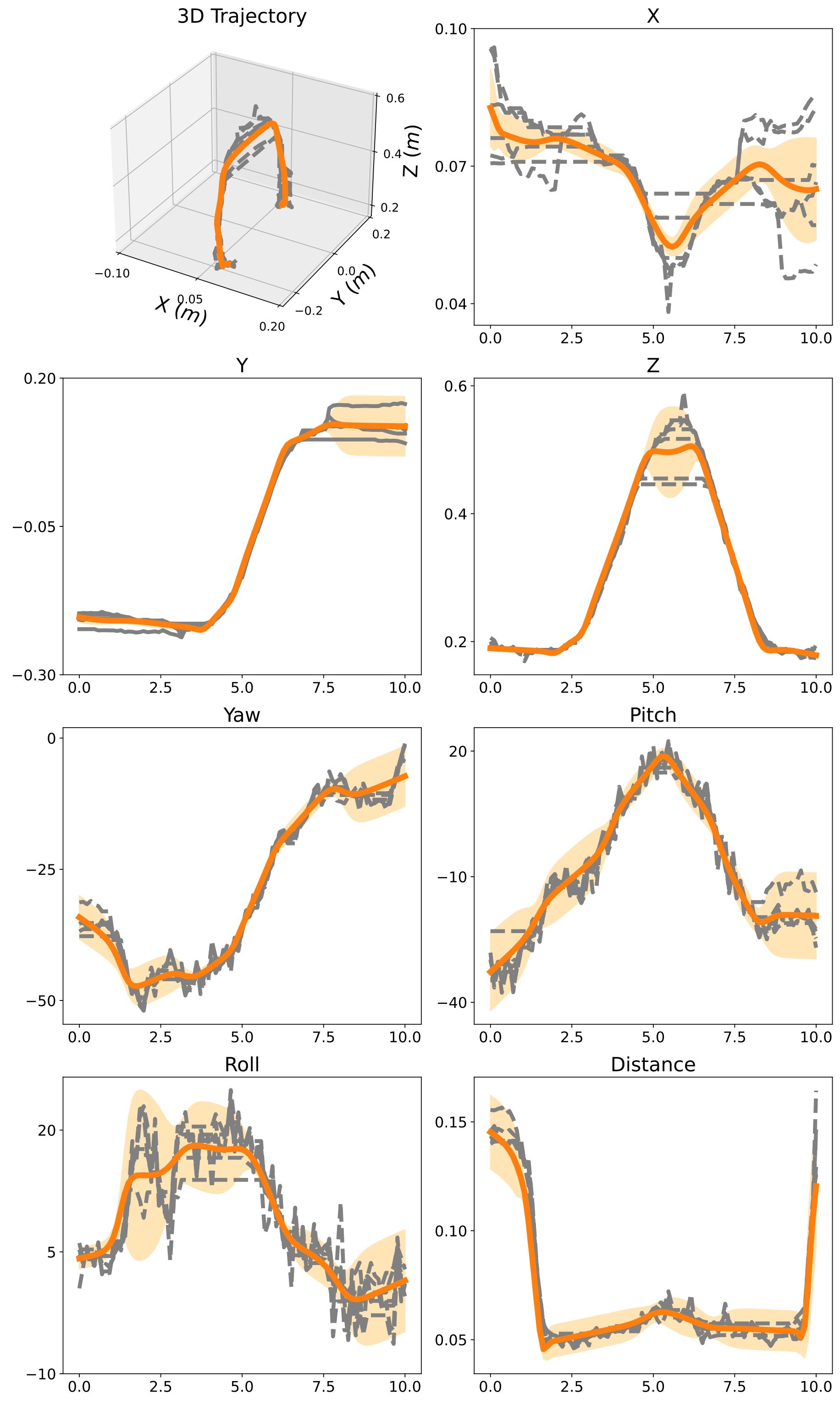


Figure 4.18: Final results of multiple demonstrations processing

Chapter 5

Path Planning and Controller Design

This chapter introduces the dynamic movement primitives (DMP) as the path planning method, Lyapunov stability, and four controller methods. With the variance from multiple demonstrations, a variance-avoidance DMP is developed. Three experiments are designed to validate the result of the proposed DMP and the tracking performance of controllers.

5.1 Dynamic Movement Primitives

Upon acquiring the human demonstration dataset, it is significant to represent this data in the form of mathematical equations. The advantage of transforming the dataset from discrete points to equations lies in the enhanced flexibility of mathematical formulations. By encapsulating the human demonstrations within equations, we have the ability to manipulate and adapt these equations to more tasks. This approach not only facilitates a deeper understanding of the underlying dynamics but also enables the seamless transfer of learned behaviors to diverse contexts, thus elevating efficiency in robotic applications. Dynamic movement primitives serve as a methodological approach for synthesizing and regulating movements based on human demonstrations [50]. This section shows the original dynamic movement primitives and methods developed through it.

5.1.1 Original DMP Method

In [22], it is proposed that complex actions can be decomposed into a set of primitive actions that are executed sequentially or in parallel. The DMP framework is introduced as the mathematical formalization of these primitive actions. Each motion primitive is characterized as a nonlinear system, with dynamic properties influenced by a guided trajectory. This design allows the primitives to be reused and adapted across various settings.

At its core, the system model of DMP combines a PD controller with the inclusion of the control term f, which is notably a nonlinear function. This design enables the system not only to converge to the goal point but also allows the motion process to emulate the original trajectory. The dynamic system is represented as

$$\begin{cases} \delta\dot{\hat{\boldsymbol{v}}}_h = K(\boldsymbol{g}_h - \hat{\boldsymbol{x}}_h) - D\hat{\boldsymbol{v}}_h + (\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0})\boldsymbol{f_h}, \\ \delta\dot{\hat{\boldsymbol{x}}}_h = \hat{\boldsymbol{v}}_h, \end{cases} \tag{5.1}$$

where $\hat{\boldsymbol{x}}_h$ and $\hat{\boldsymbol{v}}_h$ is the position and velocity of the DMP trajectory, $\hat{\boldsymbol{x}}_{h_0}$ and $\boldsymbol{g}_h$ are the start and goal points of the trajectory, δ is the time duration, $K > 0$ and $D > 0$ are the stiffness and damping parameters, respectively, and $\boldsymbol{f_h}$ is the force coupling term. To generate $\boldsymbol{f_h}$, it is imperative to first acquire $\boldsymbol{f_{h_{target}}}$, which can be represented by the demonstration trajectory as

$$\boldsymbol{f_{h_{target}}} = \frac{\delta^2\ddot{\boldsymbol{x}}_h^* - K(\boldsymbol{g}_h - \boldsymbol{x}_h^*) + D\delta\dot{\boldsymbol{x}}_h^*}{\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0}}, \tag{5.2}$$

where $\boldsymbol{x}_h^*$, $\dot{\boldsymbol{x}}_h^*$, $\ddot{\boldsymbol{x}}_h^*$ are the position, velocity, and acceleration of the processed demonstration trajectory.

The work in [22] used Gaussian functions as the basis functions to represent the nth element of $\boldsymbol{f_h} = \begin{bmatrix} f_{h_1} & f_{h_2} & \cdots & f_{h_7} \end{bmatrix}^T$. Note that for the remainder of this section, the element subscripts for the Gaussian function parameters are omitted for clarity. Assume that each element, f_{h_n}, has its own set of parameters. The basis functions are:

$$f_h = \frac{\sum_{i=1}^N \omega_i \Psi_i(s)}{\sum_{i=1}^N \Psi_i(s)} s, \tag{5.3}$$

where $\dot{s} = -\alpha_s s, \Psi_i(s) = \exp(-h_i(s - c_i)^2)$, and $s > 0$ starts at one and gradually tends toward zero, thereby ensuring that $\boldsymbol{f_h}$ approaches zero when $\hat{\boldsymbol{x}}_h$ converges to $\boldsymbol{g}_h$. α_s is a constant value, Ψ_i is the Gaussian function, where c is the center and h is the width. Each Gaussian function is endowed with a respective weight ω, and the objective is to find such a set of weights that minimizes the error between $\boldsymbol{f_h}$ and $\boldsymbol{f_{h_{target}}}$. Locally weighted regression is utilized to obtain ω_i as:

$$\omega_i = \frac{s_x^T \Gamma_i f_{h_{target}}}{s_x^T \Gamma_i s_x}, \tag{5.4}$$

where

$$s_x = \begin{pmatrix} s_1(g_h - \hat{x}_{h_0}) \\ s_2(g_h - \hat{x}_{h_0}) \\ \cdots \\ s_n(g_h - \hat{x}_{h_0}) \end{pmatrix}, \quad \Gamma_i = \begin{pmatrix} \Psi_i(1) & & 0 \\ & \ddots & \\ 0 & & \Psi_i(n) \end{pmatrix},$$

and n is the number of sampling points.

