Intelligent Robotic Systems for Learning and Reproducing Therapeutic Interventions in Rehabilitation Medicine

by

Jason Fong

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Biomedical Engineering

Department of Electrical and Computer Engineering

University of Alberta

© Jason Fong, 2019

Abstract

As the world's population increases and ages, the demand for rehabilitation medicine services is on the rise. Recently, robot-assisted rehabilitation has become an appealing, powerful and economical means with which to address this demand while lowering the burden on practicing therapists and the healthcare system. However, current robotic rehabilitation systems take therapists out of the loop and replace them with an algorithm that determines what interactions to provide to a patient participating in therapy. The therapist can intervene but only through a computer interface and not on a hands-on basis. This is problematic because years of therapist's education and experience are not being used. Also, substantial changes to the way the robot interacts with the patient requires computer programming know-how that does not usually exist in clinics. This thesis focuses on the incorporation of Learning from Demonstration (LfD) algorithms into rehabilitation robotics for the purposes of efficiently time-sharing therapists across multiple patients, and to enable therapists with minimal computer programming knowledge to easily and intuitively reprogram the behaviors of rehabilitation robots on a patient-specific, task-specific, and session-specific basis. A secondary aim of introducing a generalized, manipulator-based robotic system to various areas of rehabilitation medicine is also explored. In current clinical practice, a multitude of equipment is often required to facilitate both rehabilitation and assessment, which is inefficient in both space and cost. Current robotic rehabilitation systems also come in many different designs that are specific to at most a few select

therapy tasks, thereby failing to address either of these inefficiencies. The work in this thesis shows that one single system can be applied to both upper and lower limb post-stroke therapy, as well as occupational rehabilitation of injured workers.

This thesis presents the use of LfD algorithms for learning position-based and impedance-based behaviors of a therapist when guiding a patient through a post-stroke robot-assisted therapy task. The proposed system is evaluated for upper limb tasks meant to resemble Activities of Daily Living (ADLs), and for lower limb therapy in the form of treadmill-based gait training. In each case, the therapeutic interventions associated with each exercise are learnt from demonstrations provided by a therapist and are then reproduced by the rehabilitation robotic system. The fidelity of these semi-autonomous reproductions is then evaluated. In addition, a comparison between the use of a telerobot (i.e., a pair of teleoperated robots) versus a single robot as the therapeutic medium is performed.

Lastly, the intelligent robotics technologies developed throughout the thesis is applied to the field of occupational rehabilitation, which has seen relatively little robotic integration. The purpose of this application is to introduce the benefits of robotics specifically for Functional Capacity Evaluation (FCE) of workers injured in the course of their duties. The system is integrated with a virtual environment that simulates a workplace task, which is presented to users through both haptic feedback and an augmented-reality display for greater immersion. The realism of the system is evaluated by comparing user biomechanics when interacting with the robotic system, and when performing a real-world equivalent version of the therapy task.

Evaluations for all of the included works are performed with able-bodied participants. The term "therapist" refers to participants with no formal training in rehabilitation medicine who are asked to train the robots. The term

"patient" refers either to able-bodied participants (who may have a disability simulated through worn equipment), or to mechanical devices such as springs that are meant to simulate a patient with disability.

Robotic rehabilitation systems such as the one presented in this thesis represent a large step forward in the rehabilitation medicine field. By enabling robots to learn from human experts and focusing on making designs generalizable, the potential to provide cost-efficient, intensive, and immersive therapy with minimal burden on healthcare providers while keeping specialists in the loop becomes closer to being realized.

Preface

This thesis is an original work by Jason Fong. The research project, of which this thesis is a part, received research ethics approval from the University of Alberta Research Ethics Board, Project Name "Measuring the mechanical impedance of the upper limb using a rehabilitation robot", No. MS7 Pro00033955. April 27, 2017. Initially used as the ethics approval by Matthew Dyck in 2013, this ethics approval was revised to encompass the work done in this thesis.

Chapters 4 and 5 are my original work. Chapter 4 has been published as Jason Fong, Hossein Rouhani and Mahdi Tavakoli, "A Therapist-taught Robotic System for Assistance During Gait Therapy Targeting Foot Drop," in *IEEE Robotics and Automation Letters (Special Issue on Intelligent Human-Robot Interaction for Rehabilitation and Physical Assistance)*, 2019. For this work, I was responsible for the development of the system, experimental validation, data collection and analysis, and manuscript composition. Hossein Rouhani and Mahdi Tavakoli acted as supervisory authors and were involved with concept formation and manuscript composition. Chapter 5 was presented as Jason Fong and Mahdi Tavakoli, "Kinesthetic Teaching of a Therapist's Behavior to a Rehabilitation Robot," at *The International Symposium on Medical Robotics*, Atlanta, GA, 2018. For this paper, I was responsible for the same roles. Mahdi Tavakoli acted again as a supervisory author and was involved with the same roles.

Chapter 3 has been submitted for review as Carlos Martinez, Jason Fong and Mahdi Tavakoli, "Semi-autonomous Robot-assisted Cooperative Therapy Exercises for a Therapist's Interaction with a Patient" to *IEEE Robotics and Automation Letters* and *IEEE International Conference on Automation Science and Engineering*, Vancouver, Canada, 2019 through a dual submission procedure. Chapter 6 has been accepted for presentation as Jason Fong, Carlos Manuel Martinez, Mahdi Tavakoli, "Ways to Learn a Therapist's Patient-specific Intervention: Robotics- vs Telerobotics-mediated Hands-on Teaching," at *IEEE International Conference on Robotics and Automation*, Montreal, Canada, 2019. Both works are collaborative efforts with Carlos Martinez. In both cases, Carlos and I were responsible for the development of the system, experimental validation, data collection and analysis, and manuscript composition, all of which we performed with equal contribution. Mahdi Tavakoli acted as a supervisory author and was involved with concept formation and

manuscript composition.

Chapter 7 has been submitted for review as Jason Fong, Renz Ocampo, Douglas P. Gross and Mahdi Tavakoli, "A Robot with an Augmented-Reality Display for Functional Capacity Evaluation and Rehabilitation of Injured Workers" to *IEEE-RAS-EMBS International Conference on Rehabilitation Robotics*, Toronto, Canada, 2019. The work is a collaborative effort with Renz Ocampo. I was responsible for the robot admittance control and biomechanics tracking with a motion tracker camera, as well as the majority of the data analysis. Renz was responsible for implementing the augmented-reality system and programming the virtual task environment. Renz and I were responsible for the experimental validation, data collection, and manuscript composition, all of which we performed with equal contribution. Douglas Gross and Mahdi Tavakoli acted as supervisory authors and were involved with concept formation and manuscript composition.

All described devices and algorithms that are the work of others are acknowledged where appropriate.

Dedication

In memory of my mother.
You taught me how to be strong, courageous, how to persevere in hard times, and, most importantly, to be kind and compassionate. You will always be my role model in life.

To my father:
Thank you for always sharing your wisdom, lending an ear, and for having infinite patience. Words cannot express how grateful I am for all the kindness and patience you have provided me throughout my life and all of its ups and downs. Last but not least, thank you for not only providing me with the opportunity to follow my dreams, but for also encouraging me to do so.

To my brothers:
Thank you both for sticking with me through thick and thin, and showing me how to navigate life every step of the way. You guys are the best of the best of the best, with honors.

Acknowledgements

I would like to first and foremost thank my supervisor Dr. Tavakoli. The immeasurable amount of support you provided allowed me to do the very best I could throughout my studies. The culture you set for all of the students in your laboratory is what makes researching under your guidance so rewarding, whether it be when working through problems together when someone is stuck or when sharing a laugh over a casual conversation. I am deeply inspired by your passion for robotics and your ability to encourage students to challenge themselves. Thank you for providing me with the opportunity to study under you these past few years.

I would also like to thank all of my family for their tremendous support. Your encouragement, empathy, advice, and love are the driving forces behind my studies and achievements. Thank you for believing in me and the goals I want to achieve.

I would like to thank all of my friends that I met along the way. Each one of you has impacted my life and made me the person that I am today. I would especially like to thank my colleagues in the Telerobotic and Biorobotic Systems lab for all of their support, advice, and comradery.

Special thanks to Dr. Tavakoli, Dr. Adams, Dr. Cheng, and Dr. Ardakani for serving on my examination committee.

Last but not least, I would like to recognize the monetary support and in-kind contributions provided by the Canada Foundation for Innovation (CFI), the Alberta Innovation and Advanced Education Ministry, the Natural Sciences and Engineering Research Council of Canada (NSERC), and Quanser Inc.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AAN	Assistance-as-needed
ADL	Activities of daily living
AR	Augmented reality
BWSTT	Body weight supported treadmill training
CIMT	Constraint-induced movement therapy
CPG	Central pattern generator
DOF	Degree of freedom
E2S	Elbow-to-shoulder
EEG	Electroencephalogram
EM	Expectation-maximization
FCE	Functional Capacity Evaluation
FES	Functional electrical stimulation
GAS	Global asymptotic stability
GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression
H2E	Hand-to-elbow
HRI	Human-robot-interaction

KS	Kolmogorov-Smirnov (test)
LfD	Learning from Demonstration
LTI	Linear time-invariant
(M)AE	(Mean) absolute error
MTC	Motion tracker camera
NN	Nearest-neighbor (search)
PHRI	Physical human-robot-interaction
RMKT	Robotics-mediated kinesthetic teaching
SEDS	Stable estimator of dynamical systems
TDE	Time-delay estimation
TIL	Therapist-in-the-loop
TMKT	Telerobotics-mediated kinesthetic teaching
TOOL	Therapist-out-of-the-loop
VR	Virtual reality
WLS	Weighted least squares

Chapter 1

Introduction

1.1 Motivation

As the world's population ages, the demand for rehabilitation medicine services continues to increase. An older population equates to an increase in the prevalence of physical injuries (e.g., from falls) and mobility impairments, but also neurological disabilities such as those experienced by stroke survivors. For rehabilitation-focused post-stroke therapy (as opposed to therapy sessions for teaching compensatory techniques), strength training and neuroplasticity are regarded as key mechanisms of recovery. Neuroplasticity is the ability of the brain to alter and adapt its functionality through the formation of new neuronal connections. In stroke survivors, the repetitive exercise of affected neuromuscular pathways has been found to act as a catalyst for neuroplasticity [1]. As such, there is a strong motivation for rehabilitation services to emphasize repetitive physical exercise for both patients' strength training as well as neuromuscular recovery. However, this means the burden on healthcare providers also increases not only through increased cost expenditures [2] but also through the higher physical demands on therapists who have to support the patients in these repeated exercises. This has resulted in a need for innovation, as evidenced by the growth of research on technology-focused assistance over the last few decades [3].

Rehabilitation robots are one such innovation that has seen significant development over the last two decades. The ability of robots to provide repetitive, high-intensity interactions without being subject to fatigue makes them

an attractive means of providing the repetitive exercise that is fundamental to expediting a patient's rehabilitation [4]. A significant amount of research in the area has sought to improve the stability of these robots to make them patient-safe, as well as to provide them with the ability to adapt their behaviors, whether it be assisting or resisting a patient during exercise. However, despite having these advantages, attempts to show that robotic rehabilitation provides greater patient recovery than conventional rehabilitation medicine practices have been inconclusive [5]. This coupled with prohibitively high initial costs and potential patient distrust of robots have resulted in relatively low adoption of rehabilitation robots in standard clinical practices.

An important consideration is that the field of rehabilitation robotics should instead focus on the use of robots as supplementary to conventional therapy and as enabling tools in the hands of therapists, instead of as replacements for them [6]. In this frame of mind, the field can be further developed even if rehabilitation robots are as effective as (but not more effective than) conventional therapy methods if improvements are made in other areas (e.g., cost savings or freeing up the therapists' time). Providing semi-autonomy is one way to do so: semi-autonomy keeps the therapist in the loop but allows them to save time and effort through the robot taking a share of the intervention to be done on the patient. Learning from Demonstration (LfD) is an ideal method of introducing semi-autonomy in the field of rehabilitation robotics. LfD describes a family of machine learning processes in which a robot observes and learns certain desired behaviors (i.e., therapeutic intervention) from demonstrations made by a human operator (i.e., the therapist), allowing the robot to later reproduce the same behaviors in the absence of that human operator. It is a plausible method with which therapists with minimum programming experience can easily adjust not only the level of therapeutic assistance or resistance provided to a patient but also set up any number of different therapy tasks. This aspect of mutual adaptation, where users can explore and train robotic aides themselves, is an important step for rehabilitation robotics [7] and is proposed in this thesis as a viable method of making robotic therapy cost-effective and personalized.

1.2 Objective

The first overall objective of this thesis is to explore different implementations of the LfD paradigm and show how they can be incorporated into various areas of rehabilitation medicine in order for the robot to learn therapeutic interventions from therapists. The proposed intelligent robotic systems are developed with two sub-objectives in mind. First, the learning algorithms should enable the time-sharing of a therapist across multiple patients. Second, the system should be intuitive enough to allow therapists with minimal programming experience to reprogram the rehabilitation robot with relative ease. The second overall objective of this thesis is to introduce a generalized, proof-of-concept manipulator-based approach to both areas of rehabilitation medicine that have seen robotic integration, and to those that are still practiced with conventional, i.e., human-only, methods. Developing a generalizable rehabilitation robot and showing its applicability to different fields within rehabilitation medicine represents a step towards simplifying the provision of physical therapy, which usually requires a variety of equipment. In a clinical perspective, this can lead to potential space and cost-savings.

1.3 Organization of the Thesis

Chapter 2 provides a summary of the concepts and principles that form the basis of this thesis. The topics of interest, namely stroke rehabilitation, robotic rehabilitation, and the Learning from Demonstration (LfD) family of machine learning algorithms, are discussed.

Chapter 3 presents a system for learning position and velocity-based therapeutic interventions. A therapist and a patient interact over a telerobotic medium to perform a cooperative, 1 Degree-of-Freedom (DOF) task designed to target upper limb rehabilitation. The robotic system learns the behaviors of the therapist and the accuracy of its reproductions are then evaluated when the robotic system is interacting with the patient in the absence of the therapist.

Chapter 4 presents a system for learning position-based therapeutic interventions for a lower-limb rehabilitation task and represents a shift in the application of the LfD paradigm towards the use of single robot interfaces (as opposed to the teleoperation scheme used in Chapter 2). A therapist provides lifting assistance to a patient with simulated foot drop who practices walking on a treadmill. The robotic system learns the assistance provided by the therapist in order to provide minimum toe clearance to the practicing patient. The similarities in toe clearance amounts between the demonstrations and reproductions are measured as points of comparison.

Chapter 5 presents a system for learning impedance-based therapeutic interventions for an upper limb rehabilitation task. While learning therapeutic interventions in the form of desired positions is plausible, learning a therapist's guidance in the form of impedances is much more realistic and allows for safer and more intuitive interactions between the practicing patient and robot. The accuracy of the system's impedance reproductions is evaluated in a 1-DOF task by comparing the force output exerted by the robot to the force exerted by the therapist in the demonstrations.

Chapter 6 provides a comparison of feasibility between teaching a robot through teleoperation-based demonstration and teaching through a single robot interface. The amount of variance in demonstrations for user trials in a 1-DOF task when using each modality is taken as the primary measure of demonstration quality. Motivation is given for moving towards using a single robot interface for demonstration and reproduction of therapeutic interventions as it provides a more intuitive perception of controlling the robot.

Chapter 7 shifts focus from post-stroke rehabilitation to occupational rehabilitation for injured workers. A proof-of-concept robotic system that incorporates an augmented reality (AR) display for facilitating Functional Capacity Evaluations (FCE) is presented, providing a platform for future development of robotic rehabilitation systems in the area. The utility and realism of the system is evaluated by examining the biomechanics of user participants when performing a 3-DOF upper-limb task in a real-world version of the task, and with the robotic-AR system providing physical and visual feedback for a vir-

tual version of the task.

Chapter 8 summarizes the research performed in this thesis and suggests future directions to explore.

1.3.1 Note on Terminology

Throughout this thesis, the terms *therapist* and *patient* do not refer to actual clinical therapists or post-stroke/rehabilitation patients. The role of a therapist was always played by user participants who had no formal training in rehabilitation medicine. Patients were either simulated by healthy participants (who may have worn devices temporarily inducing symptoms similar to those seen in motor impairment such as in Chapter 4), or by equipment such as springs.

1.4 Contributions of the Thesis

This thesis makes contributions in the areas of robotic semi-automation in rehabilitation medicine, physically teaching a robot a desired intervention via LfD (known as kinesthetic teaching), and development of immersive rehabilitation technology.

1. **Introduction of LfD in rehabilitation robotics for learning therapeutic interventions.** The work performed in this thesis targets a wide array of topics in rehabilitation medicine. Chapters 3 and 5 provide insight into the implementation of LfD for supplementing robotic upper limb post-stroke therapy, while Chapter 4 does the same for lower-limb gait therapy. Automation of therapy in this manner allows for time-sharing of therapists between multiple patients in previously impossible ways and potentially allows for cost-savings in clinical settings. In addition to the novel application of LfD in these areas, the inclusion of the therapist in the automation process allows for an intuitive approach to tuning a robot's behavior that leverages their knowledge and experience. The proposed approach will be more intuitive since therapists are trained to provide hands-on intervention as opposed to adjusting

parameters through a conventional computer interface (i.e., through a graphical user interface).to provide the optimal amount of assistance.

2. **Learning therapeutic interactions for ADLs.** The tasks targeted in this thesis represent common Activities of Daily Living (ADLs), which will be described in greater detail in Chapter 2. This represents a shift away from the simpler generic movement-based therapies typically seen in robotic rehabilitation. While previous studies have explored the characterization of ADLs with machine learning techniques, the work in this thesis presents a novel method with which to learn specifically ADL-based, therapist-demonstrated interventions.

3. **Learning of position and impedance-based interventions through kinesthetic teaching.** Chapters 3 and 4 provide implementations of LfD algorithms for learning position-based therapeutic intervention. As a further step, Chapter 5 demonstrates one possible process to allow for learning more complex impedance-based interventions. The LfD algorithms presented are not novel by themselves and have been previously applied to robot control in the literature. Instead, their usage in this thesis is novel in that the algorithms are used to discriminate between two users in order to specifically learn assistive behaviors demonstrated by the therapist. The implementations of these algorithms for learning therapeutic interventions are novel in that they allow for either consecutive or simultaneous demonstrations of the therapy tasks. Consecutive demonstrations are where the patient and therapist provide separate demonstrations of the task and the difference between the two is learned as the desired assistance. Simultaneous demonstrations of therapy tasks are where the patient and therapist perform the task together, and the assistance is learned by treating the demonstration as an ideal target for the patient to achieve.

4. **A comparison between single-robot and telerobotic interfaces as mediums for LfD.** Single-robot interfaces are most often imple-

mented in LfD works. Telerobotic interfaces have received moderate interest in the context of LfD research, but are implemented in tasks that are already well suited to the advantages of telerobotics (such as underwater exploration and manipulation tasks). Chapter 6 provides a novel comparison of the two interface designs as media for providing demonstration datasets for the same upper-limb post-stroke rehabilitation task as in Chapter 5.

5. **Introduction of a generalized robotic rehabilitation system for occupational rehabilitation.** The field of occupational rehabilitation has seen few applications of robotics. Chapter 7 presents a novel, proof-of-concept robotic system for facilitating FCE of injured workers. Current robotic systems for FCE lack flexibility and immersion with regards to simulating FCE items (i.e., tasks). The presented system addresses both issues by utilizing a virtual environment, which is presented to the user through haptic feedback and projection-type AR for increased immersion.

Chapter 2

Background

This chapter provides a high-level summary of the concepts and principles that form the basis of this thesis. Section 2.1 discusses stroke, the concept of neuroplasticity and its application to stroke rehabilitation, and the importance of Activities of Daily Living in rehabilitation. Section 2.2 provides a brief overview of the concept of robotic rehabilitation and explains its advantages and disadvantages. Lastly, Section 2.3 discusses the branch of machine learning known as Learning from Demonstration along with its application to robotic programming.

2.1 Post-stroke Rehabilitation

2.1.1 Stroke

Stroke is the fifth leading cause of death globally, causing approximately 6.5 million deaths each year [8]. It is defined as "a clinical syndrome, of presumed vascular origin, typified by rapidly developing signs of focal or global disturbance of cerebral functions lasting more than 24 hours or leading to death" [9]. Common symptoms of stroke include a reduction in motor control (i.e., reduced movement, weakness, and incoordination), sensory loss or alteration, impaired balance, and impaired muscle tone (i.e., spasticity) [9]. In Canada, there are more than 62,000 cases of stroke each year, and 405,000 Canadians are living with long-term stroke disability [10]. In Canada alone, stroke and other cardiovascular disease cost the healthcare system $22.2B in 2009 [11], and $20.9B in 2015 [2]. Therefore, there is a great incentive for making post-

stroke rehabilitation as effective and efficient as possible, not only to lower its economic burden but to improve the quality of life for a significant number of stroke survivors.