5.1.2 Modified DMP Method

In Eq. (5.1), a potential issue arises with the term $(\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0})\boldsymbol{f_h}$ when the starting point of the demonstration closely approximates the target position. In such cases, as $(\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0})$ approaches zero, the term $\boldsymbol{f_h}$ tends towards nullity. Additionally, the opposite signs of $\boldsymbol{g}_h$ and $\hat{\boldsymbol{x}}_{h_0}$ lead to a mirroring effect in the trajectory shape. To address this, a modified DMP is proposed in [50] wherein the system separates $\boldsymbol{f_h}$ and $\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0}$ so that $\boldsymbol{f_h}$ remains unaffected by the initial and goal poses:

$$\begin{cases} \delta\dot{\hat{\boldsymbol{v}}}_h = K(\boldsymbol{g}_h - \hat{\boldsymbol{x}}_h) - D\hat{\boldsymbol{v}}_h - (\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0})\hat{\boldsymbol{x}}_h + \boldsymbol{f_h}, \\ \delta\dot{\hat{\boldsymbol{x}}}_h = \hat{\boldsymbol{v}}_h. \end{cases} \tag{5.5}$$

The DMP system represented by Eq. (5.5) can successfully regress the demonstration and the system can adapt to different initial and endpoints. However, it has practical limitations because environmental conditions may change during the trajectory execution. For example, for a pick-and-place task, the height of the obstacles may vary. Therefore, it becomes necessary to introduce a coupling term into the DMP system to enable adaptation to the dynamic environment. In [24], Gams et al. introduced a virtual force coupling term $\boldsymbol{F}(t)$ in the system to facilitate the adaptation of the DMP trajectory to the dynamic environment.

$$\begin{cases} \delta\dot{\hat{\boldsymbol{v}}}_h = K(\boldsymbol{g}_h - \hat{\boldsymbol{x}}_h) - D\hat{\boldsymbol{v}}_h - (\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0})\hat{\boldsymbol{x}}_h + \boldsymbol{f_h} + c\dot{\boldsymbol{F}}(t), \\ \delta\dot{\hat{\boldsymbol{x}}}_h = \hat{\boldsymbol{v}}_h + c\boldsymbol{F}(t), \\ \boldsymbol{F}(t) = k_1\boldsymbol{h}(t), \end{cases} \tag{5.6}$$

where $\boldsymbol{h}$ is the obstacle dimensions, c and k_1 are tuneable parameters.

5.1.3 Variance-Avoidance DMP

As mentioned in Chapter 4, the variance of different demonstrations can be obtained. Variance serves as an indicator of the confidence level in a demonstration. When the variance is smaller, it implies higher confidence in the demonstration trajectory. This observation inspires a method of tuning the DMP system based on the variance.

As shown in Eq. (5.1), the DMP system comprises a PD controller and a control term $\boldsymbol{f_h}$. The control term $\boldsymbol{f_h}$ is learned from demonstration to change the trajectory and the PD controller ensures the convergence of the endpoint. In the DMP system, when the variance of a segment is small, it means that the trajectory of this segment is more repeatable among multiple demonstrations, so the system should trust the demonstration data of this segment more, thus reducing the effect of virtual force term $\boldsymbol{F}(t)$. On the contrary, when the variance of a segment is larger, it indicates that the trajectory of this segment is more flexible in multiple demonstrations. Therefore the system can give more influence to the virtual force term $\boldsymbol{F}(t)$. This tuning approach introduces the variance of the multiple demonstrations part into the path planning. It can allow the confidence level of the various demonstrations to influence the stiffness of the path planning, which showcases the superiority of the framework proposed in this thesis. The proposed DMP system with variance and obstacle dimensions is proposed as

$$\begin{cases} \delta\dot{\hat{\boldsymbol{v}}}_h = K(\boldsymbol{g}_h - \hat{\boldsymbol{x}}_h) - D\hat{\boldsymbol{v}}_h - (\boldsymbol{g}_h - \hat{\boldsymbol{x}}_{h_0})\hat{\boldsymbol{x}}_h + \boldsymbol{f_h} + c\dot{\boldsymbol{F}}(t), \\ \delta\dot{\hat{\boldsymbol{x}}}_h = \hat{\boldsymbol{v}}_h + ck_2\sqrt[3]{\left|\hat{\boldsymbol{\Sigma}}\right|}\boldsymbol{F}(t), \\ \boldsymbol{F}(t) = k_1\boldsymbol{h}(t), \end{cases} \tag{5.7}$$

where $\hat{\boldsymbol{\Sigma}}$ is the variance from the GMR output and k_2 is a tuneable parameter.

5.2 Controller

This section presents Lyapunov stability and four controllers, which are popularly utilized in manipulators and human-robot interaction.

5.2.1 Lyapunov Stability

Lyapunov stability is an important theory in the analysis of nonlinear systems, providing a way to determine the stability of a system without solving differential equations [51]. An equilibrium is asymptotically stable if, for any small perturbation of the initial state near the equilibrium point, the state of the system converges to the equilibrium point over time.

The basis of Lyapunov theory can be described as follows. Consider a nonlinear system $\dot{x} = f(x)$, the system is considered asymptotically stable if there exists a Lyapunov function $V(x)$ that satisfies these three conditions [52]:

1) The Lyapunov function is zero at the initial point: $V(x) = 0$.
2) The Lyapunov function is positive definite in the region beyond the initial point: $V(x) > 0, \forall x \neq 0$.
3) The derivative of the Lyapunov function with respect to time is negative definite in some neighborhoods near the initial point: $\dot{V}(x) < 0, \forall x \neq 0$.

Lyapunov stability theory plays an important role in the design and analysis of controllers, especially for nonlinear control systems. By using Lyapunov theory, the stability of the system can be ensured and controllers can be designed to meet specific performance requirements. The negative definite condition of the Lyapunov function is utilized to derive a control law such that the derivative of the Lyapunov function $\dot{V}(x)$ is negative for all $x \neq 0$. Specifically, the control input u is designed such that $\dot{V}(x,u) = \frac{\partial V}{\partial X} f(x,u)$ where $f(x,u)$ is the system state equation.

5.2.2 PD Control

Proportional-derivative control is a classical control algorithm that is widely used in automatic control systems. The PD controller generates control signals u to stabilize the control system by taking into account the current value of the control error e and the rate of change of the error $\dot{e}$. PD controller can be represented as

$$u(t) = K_p e(t) + K_d \dot{e}(t), \tag{5.8}$$

where K_p and K_d are proportional gain and derivative gain. The PD controller is a simple and effective control strategy to regulate the output of the control system through both proportional and differential aspects, which is suitable for application scenarios with high response speed requirements and low requirements for steady-state error. Correctly selecting and adjusting the proportional and differential gains is the key to realizing a good control effect.

In this thesis, we design the PD controller in Cartesian space for the rotational motion of the manipulator end-effector by Eq. (5.9) as

$$\boldsymbol{\tau}_{PD} = J_H(\boldsymbol{q})^T(-K_H\boldsymbol{e}_H - D_H\dot{\boldsymbol{e}}_H), \tag{5.9}$$

where K_H and D_H are the position and velocity gains, respectively and $\boldsymbol{e}_H = (\eta_1\eta_2 + \boldsymbol{\epsilon}_1^T\boldsymbol{\epsilon}_2)(-\eta_1\boldsymbol{\epsilon_2} + \eta_2\boldsymbol{\epsilon_1} - S(\boldsymbol{\epsilon_1})\boldsymbol{\epsilon_2}) \in \mathbb{R}^3$, $\dot{\boldsymbol{e}}_H = \boldsymbol{\omega} - \boldsymbol{\omega}_d \in \mathbb{R}^3$ [53], and

$$S(\boldsymbol{\epsilon}) = \begin{bmatrix} 0 & -q_z & q_y \\ q_z & 0 & -q_x \\ -q_y & q_x & 0 \end{bmatrix}.$$

5.2.3 Impedance Control

Impedance control is a type of soft control in which a joint or end-effector of a manipulator can move along a new desired trajectory when an external force is applied to the joint or end-effector, instead of moving along the original desired trajectory. Compared to traditional position control, impedance control avoids damage to the robot due to flexible stiffness. In conventional position control, when the manipulator is subjected to an external force $\boldsymbol{F}_{ext}$, the error in the angle of the joints increases, resulting in failure to obtain the correct position of the end-effector and possible damage to the hardware. The principle of impedance control is to consider a joint or end-effector as a mass-spring-damper system, as illustrated in Fig. 5.1, whose equation is shown as Eq. (5.10),

$$M_d\ddot{\boldsymbol{x}} + D_d\dot{\boldsymbol{x}} + K_d\boldsymbol{x} = \boldsymbol{F}_{ext}, \tag{5.10}$$

where M_d is the mass value, K_d is the spring stiffness, D_d is the damper damping, and $\boldsymbol{F}_{ext}$ is the external force.