2.1.2 Neuroplasticity and Stroke Rehabilitation

Neuroplasticity can be defined as the ability of the brain to reorganize its structure, function, and connections for the purposes of development and learning or in response to the environment, disease, or injury [12]. For stroke patients, research has found that neuroplasticity can be promoted in patients by actively engaging in repetitive exercises, and has been extensively explored for the purposes of aiding motor recovery [13, 12, 1].

Hemiparesis is one example of a manifestation of post-stroke neuromotor disability where the principles of neuroplasticity are taken advantage of. Hemiparesis can be described as the paralysis of one side of the body (and may also be characterized by spasticity and sensory loss) [14] which, in the case of stroke, occurs as a result of a hemispheric lesion in the brain. Degeneration of motorneurons as a result of hemiparesis contributes greatly towards resultant muscle weakness [14, 15], and may be further compounded by hemispatial neglect [16, 17]. Constraint-Induced Movement Therapy (CIMT) is a family of treatments that involves constraining the movement of a patient's unaffected limbs, and mass (i.e., high intensity) practice using their affected limb; this approach has seen significant success when applied to hemiparetic stroke patients with the objectives of overcoming learned non-use and promoting neuroplasticity [18]. More recently, the facilitation of neuroplasticity has shifted away from having patients practice generic and repetitive single-limb exercises and instead towards bimanual exercises (for upper-limb therapy) [19] or task-oriented therapy [20].

2.1.3 Activities of Daily Living

In the case of post-stroke rehabilitation, task-oriented therapy typically refers to the training and assessment of patients' abilities to perform Activities of Daily Living (ADLs). The most basic of ADLs typically include bathing,

personal hygiene and grooming, dressing, toileting, functional mobility (i.e., locomoting and transferring to and from beds and chairs), and self-feeding [21]. The definition, however, can also encompass most actions that allow an individual to live independently. Conventionally, patients will be rehabilitated through a combination of strength training and movement strategies, compensatory or otherwise, that allow them to perform ADLs; assessments of their capabilities are then performed by having patients perform the ADLs themselves. Therapists may also perform in-home training of patients for ensuring they can successfully carry out ADLs. However, the use of ADLs as rehabilitation tasks themselves is not standard, although the use of ADLs as a focus for stroke rehabilitation has been shown to provide improved independence and quality of life outcomes [22, 23].

2.2 Robotic Rehabilitation

Robots have been associated with the treatment of disability since 1989. Early uses of robots revolved around assisting patients living with disability to navigate their daily lives [24, 25]. Over the past decades, there has been a growing demand for rehabilitation services, motivating robotics technologies for assisting recovery following disability. As a result, the use of robots to reproduce repetitive rehabilitation tasks and therapeutic guidance interactions has become popular [26, 4]. A few particularly notable examples include the MIT-MANUS [27] and ARM guide [28] robots. In fact, many systems have been created that target elements of post-stroke rehabilitation mentioned before such as hemiparesis [29] and, more recently, ADLs [30, 31].

The inclusion of robots in therapy to provide therapist-robot-patient interactions presents distinct advantages over conventional therapist-patient interactions:

- Conventional hand-over-hand therapy for stroke patients, in which the therapist would monitor the patient and directly apply assistive/resistive forces when necessary, is too burdensome on the therapist. As a result, in practice, most therapy sessions are designed to maximize a patient's self-

direction, with a therapist designing and supervising interventions while providing mostly verbal guidance [32]. Otherwise, therapy sessions that involve the more preferred hands-on interaction between the therapist and patient must be limited in duration and intensity (where intensity refers to the amount of activity performed), resulting in less practice for patients. On the other hand, robots can perform repetitive movements with superhuman accuracy and reliability, but without suffering from fatigue. This characteristic suits the repetitive nature of strength training and task-oriented therapies alike and can alleviate the physical burden on therapists.

- Assessment in current rehabilitation practice is performed by using standardized assessments such as the Chedoke-McMaster Stroke Assessment [33], Fugl-Meyer Assessment of Sensorimotor Recovery After Stroke [34], and Modified Ashworth Scale [35]. These assessments are driven by therapists' observations and are therefore examined strictly for validity and inter-rater reliability. While specific assessments may rate highly for either criterion, there can never be complete certainty in the results they provide due to the subjective nature of human raters and the resultant coarse resolution of the assessments. On the other hand, sensors in a robotic system can provide numerical measurements that can describe a patient's performance during rehabilitation, which is ideal for supplementing the assessments mentioned.

- The ability of robots to be automated is one of their most important strengths. In the context of facilitating rehabilitation, the automation of rehabilitation robots provides an opportunity to streamline therapy. For example, the ability to time-share a single therapist across multiple patients more efficiently becomes possible. Another example is intelligently automating the amount of assistance or resistance provided during therapy, a concept known as Assistance-as-Needed (AAN), which has received significant interest [36, 37, 38, 39]. AAN makes it possible to introduce robotic assistance on a graduated basis, where the level of

automation (from fully manual to fully autonomous) that best suits the needs of the therapist and patient can be automatically chosen.

- Rehabilitative therapy, especially when presented in a "traditional" (i.e., face-to-face) manner, is inherently restricted by distance. Patients must either participate in rehabilitation sessions at a hospital or other rehabilitation center, or a therapist must visit a patient at their home. In the case where patients are situated in remote or otherwise difficult to access locations, providing rehabilitation may be exceedingly challenging and cost inefficient. Telerehabilitation is the concept of providing rehabilitative support, assessment and intervention over a distance, using internet-based communication as a medium for therapist-patient interaction [40]. This can take the form of purely audio or video communication, audio-visual communication with patient-robot (unilateral) interaction with performance communicated over the internet, or haptic (bilateral) interaction between a therapist side robot and patient side robot, also known as telerobotic therapy [41, 42, 43]. Through telerehabilitation, remote access is inherently addressed and has received significant focus [44, 45]. Another possible advantage comes in the form of early indications from longitudinal studies on telerehabilitation that show small cost savings [46].

Despite these advantages, however, robotic rehabilitation faces limitations. First and foremost is that analyses of the efficacy of robotic rehabilitation are largely inconclusive as to whether robotic rehabilitation is more or as effective as "conventional" therapy [5]. As a result, when put in context with the high initial costs of purchasing such robots, acceptance of robotic rehabilitation remains relatively low in clinical settings. Therefore, as targeted in this thesis, a more appropriate application of rehabilitation robotics is to not replace human therapists, but rather to find ways to complement their skills with the advantages mentioned above and to bring costs down in other ways. As long as these aspects of robotics can be applied in a cost-efficient manner, it is then sufficient for robotic rehabilitation to specifically be as effective as "con-

ventional" therapy, as opposed to more effective. Secondly, the programming of rehabilitation robots has always been done such that the robots provide interactions associated with a specific set of tasks, with no easy method of changing these tasks. As a result, the kinds of interactions a therapist can provide through the robotic medium are limited unless they or another person (usually a technician) are familiar with computer programming principles and can change the task and/or task-oriented behavior of the robot.

2.3 Learning from Demonstration

The concept of Learning from Demonstration (LfD) describes a family of machine learning techniques in which a robot observes demonstrations of a task by a human operator (the "demonstration" phase) and learns a policy to describe the desired task-oriented actions, which may or may not be acted upon by the robot in a "reproduction" phase later [47]. The terms "programming by demonstration" or "imitation learning" also refer to the same concept. The policy learned through LfD techniques is a central point to its innovation, and has seen implementation through mapping functions (classification and regression), or through system models (reinforcement learning) [48].

The advantages of using LfD techniques to program robots are clear. After the initial challenge of making the machine intelligent, i.e., teachable, programming the robot can be made as easy as physically holding a robot and moving it through a desired trajectory. Users themselves do not require knowledge of computer programming. The capabilities of the robot are completely dependent on the level of sophistication of the underlying learning algorithms and the amount of sensors used to characterize a behavior; with highly sophisticated algorithms and sufficient sensors, it is possible to teach more complex aspects of tasks to robots (e.g., understanding a user's intent). The methodology of LfD also requires a human user to be involved in the programming process, meaning the aspect of interacting with an actual human is preserved and conveyed by means of imitation. Lastly, like any other implementation of machine learning for robotics, LfD allows for automation, which translates to

time and cost savings.

Chapter 3

Learning Therapists' Position-based Behaviors for Upper Limb Rehabilitation

3.1 Introduction

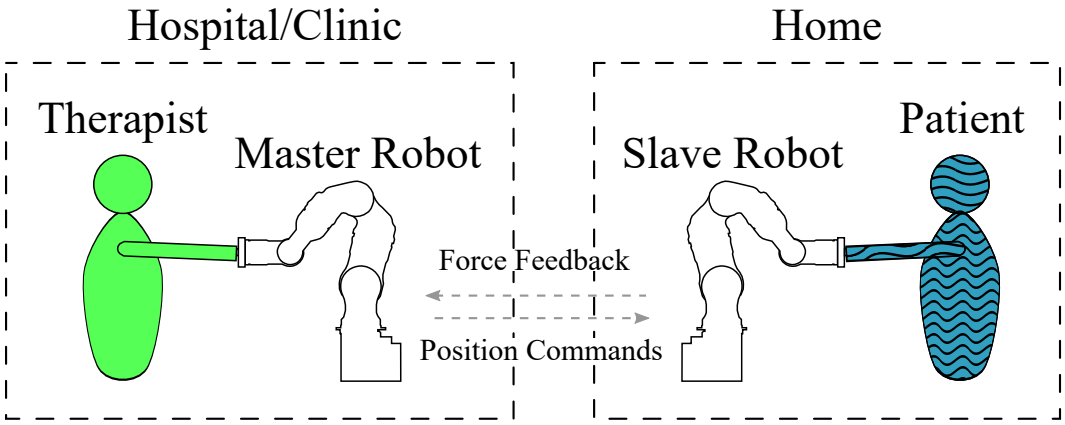
Over the past decades, there has been a growing demand for rehabilitation services, motivating robotics technologies for assisting recovery following disability. As a result, the use of robots to reproduce repetitive rehabilitation tasks and, for therapeutic purposes has become popular [26, 4]. Traditionally, haptics-enabled robotic rehabilitation has facilitated two categories of movement therapies: assistive therapy and resistive therapy. Assistive therapy involves the use of a haptic device to assist the patient to complete the task, while in resistive therapy the device will oppose the patient's actions to build muscle strength. Focus has more recently shifted towards functional therapies, in which the tasks reproduced during therapy are meant to directly emulate Activities of Daily Living (ADLs), translating into more meaningful and efficient recovery for patients [30].

Often, the behaviors of robots during rehabilitative therapies are preprogrammed, which is highly restrictive in the presence of unstructured task environments and given the variation in patients and therapists' abilities. This is in contrast to the flexibility with which a skilled therapist can adjust the parameters of therapy such as the therapy intensity. To directly incorporate the therapist's skills in robotic therapy, the field of telerobotic rehabilitation,

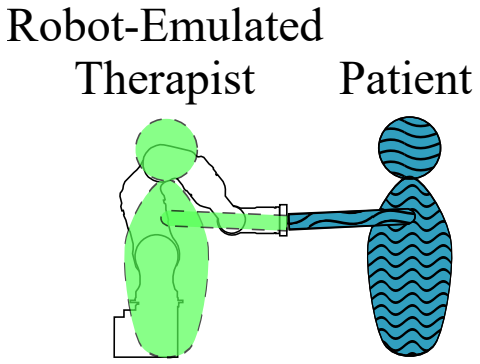
where there is one robot for the patient and one robot for the therapist, has received increasing interest [44]. In this chapter, our focus is on telerehabilitation through a bilateral (haptics-enabled) telerobotic system. Haptic feedback, which provides a human who operates a robot/tele-robot with a sense of touching a virtual/physical environment, allows interactions that are gentle, safe, reliable, and precise and is of high importance in rehabilitation robotics and telerobotics [49]. The main strength of haptics-assisted telerehabilitation is its ability to simulate the so-called "hand-over-hand" therapy over a distance. Haptic tele-robots are also the ideal vehicles for moving the rehabilitation process to the home and remote areas for increased access to and reduced costs of healthcare [50].

Telerobotic rehabilitation also allows therapist-administered therapies to take the form of cooperative tasks performed collaboratively by the therapist and the patient. We define cooperative, upper-limb tasks as tasks that require the use of two hands to complete [51], such as holding a jar and unscrewing its lid or lifting an object with two hands. Allowing for cooperative tasks to be practiced not only provides more variations of therapy tasks to administer, but also provides therapists an opportunity to monitor and guide patients in situations where they may undesirably compensate for their affected limb with the contralateral limb.

We refer to the situation where the therapist is haptically interacting with the patient remotely as Therapist-In-Loop (TIL). While TIL bilateral telerehabilitation has many advantages over unilateral telerehabilitation, a therapist may not always be available to interact with the patient over the telerehabilitation medium. In fact, since the number of patients afflicted with strokes has increased in recent years [52], the number of therapists and the hospital resources may become tightly stretched across patient caseloads in the future, resulting in a lack of proper health care delivery. A solution to this problem proposed here is to have the patient-side robot first learn the therapy administered by the therapist during the live telerehabilitation session, and then imitate it. As a result, in the absence of the therapist, the patient can continue to practice the task in cooperation with the semi-autonomous



(a)



(b)

Figure 3.1: Illustrations for the TIL phase (a) where the patient interacts with the therapist, and TOOL phase (b) where the patient interacts with a slave robot that emulates the therapist's behavior.

patient-side robot. We refer to this situation where the therapist is absent as Therapist-Out-Of-Loop (TOOL). The paradigm to transition from TIL to TOOL will be based on learning from demonstration (LfD) techniques, which will be discussed later. Fig. 3.1 depicts the TIL and TOOL phases.

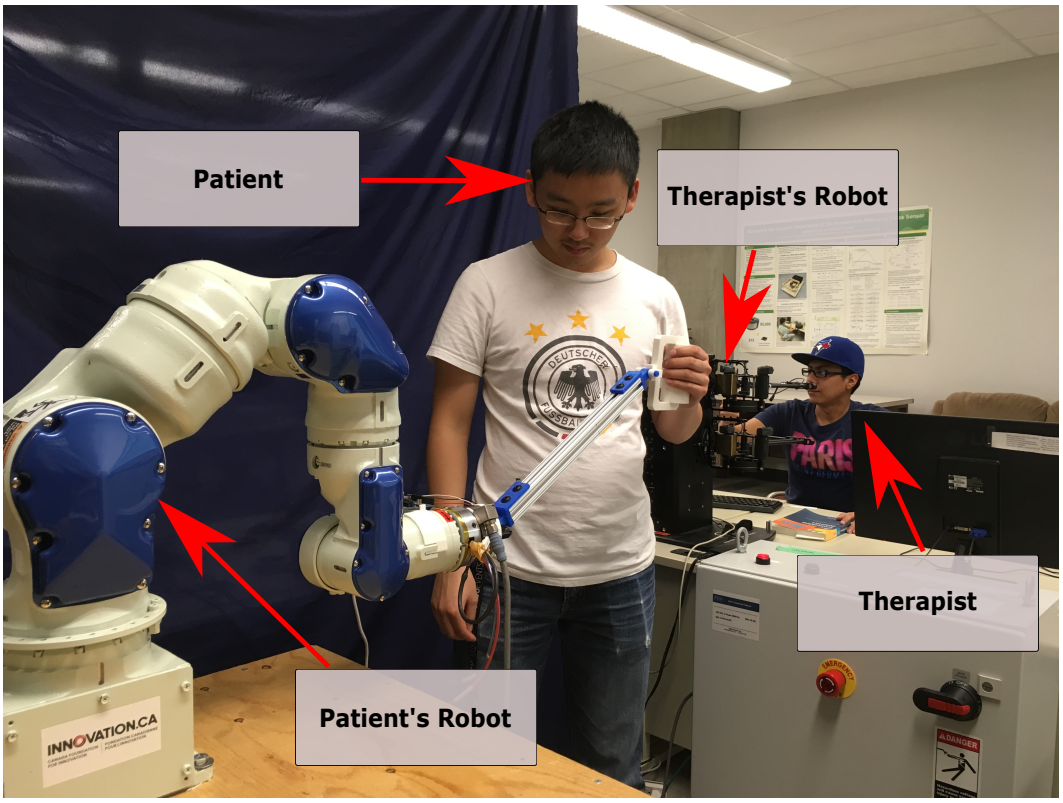
In this chapter, we are interested in creating a variable-difficulty cooperative task. The task will be cooperatively performed, with the therapist controlling the slave robot through the master robot to intervene in the task and the patient interacting with the task directly. The master and slave robots as well as the placement of the therapist, the patient, and the task are shown in Fig. 3.2. LfD based on Gaussian Mixture Models (GMM) and Gaussian Mixture Regression (GMR) will then be implemented for the purposes of learning how the therapist interacts with the patient. We hypothesize that the com-

bination of these techniques can provide a suitable middle ground between hand-over-hand and fully semi-autonomous therapy.

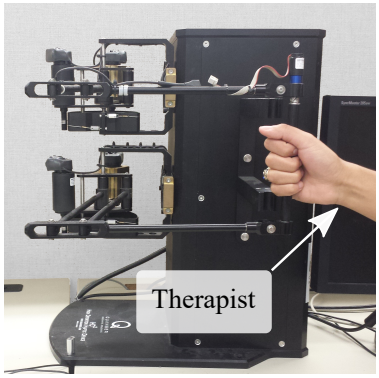
The chapter is divided in six sections: Section 3.2 provides a quick introduction to LfD, GMM and GMR. Materials and methods are discussed in Section 3.3. Section 3.4 presents results while Section 3.5 outlines conclusions and discusses future directions.

3.1.1 Prior Art

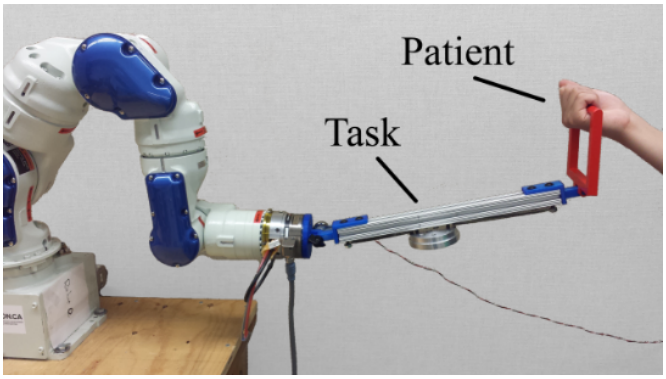
The concept of semi-autonomous systems and LfD has seen extensive research in the past few decades. Application of LfD principles to human-robot interaction has naturally led to exploration of cooperative tasks. Calinon et al. [53] taught a robot to cooperatively lift a beam in a setup similar to what we propose here. Gribovskaya et al. [54] built upon the same work to ensure global asymptotic stability (GAS) of the system. Peternel et al. [55] created a variant to learn motion and compliance during a highly dynamic cooperative sawing task. However, few groups have applied LfD techniques towards the practice of physical therapy in rehabilitation medicine. Maaref et al. [56] described the use of LfD as the underlying mechanism for an assist-as-needed paradigm. Lydakis et al. [57] learned and classified demonstrations of therapy tasks through EMG measurements. Lauretti et al. [58] optimized a system built on dynamic motor primitives for learning therapist-demonstrated paths for activities of daily living. Najafi et al. [59] learned the ideal task trajectory and interaction impedance provided by an able-bodied user and provided user experiment evaluations. These previous works show well-developed innovations in human-robotic interaction, robot-cooperative tasks and LfD in separate and different contexts. Our work gathers these different ideas into a single system to create a new way to provide human post-stroke therapy. We propose to combine the best and most important contributions of each of these works to show that robotic therapy can be streamlined for greater practicality.



(a)



(b)



(c)

Figure 3.2: (a) Experiment setup and demonstration; (b) HD^2 High Definition Haptic Device (Quanser Inc., Markham, Ontario, Canada) used as the master robot by the therapist; (c) a Motoman SIA-5F (Yaskawa America, Inc., Miamisburg, Ohio, USA) industrial robot.

3.2 Learning from Demonstration

LfD is a paradigm focused on allowing a human user to program a robot through demonstration of desired behaviors. In other words, a trainer (which can be a human or even another machine) physically demonstrates the behaviors to be imitated by the robot [60, 61, 48]. In general, the behaviors are actions or movements to be later imitated by the robot.

A cornerstone and driver of our LfD-based approach is the assumption that programming know-how is limited in clinical settings. This requires that reprogramming the robotic system between different tasks must be made as simple and user-friendly as possible. State-of-the-art LfD techniques allow for this and facilitate robot learning based on only a few real demonstrations of the task by a human without any additional computer programming overhead.