Mass-spring-damper system function in joint space is shown in Eq. (5.11),

$$M_d(\ddot{\boldsymbol{q}}_d - \ddot{\boldsymbol{q}}) + D_d(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}) + K_d(\boldsymbol{q_d} - \boldsymbol{q}) = \boldsymbol{\tau}_{ext}. \tag{5.11}$$

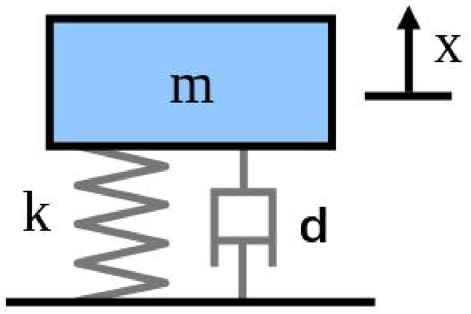


Figure 5.1: Mass spring damper system

The dynamic equation of manipulator in joint space with external torque $\boldsymbol{\tau}_{ext}$ is as

$$\mathcal{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} + \mathcal{G}(\boldsymbol{q}) = \boldsymbol{\tau} - \boldsymbol{\tau}_{ext}. \tag{5.12}$$

According to Eq. (5.11) and Eq. (5.12), we obtain the equation of $\boldsymbol{\tau}_{js}$ in joint space:

$$\begin{aligned}\boldsymbol{\tau}_{js} =& \mathcal{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d + \mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} + \mathcal{G}(\boldsymbol{q}) \\ &+ \mathcal{M}(\boldsymbol{q})M_d^{-1}(D_d(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}) + K_d(\boldsymbol{q_d} - \boldsymbol{q})) + (\mathcal{I} - \mathcal{M}(\boldsymbol{q})M_d^{-1})\boldsymbol{\tau}_{ext}.\end{aligned} \tag{5.13}$$

The transformation of Eq. (5.13) from the joint space to the Cartesian space is

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \ddot{\boldsymbol{x}} \\ \boldsymbol{F}_{ext} \end{bmatrix} = \begin{bmatrix} J(\boldsymbol{q})\dot{\boldsymbol{q}} \\ J(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} \\ J(\boldsymbol{q})^{-T}\boldsymbol{\tau}_{ext} \end{bmatrix}. \tag{5.14}$$

According to Eq. (5.13) and Eq. (5.14), we obtain the equation of $\boldsymbol{\tau}_{cs}$ in Cartesian space:

$$\begin{aligned}\boldsymbol{\tau}_{cs} =& \mathcal{M}(\boldsymbol{q})J^{-1}(\boldsymbol{q})M_d^{-1}(M_d\ddot{\boldsymbol{x}}_d + D_d(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}) + K_d(\boldsymbol{x}_d - \boldsymbol{x}) - M_d\dot{J}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}}) \\ &+ (J^T(\boldsymbol{q}) - \mathcal{M}(\boldsymbol{q})J^{-1}(\boldsymbol{q})M_d^{-1})\boldsymbol{F}_{ext} + \mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q}) + \mathcal{G}(\boldsymbol{q}).\end{aligned} \tag{5.15}$$

5.2.4 Sliding Mode Control

Sliding mode control (SMC) is a robust control strategy especially suitable for nonlinear systems with large uncertainties and perturbations [54]. For instance, when the manipulator picks up an unknown object, the PD and impedance controllers cannot compensate for gravity, which can lead to errors, especially in the case of heavy objects.

A sliding mode controller achieves the desired control objective by designing a sliding surface along which the system state slides. The core of sliding mode control is to define a sliding surface s, which is usually some linear or nonlinear combination of state variables. During the sliding motion, the dynamic behavior of the system is determined by the equation for the surface. The goal of sliding mode control is to drive the system state to the desired equilibrium point quickly and with high robustness at the same time. Consider a nonlinear system

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = f(\boldsymbol{x}) + g(\boldsymbol{x}) + b(\boldsymbol{x})u, \end{cases} \tag{5.16}$$

where $\boldsymbol{x} = \begin{bmatrix} x_1, x_2 \end{bmatrix}^T$ is the state vector, $f(\boldsymbol{x})$ and $b(\boldsymbol{x})$ are nonlinear functions, $g(\boldsymbol{x})$ is the external bounder disturbance, and u is the control input. The sliding surface is defined as

$$\begin{cases} s = \lambda x_1 + x_2, \\ \dot{s} = \lambda \dot{x}_1 + \dot{x}_2, \end{cases} \tag{5.17}$$

where $\lambda > 0$ is a design constant.

When $s = 0$, the sliding surface can be represented in a coordinate system where x_1 is the x-axis and x_2 is the y-axis, as illustrated in Fig. 5.2. When the point (x_1, x_2) is on the sliding surface in the second quadrant, $x_1 < 0$ and $x_2 > 0$, resulting in an increase in x_1. Conversely, when (x_1, x_2) is on the sliding surface in the fourth quadrant, $x_1 > 0$ and $x_2 < 0$, causing x_1 to decrease. Therefore, when (x_1, x_2) lies on the sliding surface, it will converge to the origin, ensuring that both the error in joint angle and velocity become zero. This guarantees the convergence on the sliding mode surface.

Since the point (x_1, x_2) is guaranteed to converge to the origin on the sliding surface, it is significant that the point (x_1, x_2) reaches the sliding surface, which makes $s = 0$, so we need the reaching law. The objective of reaching law is to make the system approach the sliding surface.

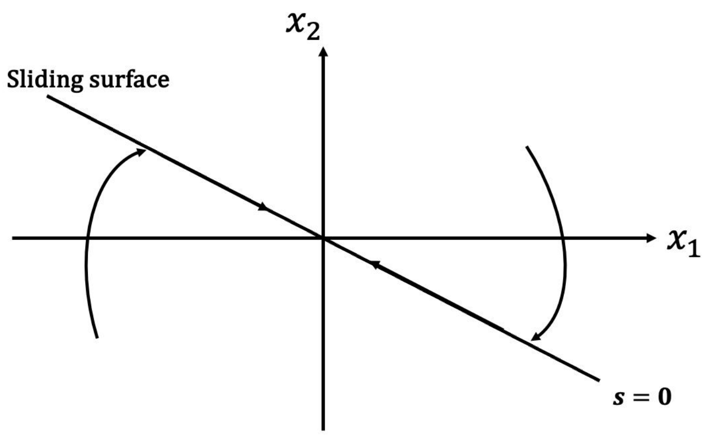


Figure 5.2: Sliding coordinate system

According to Lyapunov function:

$$\begin{cases} V(s) = \frac{1}{2}s^2, \\ \dot{V}(s) = s\dot{s}. \end{cases} \tag{5.18}$$

To make $s = 0$, we have to find a reaching law $\dot{s}$ such that $\dot{V}(s) < 0$. The commonly used reaching law is $\dot{s} = -\epsilon sgn(s) - ps$, where $\epsilon > 0, p > 0$. According to the reaching law and Eq. (5.18), we can get $\dot{V}(s)$ as

$$\dot{V}(s) = -\epsilon sgn(s)s - ps^2, \tag{5.19}$$

where the switching function in this work is $sgn(s)$ which demotes the sign function, defined as

$$sgn(s) = \begin{cases} 1\ ,\ s > 0 \\ 0\ ,\ s = 0 \\ -1\ ,\ s < 0 \end{cases} \tag{5.20}$$

Because $sgn(s)$ and s are always greater or less than zero at the same time, so $\dot{V}(s) < 0$. According to Lyapunov stability theory, s will converge toward zero, so as to realize the objective of reaching the sliding surface. Lyapunov stability makes a contribution to designing the reaching law and designing the controller.

According to the reaching law and the representation of $\dot{s}$, we can get the equation of $\ddot{\boldsymbol{q}}$,

$$\ddot{\boldsymbol{q}} = c\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}}_d + \epsilon sgn(\boldsymbol{s}) + p\boldsymbol{s}. \tag{5.21}$$

So we can get the equation of $\boldsymbol{\tau}_{sm}$ for manipulator:

$$\boldsymbol{\tau}_{sm} = \mathcal{M}(\boldsymbol{q})(c\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}}_d + \epsilon sgn(\boldsymbol{s}) + p\boldsymbol{s}) + \mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q}) + \mathcal{G}(\boldsymbol{q}). \tag{5.22}$$

Note that this controller relies on the system model $\mathcal{M}$, $\mathcal{C}$, and $\mathcal{G}$ functions. In our work, the Franka Emika robot provides real-time numerical models at each sampling time of $\mathcal{M}$, $\mathcal{C}$, and $\mathcal{G}$ functions.