LfD is divided in two phases, known as the demonstration and imitation phases. In the demonstration phase, a trainer interacts with the robot and performs an action that is to be learned by the robot. Multiple demonstrations of the task can be completed in order to provide a wider knowledge base for the robot. The imitation phase then reproduces the learned behavior based on the inputs the robot receives in real time. There are different approaches to learning and imitating a desired behavior.

In this chapter, GMM and GMR are used as the underlying learning and imitation algorithms for the LfD paradigm. The GMM algorithm takes multiple demonstrations and extracts the necessary parameters to describe the data with Gaussian functions. This process avoids redundancy of data in memory. The GMR algorithm uses the stored data and, based on the regression input, retrieves the general form of the output.

3.2.1 Gaussian Mixture Models

GMM is a probability density function widely used for generatively modeling data [62, 63]. The model parameterizes a set of datapoints and its underlying function as weighted sums of Gaussian component densities, with each Gaussian having its own mean and covariance. Because of the simplistic, adaptable

nature of Gaussian functions and the advantages that come with generative modeling, GMM is widely used for LfD.

GMM is a weighted sum of K component Gaussian densities given by the equation,

$$p(\xi_j) = \sum_{k=1}^{K} p(k)p(\xi_j|k) \tag{3.1}$$

where $p(k)$ are the prior probabilities, $p(\xi_j|k)$ is the conditional density function, and ξ_j is the D-dimensional continuous-valued data vector.

The parameters in (3.1) are defined as

$$p(k) = \pi_k \tag{3.2}$$

$$p(\xi_j|k) = \mathcal{N}(\xi_j; \mu_k, \Sigma_k) = \frac{1}{\sqrt{(2\pi)^D|\Sigma_k|}} e^{-\frac{1}{2}\left((\xi_j-\mu_k)^T\Sigma_k^{-1}(\xi_j-\mu_k)\right)} \tag{3.3}$$

Each k$^{\text{th}}$ Gaussian component is described by the parameters $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$, representing respectively prior probabilities, mean vectors and covariance matrices. The Expectation-Maximization (EM) algorithm is widely used to train GMM parameters. It takes the GMM parameters and iterates them until convergence of an optimization factor. EM has a simple local search technique that guarantees increase of the likelihood.

3.2.2 Gaussian Mixture Regression

The GMR model uses the Gaussian conditioning theorem and linear combination properties of Gaussian distributions to retrieve the desired output values from a GMM [63]. GMR traditionally uses temporal values (ξ_t) as query points to estimate the corresponding spatial values ($\hat{\xi}_s$) through regression. Given a set of temporal and spatial values for a kth component of a GMM, the representations of the mean and covariance matrices are given as

$$\mu_k = \{\mu_{t,k}, \mu_{s,k}\}, \Sigma_k = \begin{pmatrix} \Sigma_{t,k} & \Sigma_{ts,k} \\ \Sigma_{st,k} & \Sigma_{s,k} \end{pmatrix} \tag{3.4}$$

Conditional expectation ($\hat{\xi}_s$) and conditional covariance ($\hat{\Sigma}_s$) of the output ξ_s given ξ_t are then calculated for a mixture of all GMM k components.

Note that while the query points are described as temporal points, these inputs to the GMM and GMR can be any type of data. As is the case in

Figure 3.3: Design of the cooperative task. The slave robot holds one side of the bar, while the patient holds the bar from its other side. The bar's inclination can be altered in a 180° range.

our work, the learned system behaviors can be time-independent, and spatial coordinates, as an example, can be used as the query points.

GMR only requires the means and covariance matrices generated by the GMM to retrieve the signal. This helps to use memory more efficiently; otherwise, each time step would need its own mean and variance values (as in the case of lazy learning algorithms).

3.3 Experiments, materials, and methods

3.3.1 Materials

The teleoperation system has two robots: a master robot (Quanser High Definition Haptic Device, or HD^2) used directly by the therapist, and a slave robot (Yaskawa-Motoman SIA5F) handled by the patient. Even though both robots have upwards of seven DOF, the movements of the users and robots are constrained to only one DOF due to the nature of the cooperative task.

A potentiometer is used to measure the angle θ that a bar attached to the Motoman makes with the horizontal axis. A mass is placed on the bar, and allowed to slide along the length of the bar. Two identical springs attached to opposite sides of the bar pull the sliding mass towards their respective sides. Fig. 3.3 shows the design of the bar.

3.3.2 Methods

Task

Hemiparesis after stroke can leave patients unable to lift their affected arm without significant difficulty or discomfort. As a result, performing ADLs can be challenging for the patients. A common solution to these problems is to compensate the affected limb using the unaffected limb [64]. In other words, patients tend to use their unaffected limb to carry more weight during a given task, thereby neglecting their affected limb. This can result in a poor improvement of the affected limb and development of poor motor habits [64]. A commonly administered therapy activity for this kind of upper limb weakness involves having patients combine shoulder forward flexion, horizontal extension, abduction, and elbow extension to move the affected limb upwards and away from their trunk in a natural and synergistic manner [65].

We utilize a modified version in this chapter; the task now requires the therapist and the patient to collaborate to lift a bar. The spring-mass system on attached to the bar will allow the mass to slide towards one end of the bar in a manner directly proportional to θ, similar to if a box was being lifted by the participants with objects inside of it that slide back and forth freely. The therapist can thereby adjust the amount of force the patient must exert to lift the bar (i.e., the therapy intensity) by either lowering or raising his/her own end, effectively resulting in an assistive/resistive therapy provided in the context of a functional task.

GMM and GMR design

We use a GMM to learn the therapist's behavior during the task (trajectory of the movements) and GMR to reproduce them. The demonstration phase uses GMM to create K Gaussian distributions of dimensionality D. In this chapter, K has a value of 12 (decided experimentally) and D has a value of 3. D has as many dimensions as inputs to the GMM. These inputs are:

- Therapist position in vertical axis (X_{Th})
- Patient position in vertical axis (X_{Pa})

- Patient velocity in vertical axis ($\dot{X}_{Pa}$)

These inputs are used in the GMM algorithm to learn the trajectory of the therapist's movements. We implement the GMM and GMR algorithms using the code presented in [66].

The imitation phase uses GMR to retrieve the trajectory of the movements. The GMR algorithm takes X_{Pa} and $\dot{X}_{Pa}$ as inputs, and based on the GMM distributions, retrieves an appropriate value for X_{Th} as an output. Fig. 3.4 shows the process of learning and reproducing the therapist's behavior with the given GMM inputs.

Experiments

In order to show the accuracy and robustness of the system, we split the experiment into two phases. In the first phase, the system is trained to execute a single behavior, which could be to assist the patient, resist the patient or split the weight in a neutral way. These scenarios are also known as "plus 45", "negative 45" and "zero", corresponding to the angles at which the bar is held by the therapist to achieve assistance, resistance, or a neutral pose respectively. In this phase, the system is trained to keep a constant angle during the whole therapy/task/experiment. This implies that during this phase, only a single therapist behavior can be learned; if the system is trained to assist the patient, it will not be able to change that task unless the therapist records a different demonstration.

In the second phase, the system is trained with different scenarios. The idea is to create an adaptive system capable of assisting the patient, resisting the patient, or keeping a neutral behavior with the patient, now based on the patient's performance. To do so, the GMM is trained with three different demonstrations of every scenario. Later, during the demonstration phase, the GMR takes the patient's behavior as input to reproduce the therapist behavior.

The system measures the patient's position and velocity to learn the patient's behavior. Based on the patient's velocity, the therapist can make a decision on how much assistance or resistance to apply. We selected four different general scenarios for the system to learn, described as follows. A positive

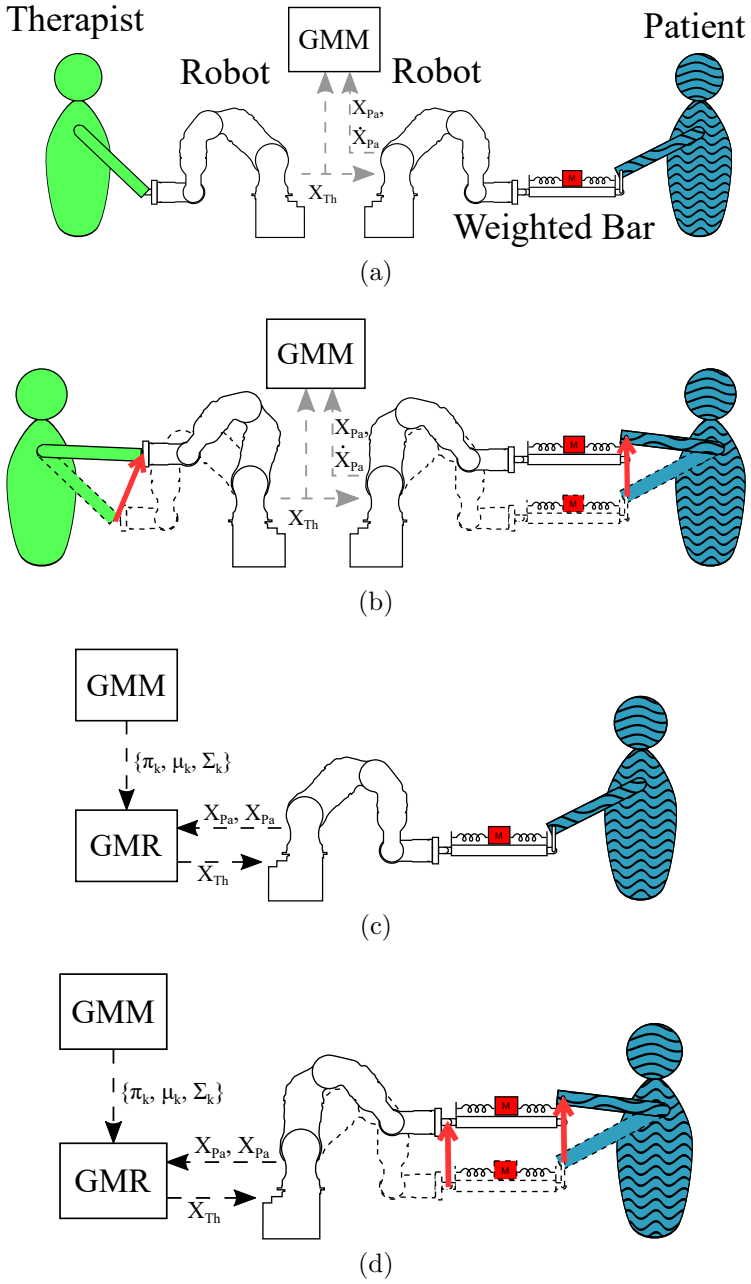


Figure 3.4: The two phases of LfD are shown separately through (a)-(b), and (c)-(d), respectively. In (a), the therapist is present (making this the TIL phase). The patient will initiate movement as they lift the bar, to which the therapist will respond as shown in (b). The data from both robots will be recorded and used to generate the GMM. Then in (c), the patient is practicing in the absence of the therapist (the TOOL phase). The robot utilizes GMR to emulate the therapist and respond to the patient's movements, as shown in (d).