5.2.5 Non-Singular Terminal Sliding Mode Control

The non-singular terminal sliding mode controller is proposed in [37] to avoid the singularity that may occur in the regular SMC. The sliding surface of NTSM controller is defined as

$$\begin{cases} s = x_1 + \beta x_2^{\alpha}, \\ \dot{s} = \dot{x}_1 + \alpha\beta x_2^{\alpha-1}\dot{x}_2, \end{cases} \tag{5.23}$$

where $\beta > 0$, $\alpha_i = p/q$, $p > 0$ and $q > 0$ are adjacent odd numbers such that $1 < \alpha < 2$. The control input is then commonly designed as

$$u = -b^{-1}(\boldsymbol{x})(f(\boldsymbol{x}) + \frac{\beta}{\alpha}x_2^{2-\alpha} + (G + \eta)sgn(s)), \tag{5.24}$$

where G is the upper bound of the disturbance and η is a constant parameter. Compared with regular SMC, NTSM control input Eq. (5.24) does not contain the x_1 term in the denominator, which eliminates the risk of zero division and encountering a singularity.

In this thesis, the NTSM controller is designed in Cartesian space for the translational motion of the manipulator end-effector as

$$\boldsymbol{\tau}_{NTSM} = J_p(\boldsymbol{q})^T \bar{\mathcal{M}}_p(\boldsymbol{q})\Big(\frac{-\dot{\boldsymbol{e}}_p^{2-\alpha}}{\alpha\beta} + \ddot{\boldsymbol{p}}_d - |\bar{\mathcal{M}}_p(\boldsymbol{q})|^{-T}\lambda\tanh(k_s\boldsymbol{s}) - \kappa\tanh(k_s\boldsymbol{s})\Big),$$

where $\lambda > 0$ and $\kappa > 0$ are the controller gains, and $k_s > 0$ is a design parameter that gives a trade-off between reducing chattering and tracking performance. The continuous $tanh(s) = \frac{e^{2s}-1}{e^{2s}+1}$ function is used in place of the discontinuous $sgn()$ function to mitigate chattering and lower the control effort at the expense of a slower response near the sliding surface. The sliding surface is defined as

$$\boldsymbol{s} = \boldsymbol{e}_p + \beta\dot{\boldsymbol{e}}_p^{\alpha}, \tag{5.25}$$

where $\boldsymbol{e}_p = \boldsymbol{p} - \boldsymbol{p}_d, \dot{\boldsymbol{e}}_p = \dot{\boldsymbol{p}} - \dot{\boldsymbol{p}}_d$.

In the pick-and-place experiment to compensate for object weight, the controller is designed as a hybrid NTSM-PD controller. NTSM control is used for translational motion to achieve accurate tracking in the presence of internal friction, unmodelled dynamics, and unknown object weight. A PD controller is used for the rotational motion for simplicity and to avoid chattering. The controller is designed as

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{HYB} + \mathcal{C}(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} + \mathcal{G}(\boldsymbol{q}), \tag{5.26}$$

where $\boldsymbol{\tau}_{HYB} = \begin{bmatrix} \boldsymbol{\tau}_{NTSMC}, & \boldsymbol{\tau}_{PD} \end{bmatrix}^T$ is the hybrid NTSM-PD controller.

5.3 Experimental Results

This section contains three experiments. A pick-and-place experiment with different start and goal points is designed to validate the adaptability of modified DMP. Variance-avoidance DMP will avoid obstacles in a 3D Cartesian space. An NTSM-PD controller is utilized to compensate a 1.2kg weight object while achieving the trajectory tracking.

5.3.1 Modified DMP

The execution of the human demonstration involves picking up a sponge from the workbench with 40° yaw, moving it over a cup, and putting the sponge in a box sloped with a pitch angle of 50°. The experiment environment is shown in Fig. 5.3. The trajectory and value of the task can be seen in Fig. 5.4. Fig. 5.4 (d) and (e) show the Euler angles and gripper distance d_{ti} in the demonstration, which will be replicated by the end-effector. Then we employ modified DMP to learn the trajectory of X, Y, and Z, respectively, with three new starting points: $\begin{bmatrix} 0.37, -0.34, 0.22 \end{bmatrix}$, $\begin{bmatrix} 0.50, -0.25, 0.21 \end{bmatrix}$, $\begin{bmatrix} 0.55, -0.34, 0.28 \end{bmatrix}$, and three new end points: $\begin{bmatrix} 0.51, 0.11, 0.31 \end{bmatrix}$, $\begin{bmatrix} 0.50, 0.19, 0.30 \end{bmatrix}$, $\begin{bmatrix} 0.50, 0.28, 0.32 \end{bmatrix}$. The unit of the above points are meters. Three new trajectories are shown in Fig. 5.4(a)(b)(c)(f). New trajectories change the start and end points, but keep the shape, quaternion, and grasping motion. The result shows that modified DMP has the ability to mimic the path and adapt to new start and goal points. The video of the experiment can be seen on the ACM Lab YouTube channel: `https://www.youtube.com/watch?v=XP22mKGLvUI`.

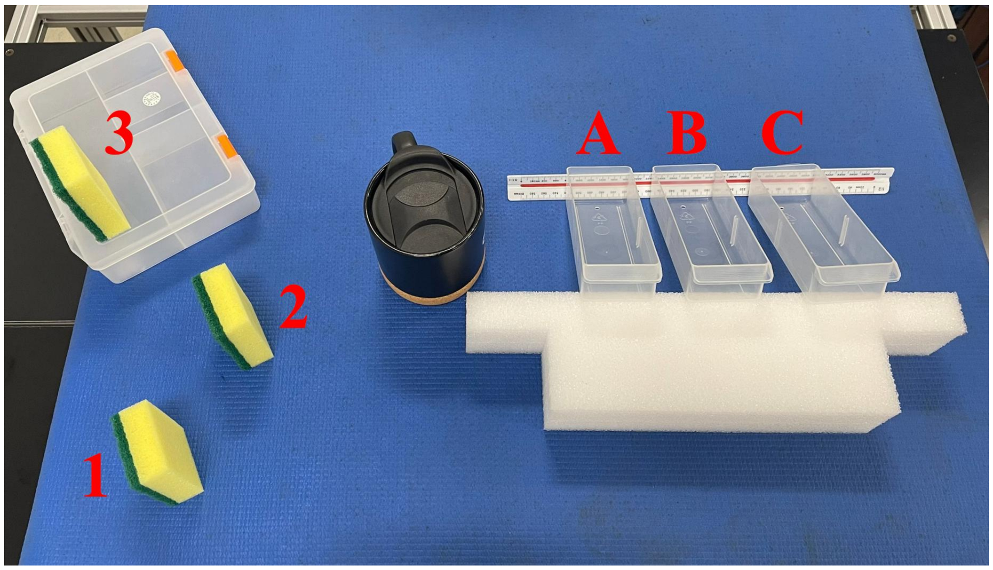
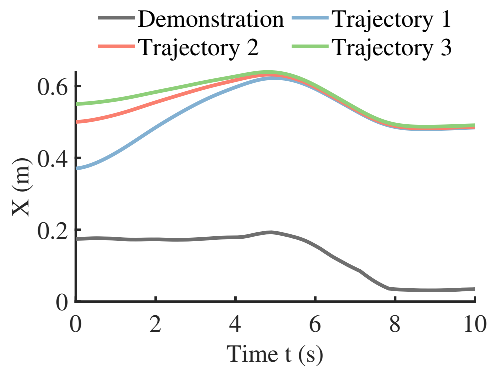


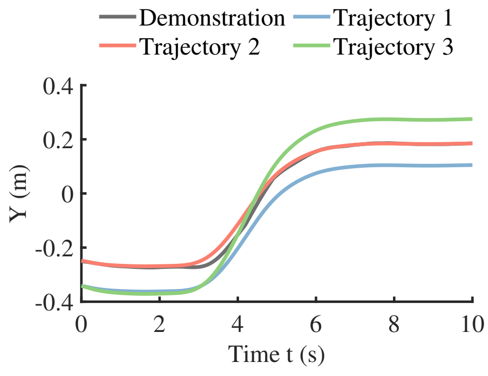
Figure 5.3: Environment of modified DMP experiment

5.3.2 Variance-Avoidance DMP

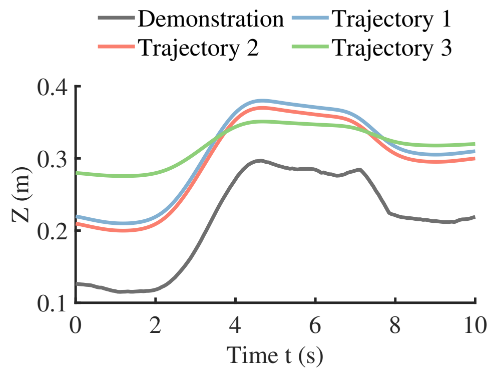
In the pick-and-place task, the demonstrations avoid the obstacle by moving in the z-axis, therefore the following results mainly focus on the z-axis. Within the DMP system, the parameters are $k_1 = 10$, $k_2 = 15$, $K = 1$, $D = 1$, and $c = 1$. Fig. 5.5 illustrates the results of the proposed variance-avoidance DMP approach. In Fig. 5.5, the obstacle z-axis dimensions are represented by the gray blocks. As the height of the obstacle varies between 0.4 m and 0.5 m, the generated trajectories in the z-axis exhibit suitable variations to avoid the different obstacles. Fig. 5.6 and Fig. 5.7 present two trajectories in which the obstacle exists at different intervals of the trajectory with varying degrees of variance. This indicates that during periods of low variance, the trajectory is less affected by obstacle avoidance, and the trajectory can closely follow the original demonstrations. Whereas during periods of high variance, the trajectory takes a more cautious and adaptive path to avoid the obstacle. These findings underscore the adaptability and precision of the variance-avoidance DMP system in navigating complex environments, validating its effectiveness in real-world applications. The proposed method also shows the novelty of the framework which contains LfD and path planning. Obstacle avoidances can also be realized in a 3D space with different obstacles in x, y, and z-axis, respectively. Fig. 5.8 illustrates the 3D scenario of avoiding an obstacle during the operation.



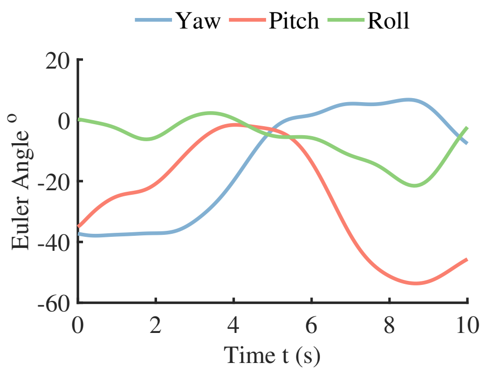
(a) X



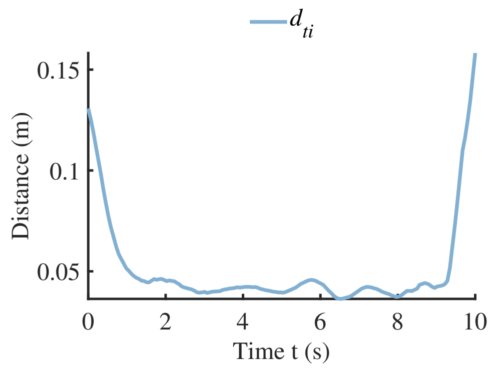
(b) Y



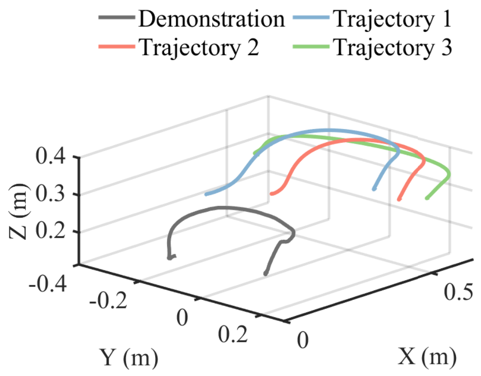
(c) Z



(d) Yaw, Pitch and Roll



(e) Distance



(f) 3D Trajectory

Figure 5.4: Human demonstration and new trajectories generated by modified DMP

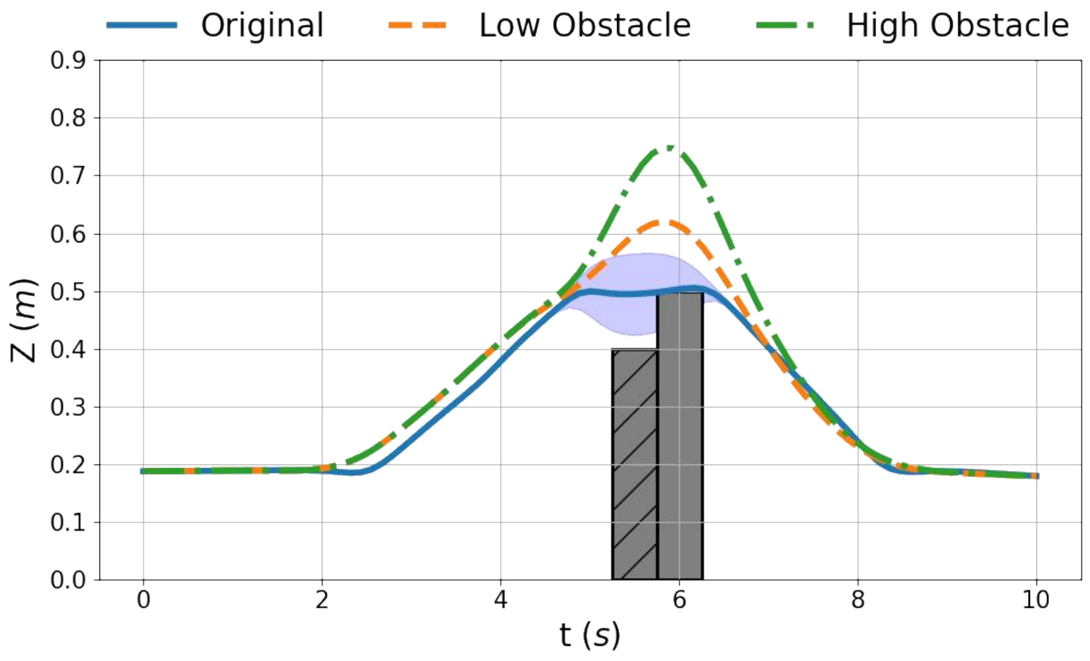


Figure 5.5: DMP trajectories exhibit varying responses to different height obstacles. The block with diagonal lines represents a low obstacle with a height of 0.4 m while the other block represents a high obstacle with a height of 0.5 m. DMP responses are more pronounced when the obstacle is higher

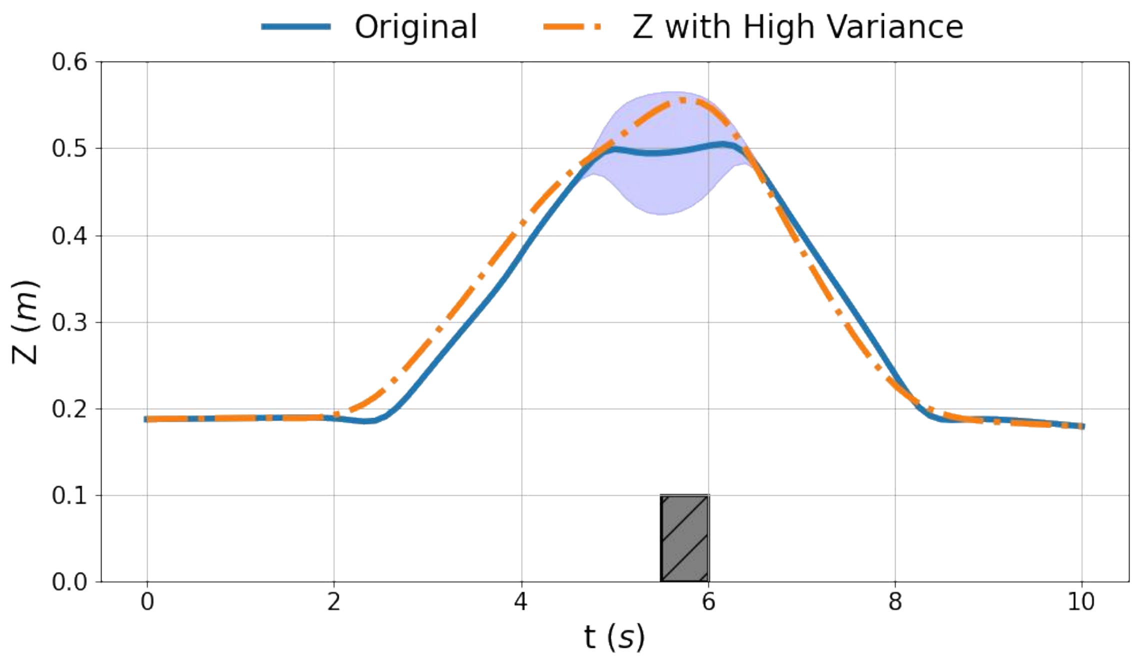


Figure 5.6: When facing obstacles of the same height, different variance parts exhibit varied responses. DMP responses are more prominent when the variance is larger. The displacement for the high variance part is 0.07m