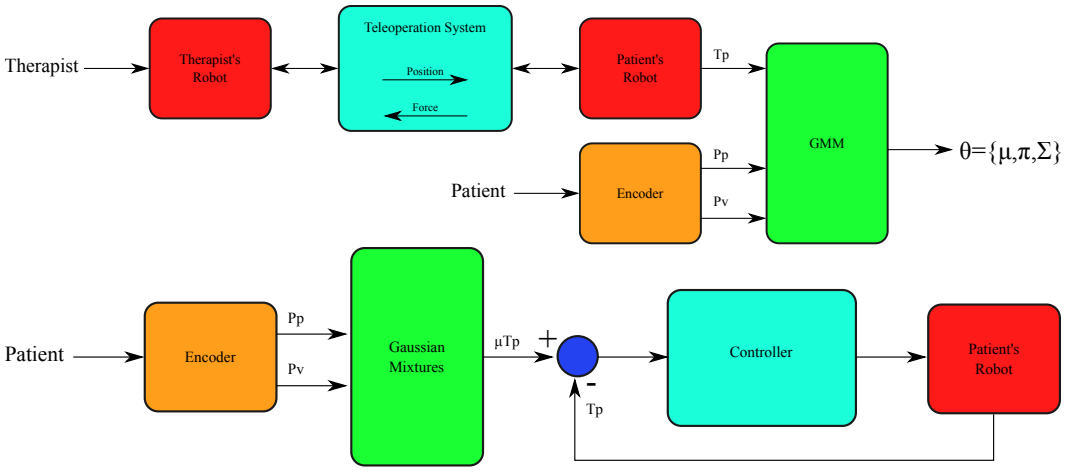


Figure 3.5: Summarizing block diagrams of the system's components needed to execute the learning and imitation phases. Top figure shows a block diagram for the demonstration phase while the bottom figure shows a block diagram for the reproduction phase.

and fast velocity means that the patient is able to easily perform the task and a resistance can be applied to challenge the patient during the therapy. A positive and medium velocity means that the patient can perform the task and the therapist only has to keep/maintain a neutral behavior. A positive and slow patient's velocity means that the patient has some problems/difficulties in performing the task so assistance is provided by the therapist. Finally, a negative patient's velocity means the patient is experiencing significant difficulty and is unable to perform the task. An even greater amount of assistance is then required to help the patient complete the task. These scenarios will also be referred to as "fast", "medium", "slow", and "back", respectively.

Two block diagrams of the system are shown in Fig. 3.5.

3.4 Results and discussion

Two able-bodied participants played the roles of the therapist and the patient. During the experiments, the therapist's position, patient's velocity and the patient's position data were recorded and then used in the GMM algorithm to train the robot. Three demonstrations were used to train the GMM. After the demonstration phase, the system's imitation performance was tested. The

GMR model takes the patient position and patient velocity as inputs and returns the estimated therapist position as an output. This estimated therapist position was used to move the slave robot in the imitation phase. Therapist positions and patient positions are mainly used in analysis of the results.

We present our results in two parts: first a qualitative examination of the system's imitation results for both phases, and second an evaluation of the training data's efficacy.

3.4.1 GMR output for different patient behaviors

GMR output results are shown in Fig. 3.6. The figures show results for the different scenarios in phase one and phase two respectively. In phase one, we show the results for assisting the patient, resisting the patient and splitting the weight in a neutral way for velocity-independent patient trajectories. For phase two, we show results for assisting, resisting or keeping a neutral behavior given velocity-specific patient trajectories.

The obtained plots show how the system is able to respond similarly to how a reference therapist would. Based on the input data, the GMR output demonstrates a reasonable accuracy through most of the different scenarios. Results for the first phase, shown in Fig. 3.6, show the system behaving exactly as in demonstrations, with clearly defined behaviors for assisting, resisting, or remaining neutral. In the second phase, the quality of the reproductions vary across the imitated behaviors. For patient trajectory data with higher velocities, GMR returns accurate trajectories with low variance, as in Fig. 3.6 (f). For slower velocities however, velocity measurements are heavily affected by noise from hand tremor, muscle fatigue, etc.; the "slow" and "back" scenarios seen in Fig. 3.6 (d) and (g) exhibit this problem. Results for these scenarios are less accurate, often switching between behaviors. A simulated data trajectory is also used to show the system's response through different scenarios given an ideal patient motion trajectory with minimal velocity fluctuation. In this situation the system produces very accurate results in accordance with the behaviors desired when training the GMM, indicating that the system could provide a perfect imitation for realistic inputs if properly adjusted. Know-

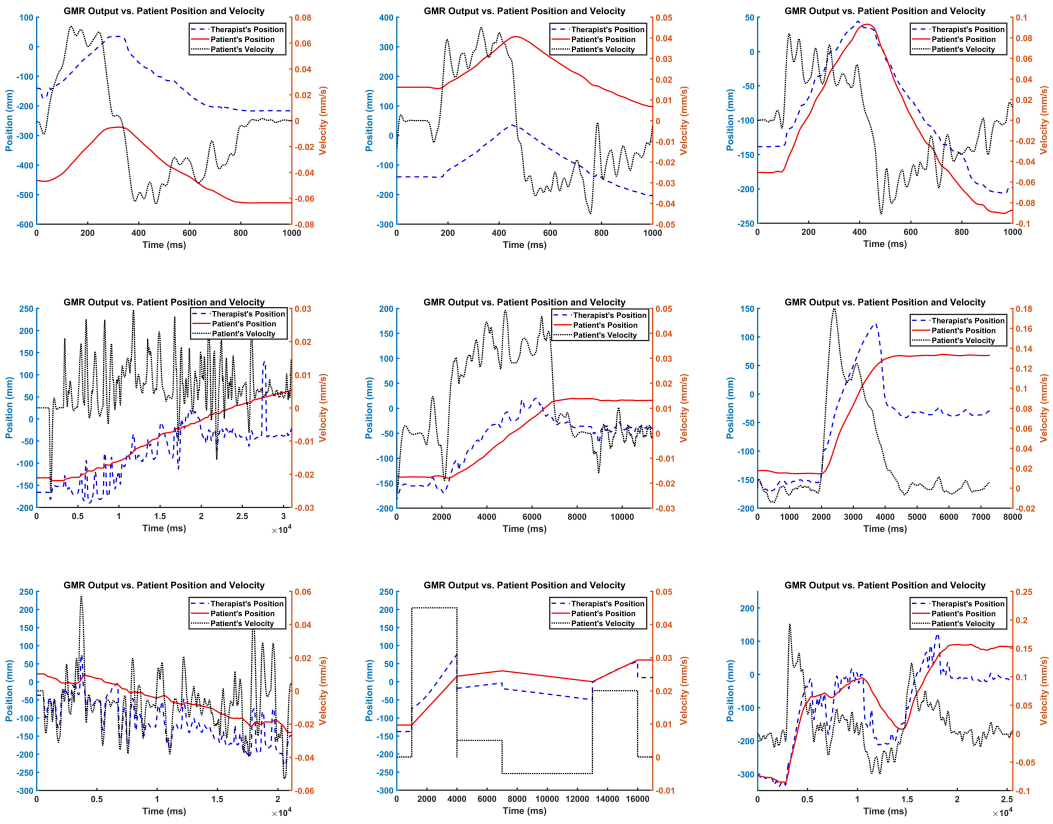


Figure 3.6: Results from phase one (plots a to c) and phase two (plots d to i). The plots shows the GMR output (blue and dashed line), the patient's position (red and solid line) and the patient's velocity (black and dotted line). (a) Shows the results for the negative 45 scenario, (b) shows the result for the plus 45 scenario, and (c) shows the result for the zero scenario, (d) show the results for the slow scenario, (e) shows the result for the medium scenario, (f) shows the result for the fast scenario, (g) shows the result for the back scenario (h) shows the result for the simulated data, and finally (i) shows the result for multi-behavioral data.

ing this level of accuracy is possible, this encourages further exploration of methods to adjust the training of the model to account for the aforementioned variations in patient data in order to achieve similar results for more realistic patient input. Finally, a real complex demonstration that combines multiple behaviors is used to show the robustness of the system, meant to resemble the interaction between a therapist and patient during a mock therapy exercise. The results show the system's behavior when presented with all the possible scenarios of patient motion, as well as including the transitions between scenarios. The system responds quite accurately throughout the task, but transitions are made too quickly to be safely implemented in clinical settings. Designing a different motion controller, for example based on adaptive control principles, is an attractive possible solution.

3.4.2 Evaluation of Training Data Quality

Motivated by the previous results, we examine the efficacy of the dataset used to train the system. In the first experiment, a total of 12 demonstrations were recorded for training the GMM, with N = 3 demonstrations for each of the four behaviors. We now remove a single demonstration and use it instead as the input for the GMR process; this is performed for every demonstration dataset used for training. All demonstrations performed are assumed to be valid, i.e. the therapist's responses to the patient's actions are always intentional. By operating under this assumption, we can find demonstrations that are less useful if their trajectories are already included in the system. We quantify this as the error between the reference therapist trajectory, used to train the system, and the GMR output. Results are presented in Fig. 3.7. For most of the demonstrations, accuracy suffers during slower speed behaviors as mentioned previously. Standard deviation results indicate the reference trajectories are typically within GMR output for higher speeds (Fig. 3.7 (c)), while for lower speeds the trajectories differ significantly (Fig. 3.7 (a) and (d)).

Table 3.1 provides the average error between the GMR output and the recorded therapist position for every removed demonstration. These results reinforce our earlier observation that responses to faster patient motions are

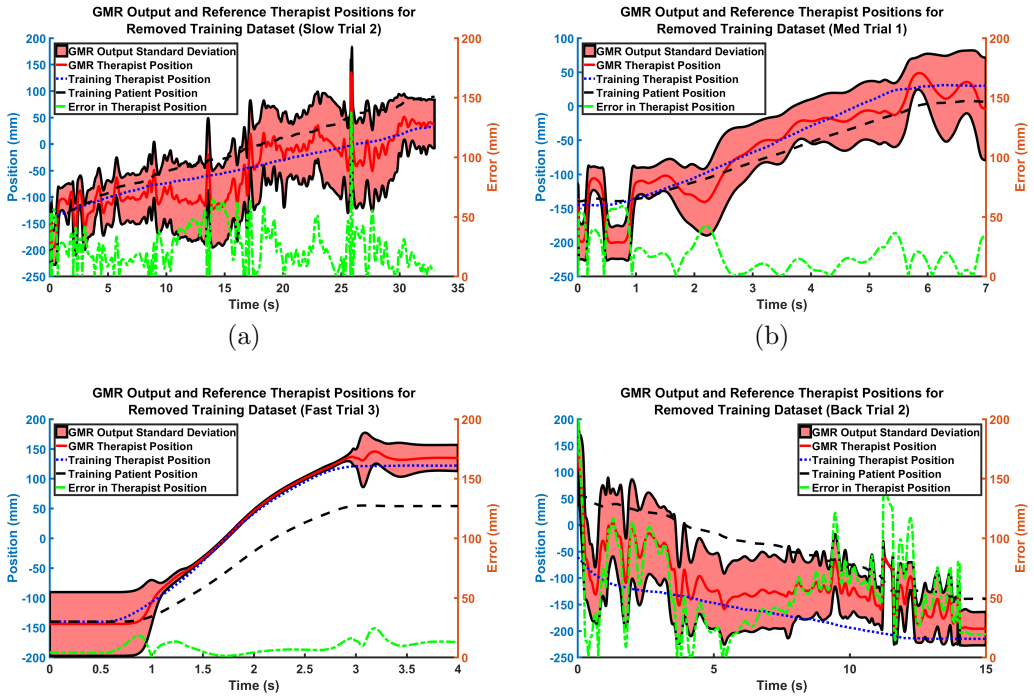


Figure 3.7: Analysis of the dataset quality for sample trials of each behavior in phase two. (a) shows the analysis for the slow scenario, (b) shows the analysis for the medium scenario, (c) shows the analysis for the fast scenario, and (d) shows the analysis for the back scenario. Behaviors associated with slower patient velocities, i.e., (a) and (d), show large inaccuracies when the size of the training dataset is reduced.

Table 3.1: Error between GMR output and recorded therapist position, averaged over the duration of the demonstration. Three trials are provided for each therapist behavior, which represent the removal of one of the three training demonstrations for cross validation.

Trial number	Average error between GMR output and recorded therapist position (mm)			
	Slow	Medium	Fast	Back
1	34.576	18.088	17.661	40.135
2	24.320	35.364	15.404	55.013
3	30.739	36.326	8.549	53.515
Overall average	29.878	29.926	13.871	49.554

captured better than slower movements with a limited number of demonstrations. By extension, we can infer that the system is able to better fit the Gaussian components to higher velocity data. Interestingly, results for overall average error of the slow and medium cases in particular are very similar. This may indicate that the GMM may not be able to distinguish between the velocities of the two cases well, resulting in equal sensitivities when demonstrations from either case is removed.

A resultant suggestion for works in the field of rehabilitation medicine looking to incorporate LfD principles would be to provide more demonstrations when aiming to imitate motions with large inherent variation, such as the slower movements seen in this work. Without resorting to more complicated regression methods with a greater focus on ensuring stability, providing more demonstrations for trajectory spaces with high uncertainty is the simplest method of better defining task space behaviors. Other methods of modelling, such as polynomial surface fitting, could conceivably provide a reasonably simple implementation of learning the task space but at the expense of ease of programmability.

3.5 Conclusion

In this chapter, LfD techniques were applied to a cooperative therapy task performed through a telerobotic system. The demonstration and imitation phases

of LfD were based upon GMM and GMR approaches, respectively. The goal was to accurately replicate the therapist's actions during a cooperative object lifting task, in which assistance or resistance was provided by the therapist through holding the therapist's side of the object lower or higher, respectively. Semi-autonomous imitations of therapist movements were performed with varying amount of success. Demonstrations provided by the therapist in response to faster patient movements were better learned, while slower patient movements had larger variations in velocity and produced less accurate imitations of the therapist's behavior. Examining the sensitivity of the system to the number of demonstrations provided for each scenario showed the differences between the GMR produced interactions and those of a user representing a therapist were shown to be fairly small (between 8.549 and 17.661 mm) at higher patient velocities, but increased substantially (up to 55.515 mm) for more fine patient motions involving lower velocities due to the variance inherent to the motions.

Future work in this study will emphasize applying this semi-autonomous telerobotic LfD paradigm to more complex tasks involving multiple DOFs as well as more components of movement performed by the therapist. As part of the future improvements to this project, a different LfD technique called Stable Estimator of Dynamical Systems (SEDS) [66] will be used to ensure GAS. Additionally, we aim to bring the proposed system to a clinical setting as part of our group's ongoing goal of conducting patient studies.

Chapter 4

Learning Therapists' Position-based Behaviors for Lower Limb Rehabilitation

4.1 Introduction

As the world's population increases in age, the demand for rehabilitation services increases with it. Post-stroke therapy, which is just one of many fields of rehabilitation medicine that is seeing growth in demand, emphasizes repeated activation and use of a patient's affected muscles in order to reassociate damaged neural structures and regain muscle tone [1]. Doing so can allow a patient to relearn how to perform activities that are categorized as essential to living, commonly referred to as Activities of Daily Living (ADLs), and thereby regain some degree of independence. ADLs are often difficult to perform for stroke survivors because of their lack of muscular coordination and muscle weakness. Hemiparesis, in particular, is common to stroke survivors, in which the decrease in muscular coordination and tone present themselves on one side of the survivor's body. ADLs can typically be divided into those that demand the use of a patient's upper limbs, such as cooking or grooming, and those that involve their lower limbs, such as walking, sitting, or standing. In performing lower limb ADLs, the difficulty experienced by hemiparetic patients can lead to the development of compensatory strategies, which may result in undesirable and unsafe walking patterns, sitting, and standing transfers, or in extreme cases the neglect of their affected side [67, 17]. As a result, specific

physiotherapy routines are designed in order to assist patients in relearning and reinforcing correct gait habits.

Body weight supported treadmill training (commonly referred to as BWSTT) is one such routine. BWSTT involves the suspension of a patient using a harnessing device above a treadmill, such that a portion of their body weight is relieved. If the patient is ambulatory, this allows them to practice walking without needing the full muscle coordination and tone normally involved. If the patient is non-ambulatory, typically 2-3 therapists will be present during therapy and will physically hold and move the patient's lower limbs through a proper walking pattern, allowing the patient to experience walking [68]. In non-ambulatory patients, in particular, BWSTT has been shown to provide greater improvements to participants' gait than traditional lower limb physiotherapy, which consists of simple muscle strengthening exercises [69, 68, 70].

However, the physical exertion required by therapists in the case of non-ambulatory patients is very intensive; therapy sessions are often limited to 15-20 minutes and with only one session per day [71]. Numerous robotic solutions have been proposed in the past decade in an effort to alleviate this physical burden [68, 72, 73, 74]. These technologies typically take the form of exoskeletons, treadmill or footplate-integrated robots, powered orthoses, or functional electric stimulators (FES).

Robotic lower-limb assistive devices have traditionally been programmed to assist the user in following predefined trajectories (i.e., gait patterns) with minimal allowed deviation [75]. This is less than ideal, as patients often feel like they are fighting the robotic assistance, should their desired movements not match with the assistance provided [76]. More recently, adaptive control schemes have received a large amount of attention. Lower-limb exoskeletons and ankle-foot orthoses, in particular, have had a significant number of such strategies developed for them. [77] utilizes adaptive oscillators as Central Pattern Generators (CPG) in order to provide natural gait patterns through learned motor primitives, while [78] applies a similar concept based on using EEG measurements as input to a CPG. [79, 80] incorporate force-feedback reflex-based neuromuscular models. [81] uses complementary limb motion esti-

mation to provide assistance to the affected lower-limb that mimics the motion of a user's healthy limb. [82, 83] implement variable-impedance controlled systems, based on different measurements of the user's performance. However, one common issue with these systems is that the role of a supervising therapist is limited. At most, their role is to provide the exoskeleton with trajectories of ideal gait (in the case of motor primitives), after which variation of the assistance level is fully up to the adaptive algorithm. This is typically undesirable for therapists, as the idea of robotic automation with minimal input from clinicians is poorly received [84]. Another possible form of adaptation we propose would be to allow the therapist to provide and modify behaviors (for the robot to imitate) as they see fit, where the behaviors do not have to be ideal but could be the most appropriate for a patient's capabilities. In this manner, the therapist stays involved in therapy as much as possible while the robot alleviates only the burdensome physical aspects.

The application of machine learning techniques to enable this adaptation across robotic rehabilitation has gained increasing interest in recent years. In upper limb rehabilitation, several uses of this same paradigm have been explored [85, 86, 87]. The inclusion of machine learning in lower limb robotic assistance has received attention mainly on recognizing gait cycle movement patterns [88, 89], but less on learning the correct corresponding assistance dependent on these patterns. A method of providing therapists with finer control for tuning the amount of assistance provided to patients during gait therapy instead of using predefined fixed assistance regimes observed in traditional assistive robotics is thus examined.

The purpose of this chapter is to propose a proof-of-concept treadmill-based gait therapy system that utilizes machine learning techniques to allow a therapist to intuitively define the amount of assistance provided to a patient and to allow a robot to learn and later reproduce the same assistance in the absence of the therapist. This is done through the use of Learning from Demonstration (LfD) techniques, where a user typically physically holds and moves a robot along a trajectory which the robot learns and is later able to imitate. The system learns by observing and generalizing between multiple

demonstrations from a therapist, as opposed to following a single desired trajectory. Additionally, this method is advantageous for use in clinical settings, as the therapists can train the system without having to possess computer programming know-how. To our knowledge, this is the first application of this paradigm to the study of lower limb rehabilitation.