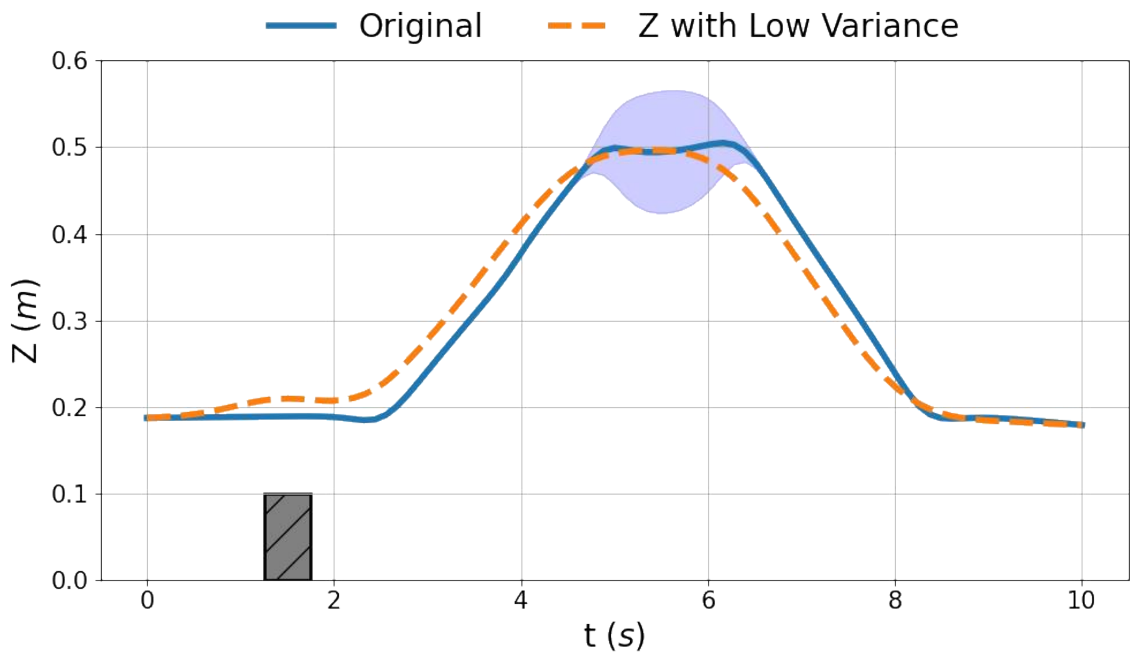
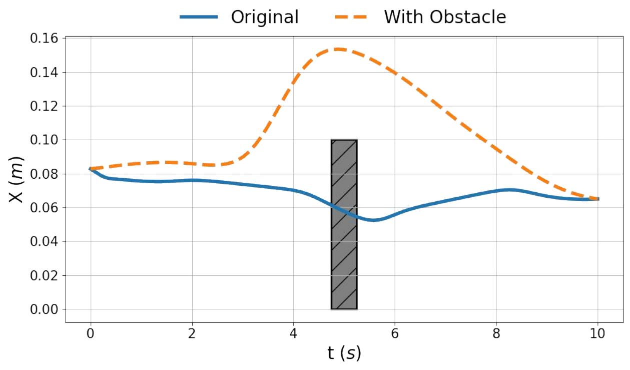
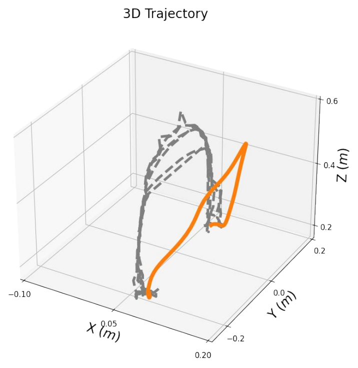


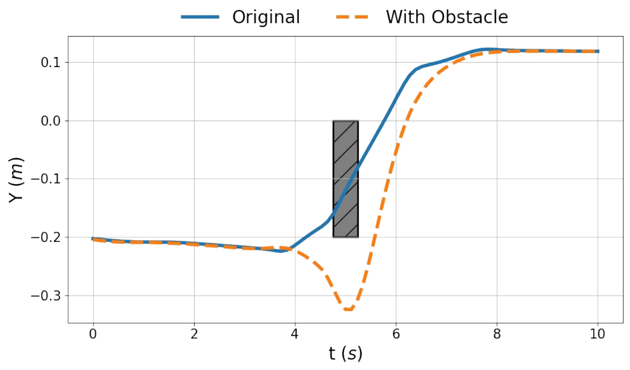
Figure 5.7: The displacement for the low variance part is 0.03m



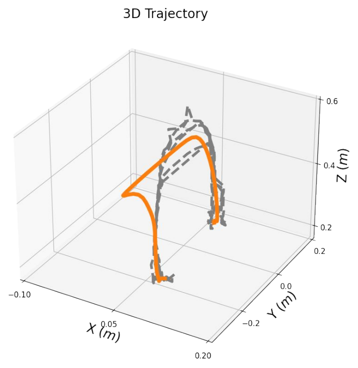
(a) Avoidance in X-axis



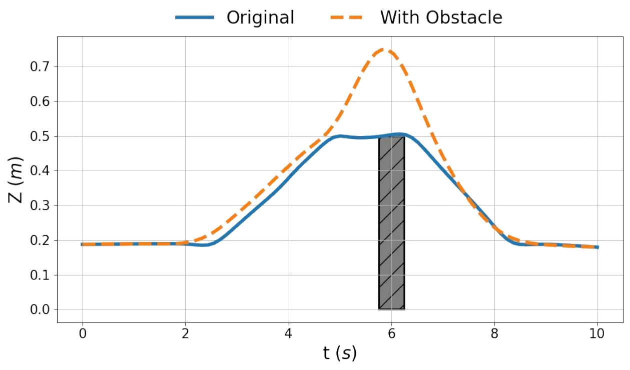
(b) 3D plot of avoidance in X-axis



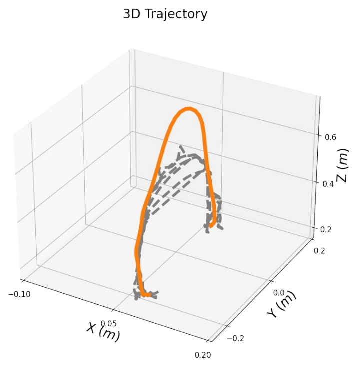
(c) Avoidance in Y-axis



(d) 3D plot of avoidance in Y-axis



(e) Avoidance in Z-axis



(f) 3D plot of avoidance in Z-axis

Figure 5.8: Variance-avoidance DMP obstacle avoidance in 3D space

5.3.3 NTSM-PD Controller

To show the advantages of using the NTSM-PD controller to track the variance-avoidance DMP trajectory, it is compared to two controllers: 1) a PD controller that is applied to both translation and rotational motion and 2) an SM controller that is applied to translational motion and applies a PD controller for the rotational motion. The proposed DMP trajectory to lift an object over an obstacle is trained and executed on the Franka Emika manipulator. Two different weight objects are set and they are unknown to the manipulator. Only the motion of the z-axis is analyzed during the motion of the object over the obstacle as this motion is highly impacted by the weight of the object.

The controller gains are tuned to minimize the tracking error and chattering. The parameters for the translational PD controller are $K_p = 600$ and $D_p = 28.3$, which are significantly higher than 'high' stiffness values outlined in [55]. The parameters for both the translational NTSM controller and translational SM controller proposed in [54] are tuned to be $\alpha = 1.66$, $\beta = 0.2$, $k_s = 10$, $\lambda = 6$, and $\kappa = 48$. These parameters are chosen to ensure that the error is minimized while preventing chattering. The rotational PD controller for the NTSM-PD controller, PD controller, and SM-PD controller are $K_H = 30$ and $D_H = 3.5$.

The tracking accuracy is evaluated by taking the Root Mean Square Error (RMSE) $\mathcal{E}$ between the desired position and the actual end-effector position for the z-axis as

$$\mathcal{E} = \sqrt{\frac{\sum_{i=1}^{N}(z - z_d)^2}{N}}, \tag{5.27}$$

where N is the number of data points that are analyzed. To show how the NTSM-PD controller can operate with the object weight better than the PD and SM-PD controller, the task is executed without the object, $\mathcal{E}_{\text{NOB}}$, and with the object, $\mathcal{E}_{\text{OB}}$. The decrease in performance due to the object is calculated as

$$\Delta_{\text{OB}} = \frac{\mathcal{E}_{\text{OB}} - \mathcal{E}_{\text{NOB}}}{\mathcal{E}_{\text{NOB}}} \times 100\%. \tag{5.28}$$

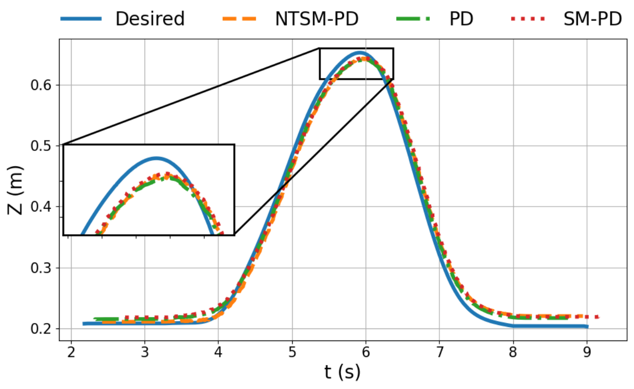
The comparative performance of the NTSM-PD, PD, and SM-PD controllers, all with and without an additional load is shown in Fig. 5.9. The three controllers perform equally well in the absence of an object, but in the presence of an object, the NTSM-PD controller's trajectory is closer to the desired one. As shown in Table 5.1,

Table 5.1: Comparison of PD, SM-PD, and NTSM-PD controllers

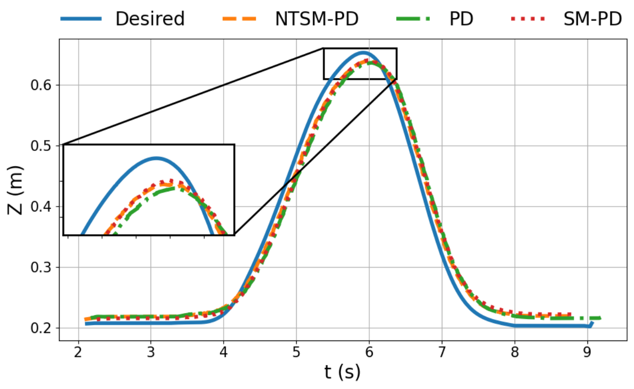
Weight (kg)	Controller	$\mathcal{E}_{\text{NOB}}$ (cm)	$\mathcal{E}_{\text{OB}}$ (cm)	Δ_{OB}
0.45	PD [55]	1.267	2.453	93.61%
	SM-PD [54]	1.626	2.245	37.45%
	NTSM-PD	**1.538**	**1.877**	**22.04%**
1.2	PD [55]	1.267	3.210	153.35%
	SM-PD [54]	1.626	2.448	50.55%
	NTSM-PD	**1.538**	**1.738**	**13.00%**

the $\mathcal{E}_{\text{OB}}$ of the NTSM-PD controller is significantly smaller than that of the other two controllers, and the magnitude of change Δ_{OB} is the smallest. When lifting the object, the PD and SM-PD controller exhibited significant reductions in tracking accuracy and when the weight increases their performance become worse. The proposed NTSM-PD controller does not show significant errors when faced with heavier objects.

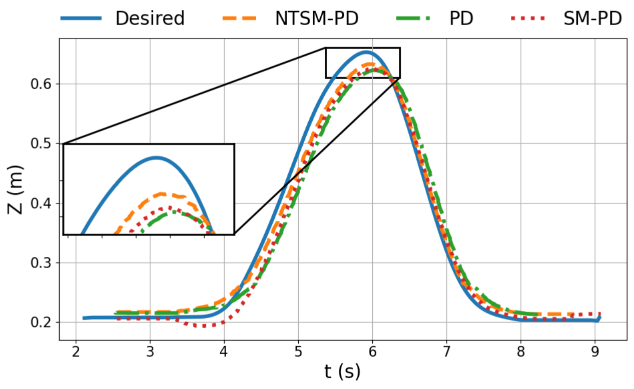
These results show the enhanced robustness of the NTSM-PD controller against the challenges posed by heavy lifting, highlighting its superior performance in contrast to the conventional PD and SM-PD controllers. In addition, although the $\mathcal{E}_{\text{OB}}$ of the PD and SM-PD controllers are also small, since the Δ_{OB} of the NTSM-PD is smaller, it is reasonable to believe that the NTSM-PD will perform significantly better than the PD and SM-PD controllers in tasks where the industrial manipulator performs a heavier payload.



(a)



(b)



(c)

Figure 5.9: Experimental results for the z-axis of the manipulator's end-effector during the lifting phase of the task (a) without object, (b) with a 0.45kg weight object and (c) with a 1.2kg weight object

Chapter 6

Conclusions and Future Work

This chapter summarizes the work in this thesis and some future research areas in force control and reinforcement learning.

6.1 Conclusions

In this thesis, a novel framework integrating multiple vision-based demonstration capture, motion planning, and non-linear robot control has been introduced for LfD for tasks in complex environments. The proposed motion capture and planning method provides contactless, robust, and dexterous 3D translational and rotational trajectories. Through the integration of MediaPipe and depth camera, the framework enables the precise calculation of the 3D coordinates of the human hand, with an error margin of less than 2 cm on each axis. The processed motion accurately extracts the common features from multiple demonstrations and eliminates noises. To augment the adaptiveness of the DMP system, a novel variance-avoidance DMP is developed to adapt to dynamic environments, considering different obstacle dimensions and demonstration trajectory variances. An NTSM-PD controller is implemented to compensate for external disturbances. The root mean square error of the proposed controller with a 1.2 kg object is 1.738 cm and the decrease in performance due to the object is 13.00%. The above experimental results showcase superior tracking performance compared to the conventional PD controller and SM-PD controller. This framework demonstrates promising robotic capabilities for learning and adapting to diverse scenarios in daily applications.

6.2 Future Work

Although this thesis captures position trajectories in the demonstrations and compensates for external forces in the execution, hybrid control, which is to control both position and force, is not implemented in this thesis. Reinforcement learning has become very popular in recent years, and it has demonstrated a strong ability for dynamic position control, but not much research has been done on force control. Therefore, the author is looking forward to making a breakthrough in the field of hybrid force and position control by combining traditional controllers with reinforcement learning in future research and applying this control framework to multiple manipulator scenarios and human-robot interaction scenarios.

Bibliography

[1] Robotnik. Complete guide to manipulator robots: benefits and applications. Website, 2024. https://robotnik.eu/complete-guide-to-manipulator-robots-benefits-and-applications/.

[2] Brennan Whitfield. 15 agricultural robots and farm robots you should know. Website, 2024. https://builtin.com/robotics/farming-agricultural-robots.

[3] Astro. The potential of mobile manipulators in health care settings. Website, 2023. https://hiastro.com/the-potential-of-mobile-manipulators-in-health-care-settings/.

[4] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7:358–386, 2005.

[5] Xinjian Deng, Jianhua Liu, Honghui Gong, Hao Gong, and Jiayu Huang. A human-robot collaboration method using a pose estimation network for robot learning of assembly manipulation trajectories from demonstration videos. *IEEE Transactions on Industrial Informatics*, 19(5):7160–7168, 2023.

[6] Ji Woong Kim, Changyan He, Muller Urias, Peter Gehlbach, Gregory D. Hager, Iulian Iordachita, and Marin Kobilarov. Autonomously navigating a surgical tool inside the eye by learning from demonstration. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7351–7357, 2020.

[7] Mina Alibeigi, Sadegh Rabiee, and Majid Nili Ahmadabadi. Inverse kinematics based human mimicking system using skeletal tracking technology. *Journal of Intelligent & Robotic Systems*, 85:27–45, 2017.

[8] Hang Su, Nima Enayati, Luca Vantadori, Andrea Spinoglio, Giancarlo Ferrigno, and Elena De Momi. Online human-like redundancy optimization for tele-operated anthropomorphic manipulators. *International Journal of Advanced Robotic Systems*, 15(6):1729881418814695, 2018.

[9] Matthew Chignoli, Donghyun Kim, Elijah Stanger-Jones, and Sangbae Kim. The mit humanoid robot: Design, motion planning, and control for acrobatic behaviors. In *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pages 1–8, 2021.

[10] Jie Li, Xiaofeng Liu, Zhelong Wang, Hongyu Zhao, Tingting Zhang, Sen Qiu, Xu Zhou, Huili Cai, Rongrong Ni, and Angelo Cangelosi. Real-time human motion capture based on wearable inertial sensor networks. *IEEE Internet of Things Journal*, 9(11):8953–8966, 2022.

[11] Boyang Ti, Yongsheng Gao, Qiang Li, and Jie Zhao. Human intention understanding from multiple demonstrations and behavior generalization in dynamic movement primitives framework. *IEEE Access*, 7:36186–36194, 2019.

[12] Xueyan Xing, Etienne Burdet, Weiyong Si, Chenguang Yang, and Yanan Li. Impedance learning for human-guided robots in contact with unknown environments. *IEEE Transactions on Robotics*, 39(5):3705–3721, 2023.

[13] Qiguang Chen, Lucas Wan, and Ya-Jun Pan. Object recognition and localization for pick-and-place task using difference-based dynamic movement primitives. *IFAC-PapersOnLine*, 56(2):10004–10009, 2023. 22nd IFAC World Congress.

[14] Caixia Cai, Nikhil Somani, and Alois Knoll. Orthogonal image features for visual servoing of a 6-dof manipulator with uncalibrated stereo cameras. *IEEE Transactions on Robotics*, 32(2):452–461, 2016.

[15] Jiahao Lin, Hai Zhu, and Javier Alonso-Mora. Robust vision-based obstacle avoidance for micro aerial vehicles in dynamic environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2682–2688, 2020.

[16] Linda van der Spaa, Michael Gienger, Tamas Bates, and Jens Kober. Predicting and optimizing ergonomics in physical human-robot cooperation tasks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1799–1805, 2020.

[17] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[18] Nuo Chen, Xinxing Chen, Chuheng Chen, Yuquan Leng, and Chenglong Fu. Research on the human-following method, fall gesture recognition, and protection method for the walking-aid cane robot. In *2022 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, pages 286–291, 2023.

[19] Chen Cai and Steven Liu. A probabilistic dynamic movement primitives framework on human hand motion prediction for an object transfer scenario. *IFAC-PapersOnLine*, 56(2):8327–8332, 2023. 22nd IFAC World Congress.

[20] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Yong, Juhyun Lee, et al. Mediapipe: A framework for perceiving and processing reality. In *Third workshop on computer vision for AR/VR at IEEE computer vision and pattern recognition (CVPR)*, volume 2019, 2019.

[21] Vaishnav Chunduru, Mrinalkanti Roy, Dasari Romit N. S, and Rajeevlochana G. Chittawadigi. Hand tracking in 3d space using mediapipe and pnp method for intuitive control of virtual globe. In *2021 IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC)*, pages 1–6, 2021.

[22] A.J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 2, pages 1398–1403 vol.2, 2002.

[23] Liang Han, Han Yuan, Wenfu Xu, and Yunzhi Huang. Modified dynamic movement primitives: Robot trajectory planning and force control under curved surface constraints. *IEEE Transactions on Cybernetics*, 53(7):4245–4258, 2023.

[24] Andrej Gams, Bojan Nemec, Auke Jan Ijspeert, and Aleš Ude. Coupling movement primitives: Interaction with the environment and bimanual tasks. *IEEE Transactions on Robotics*, 30(4):816–830, 2014.

[25] Franziska Meier and Stefan Schaal. A probabilistic representation for dynamic movement primitives. *arXiv preprint arXiv:1612.05932*, 2016.

[26] Nuo Chen and Ya-Jun Pan. Vision-based dexterous motion planning by dynamic movement primitives with human hand demonstration. In *33rd IEEE International Symposium on Industrial Electronics (ISIE)*, 2024.

[27] Aliakbar Akbari, Mohammed Diab, and Jan Rosell. Contingent task and motion planning under uncertainty for human-robot interactions. *Applied Sciences (Switzerland)*, 10, 3 2020.

[28] Takumi Sakamoto, Kensuke Harada, and Weiwei Wan. Real-time planning robotic palletizing tasks using reusable roadmaps. *International Journal of Computational Intelligence Systems*, 6:240–245, 2020.

[29] Todor Davchev, Michael Burke, and Subramanian Ramamoorthy. Learning structured representations of spatial and interactive dynamics for trajectory prediction in crowded scenes. *IEEE Robotics and Automation Letters*, 6(2):707–714, 2021.

[30] Dapeng Zhao and Jean Oh. Noticing motion patterns: A temporal cnn with a novel convolution operator for human trajectory prediction. *IEEE Robotics and Automation Letters*, 6(2):628–634, 2021.

[31] Tan Zhang, Kefang Zhang, Jiatao Lin, Wing-Yue Geoffrey Louie, and Hui Huang. Sim2real learning of obstacle avoidance for robotic manipulators in uncertain environments. *IEEE Robotics and Automation Letters*, 7(1):65–72, 2022.

[32] Hugo Nascimento, Martin Mujica, and Mourad Benoussaad. Collision avoidance interaction between human and a hidden robot based on kinect and robot data fusion. *IEEE Robotics and Automation Letters*, 6(1):88–94, 2021.

[33] Jen-Hao Chen and Kai-Tai Song. Collision-free motion planning for human-robot collaborative safety under cartesian constraint. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4348–4354, 2018.

[34] Shahab Heshmati-Alamdari, Charalampos P. Bechlioulis, George C. Karras, and Kostas J. Kyriakopoulos. Cooperative impedance control for multiple underwater vehicle manipulator systems under lean communication. *IEEE Journal of Oceanic Engineering*, 46(2):447–465, 2021.

[35] Xinghuo Yu, Yong Feng, and Zhihong Man. Terminal sliding mode control: An overview. *IEEE Open Journal of the Industrial Electronics Society*, 2:36–52, 2021.

[36] Ke Shao, Jinchuan Zheng, Chao Yang, Feng Xu, Xueqian Wang, and Xiu Li. Chattering-free adaptive sliding-mode control of nonlinear systems with unknown disturbances. *Computers Electrical Engineering*, 96:107538, 2021.

[37] Yong Feng, Xinghuo Yu, and Zhihong Man. Non-singular terminal sliding mode control of rigid manipulators. *Automatica*, 38(12):2159–2167, 2002.

[38] Henghua Shen and Ya-Jun Pan. Nonlinear state estimation and online neighbor selection for multimanipulator systems. *IEEE/ASME Transactions on Mechatronics*, 27(6):4373–4383, 2022.

[39] Lucas Wan, Ya-Jun Pan, and Qiguang Chen. Admittance-based non-singular terminal sliding mode control of multiple cooperative manipulators. In *2023 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 111–116, 2023.

[40] Teng Long, En Li, Yunqing Hu, Lei Yang, Junfeng Fan, Zize Liang, and Rui Guo. A vibration control method for hybrid-structured flexible manipulator based on sliding mode control and reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 32(2):841–852, 2021.

[41] Fotios Dimeas and Nikos Aspragathos. Reinforcement learning of variable admittance control for human-robot co-manipulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1011–1016, 2015.

[42] Bianca Sangiovanni, Gian Paolo Incremona, Antonella Ferrara, and Marco Piastra. Deep reinforcement learning based self-configuring integral sliding mode control scheme for robot manipulators. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 5969–5974, 2018.

[43] Cheng Gong, Wai-Kit Sou, and Chi-Seng Lam. Reinforcement learning based sliding mode control for a hybrid-statcom. *IEEE Transactions on Power Electronics*, 38(6):6795–6800, 2023.

[44] Claudio Gaz, Marco Cognetti, Alexander Oliva, Paolo Robuffo Giordano, and Alessandro De Luca. Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization. *IEEE Robotics and Automation Letters*, 4(4):4147–4154, 2019.

[45] Nuo Chen, Lucas Wan, Qiguang Chen, and Ya-Jun Pan. Real time vision-based human hand motion tracking and grasping for a robotic manipulator with soft hand. In *Proceedings of 2023 CSME International Congress of Canadian Mechanical Engineering*, 2023.

[46] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[47] Qiang Cheng, Wei Zhang, Hongshuai Liu, Ying Zhang, and Lina Hao. Research on the path planning algorithm of a manipulator based on gmm/gmr-mprm. *Applied Sciences*, 11(16), 2021.

[48] Iordanis Kerenidis, Alessandro Luongo, and Anupam Prakash. Quantum expectation-maximization for gaussian mixture models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 5187–5197, 2020.

[49] Zhao Man, Li Fengming, Quan Wei, Li Yibin, and Song Rui. Robot bolt skill learning based on gmm-gmr. In *International Conference on Intelligent Robotics and Applications*, pages 235–245. Springer, 2021.

[50] Dae-Hyung Park, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*, pages 91–98, 2008.

[51] Hassan K Khalil. Lyapunov stability. *Control systems, robotics and automation*, 12:115, 2009.

[52] Hassan K Khalil. *Control of nonlinear systems.* Prentice Hall, New York, NY, 2002.

[53] Ricardo Campa, Karla Camarillo, and Lina Arias. Kinematic modeling and control of robot manipulators via unit quaternions: Application to a spherical wrist. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 6474–6479, 2006.

[54] Jinkun Liu, Xinhua Wang, Jinkun Liu, and Xinhua Wang. *Advanced sliding mode control*. Springer, 2011.

[55] Mathew Jose Pollayil, Franco Angelini, Guiyang Xin, Michael Mistry, Sethu Vijayakumar, Antonio Bicchi, and Manolo Garabini. Choosing stiffness and damping for optimal impedance planning. *IEEE Transactions on Robotics*, 39(2):1281–1300, 2022.

Appendix A

Publication List

[1]. **N. Chen**, L. Wan, and Y.-J. Pan, “Robust and Adaptive Dexterous Manipulation with Vision-Based Learning from Multiple Demonstrations”, *IEEE Transactions on Industrial Electronics*, 2024. Under Review.

[2]. **N. Chen** and Y.-J. Pan, “Vision-Based Dexterous Motion Planning by Dynamic Movement Primitives with Human Hand Demonstration”, in *Proceedings of the 33rd IEEE International Symposium on Industrial Electronics (ISIE)*, Ulsan, South Korea, 2024.

[3]. **N. Chen**, L. Wan, Q.G. Chen and Y.-J. Pan, “Real Time Vision-based Human Hand Motion Tracking and Grasping for a Robotic Manipulator with Soft Hand”, in *Proceedings of the 2023 CSME International Congress of Canadian Mechanical Engineering*, vol. 6, 2023.

[4]. A. -F. Fontaine, D. Zhu, **N. Chen** and Y. -J. Pan, “Nonlinear Model Predictive Control for Autonomous Underwater Vehicle Trajectory Tracking,” *2023 IEEE 2nd Industrial Electronics Society Annual On-Line Conference (ONCON)*, SC, USA, 2023, pp. 1-6.