We draw inspiration from the design of the KineAssist, a robotic treadmill-based assistive device that operates only in one degree of freedom (DOF) [90]. The KineAssist consists of a harness that holds the patient, attached to a vertical linear rail at the back of a treadmill. Force sensors on the device allow a therapist to provide lifting assistance to the patient during parts of ADLs or the gait cycle that require compensation for the patient's disability. This design maximizes patient participation while maintaining patient stability in order to prevent falls, but lacks machine learning capabilities. Our system is based on a robot manipulator (i.e., industrial robot) available to us, instead of a mechanized rail, as in the case of the KineAssist, or a wearable robot, as in the case of most gait therapy solutions. The robot will provide lifting assistance to the user representing the patient through a rope and pulley system that will hoist the harnessed participant in one DOF (in the vertical axis); the exact setup will be discussed later. Foot drop is a commonly observed pathology in stroke survivors which we focus on in this chapter. Toe clearance during the swing phase of the affected limb is a major difficulty experienced by patients with foot drop [91]. Our objective is to learn and train the system to provide an adequate amount of lifting assistance so as to provide the minimal toe clearance for the patient to be able to practice walking by themselves and more specifically the dorsiflexion of their affected foot. We will examine if the trained system can assist the affected limb in such a way that its toe clearance matches that of the unaffected foot during assistance as well as the toe clearance values of both feet of a healthy individual.

The chapter is structured as follows: Section 4.2 outlines the approaches to LfD and human-robot interaction (HRI) incorporated into this work, Section 4.3 describes the experimental setup and presents results, 4.4 provides discussion of the system and its performance, and 4.5 provides concluding remarks

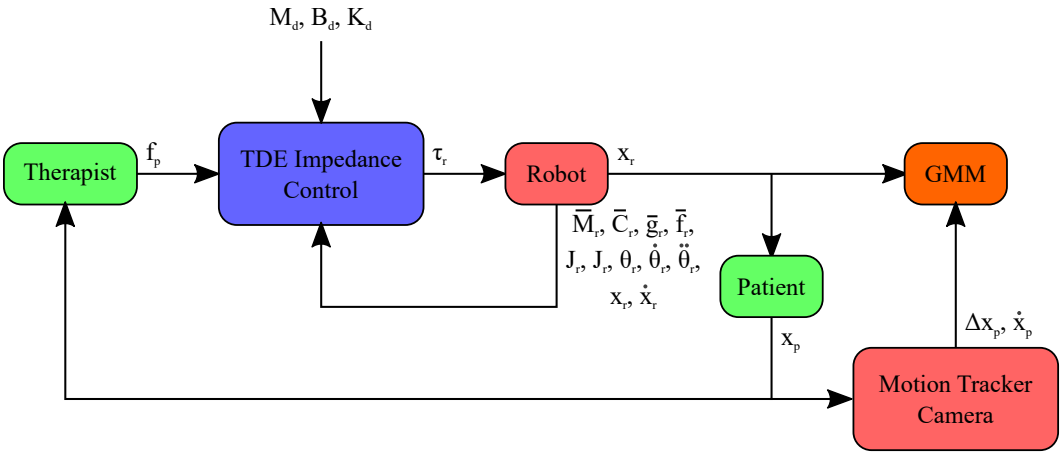
and future directions for the work.

4.2 Proposed Approach

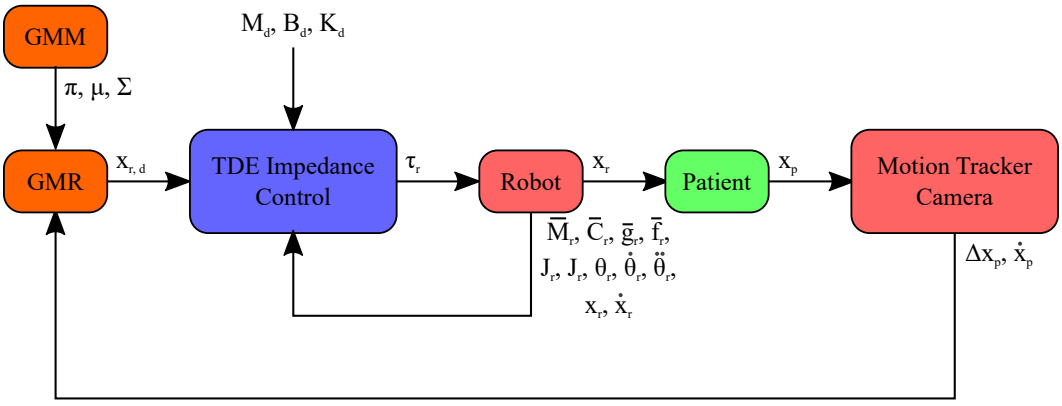
We aim to produce a robotic system that provides assistance during gait therapy by partially lifting a patient's affected leg during the swing phase of their stride. The robot should learn the amount and timing of assistance to give by first observing a therapist providing assistance to a patient as in during typical BWSTT, i.e., adjusting the trajectories of the patient's foot, knee, or hip. The therapist should provide assistance by physically holding and moving the robot, which is attached to the patient. Due to the nature of this task, we consider the patient's affected foot (and more specifically their toe) position to be the measurable outcome. This means that during the therapist's demonstration, the motion of the patient's foot and the positions the therapist moves the robot to should be recorded and a model generated to relate the two. Then, during the phase where the therapist is no longer present, which we call the imitation phase, the robot should be able to provide similar assistance by partially lifting the patient in a safe and non-disruptive way. We then need two main components to control the robot. First, since the therapist should be able to move the robot by applying force to its end-effector when demonstrating assistance, a robot control scheme designed for safe physical human-robot interaction (PHRI) should be incorporated. The Time Delay Estimation (TDE) impedance control method is selected for this (explained in Section 4.2.1). Second, an LfD algorithm that can be easily used to encode the demonstrated trajectory data is preferable. We elect to utilize Gaussian Mixture Model (GMM) and Gaussian Mixture Regression (GMR) based techniques (explained in Section 4.2.2). An overview of the system is provided in Fig. 4.1.

4.2.1 Impedance Control for PHRI

An impedance control scheme is selected to allow the therapist and patient to safely interact with the robot. Impedance controllers produce a desired



(a)



(b)

Figure 4.1: High-level block diagrams of data flow and interaction between different agents (i.e., the therapist, patient, robot, and motion tracker camera). (a) shows the process flow when the therapist and patient interact to provide training demonstrations and (b) shows the process flow when the patient is practicing alone with assistance from the robot.

force based on a predefined relationship with the robot's motion, often described in terms of mechanical inertia, damping, and stiffness. Since our task involves lifting a potentially substantial portion of the patient's body weight, a more heavy-duty robot than those typically seen in rehabilitation robotics is required. However, internal gearing in the joints of such robots produces an apparent inertia of $n^2 I$, n representing the gear ratio, meaning the robot is typically impossible to move passively. The use of impedance control addresses this issue for larger geared robots, making it easy for a user to move the controlled robot. Note that an admittance controller, an alternative force control method that produces a desired motion depending on force input, could have been employed. However, admittance controllers typically present instability when in contact with environments with high impedances, such as a human gripping and holding a robot in place. On the other hand, impedance control is ideal for maintaining safety in robotic control under environmental contact (e.g., during PHRI), but requires the dynamics of the robot to be well modelled [92]. In our scenario, the robot dynamics can be written as

$$M_r\left(\theta_s\right)\ddot{\theta}_r + C_r\left(\theta_r, \dot{\theta}_r\right)\dot{\theta}_r + g_r\left(\theta\right) + f_r\left(\theta_r, \dot{\theta}_r\right) - J_r f_p = \tau_r \tag{4.1}$$

where θ_r represents the robot joint angles, M_r the moment of inertia matrix, C_r the Coriolis and centrifugal matrix, g_r the gravity vector, J_r the robot's Jacobian, f_r the robot's joint friction vector, f_p the force exerted by the patient on the robot end-effector, and τ_r the controller motor torque. Please note that the dependence on θ_r will be dropped for brevity. The non-linear terms M_r, C_r, g_r and f_r can be roughly modelled, but will likely be inaccurate leading to potentially undesired dynamics.

The Time Delay Estimation (TDE) method, as presented in [93, 94], is used here to reduce the inaccuracy when estimating these non-linear terms. Our approximate model gives us the nominal values $\left\{\bar{M}_r, \bar{C}_r, \bar{g}_r, \bar{f}_r\right\}$. We can then rewrite (4.1) as

$$\bar{M}_r\ddot{\theta}_r + \left(M_r - \bar{M}_r\right)\ddot{\theta}_r + \ldots + \bar{f}_r + \left(f_r - \bar{f}_r\right) - J_r f_p = \tau_r \tag{4.2}$$

which separates the nominal dynamics values from the unknown model errors for each non-linear term. The uncertain non-linear terms can then be grouped

together:

$$N = \left(M_r - \bar{M}_r\right)\ddot{\theta}_r + \ldots + \left(f_r - \bar{f}_r\right) \tag{4.3}$$

The TDE method approximates the non-linear terms N at a time t, e.g., $N(t)$, by equating them to the previously measured torque values at a time $t - T$, provided T is small:

$$N(t) \approx N(t - T) = \tilde{N} = \tilde{\tau}_r + \tilde{J}_r\tilde{f}_p - \tilde{\bar{M}}_r\tilde{\ddot{\theta}}_r - \ldots - \tilde{\bar{f}}_r \tag{4.4}$$

where the tilde symbol indicates time delay measured values. The desired impedance dynamics are given as

$$M_d\left(\ddot{x}_r - \ddot{x}_{r,d}\right) + B_d\left(\dot{x}_r - \dot{x}_{r,d}\right) + K_d\left(x_r - x_{r,d}\right) = f_d \tag{4.5}$$

where M_d, B_d, and K_d represent the desired mass, damping, and stiffness impedance parameters, x_r represents the robot's Cartesian end-effector position, and f_d represents the desired output force. By equating f_d to f_p and using the relationship between Cartesian and joint space acceleration

$$\ddot{x}_r = J_r\ddot{\theta}_r + \dot{J}_r\dot{\theta}_r \tag{4.6}$$

we can combine (4.2), (4.4), and (4.5) in order to express the desired robot joint torque controller as

$$\begin{aligned}\tau_r = \bar{M}_r J_r^{-1}\left\{\ddot{x}_{r,d} - M_d^{-1}\left[B_d\left(\dot{x}_r - \dot{x}_{r,d}\right) + K_d\left(x_r - x_{r,d}\right) - f_p\right] - \dot{J}_r\dot{\theta}_r\right\} \\ + \bar{M}_r\ddot{\theta}_r + \bar{C}_r\dot{\theta}_r + \bar{g}_r + \bar{f}_r + \tilde{N} - J_r f_p\end{aligned} \tag{4.7}$$

which effectively provides interactions in Cartesian space. For more details on this process, readers are encouraged to see [93].

4.2.2 Gaussian Mixture Model and Regression

As stated, the basis of the LfD paradigm lies in the incorporation of two phases: the demonstration phase, where the robot observes and statistically encodes trajectories that are physically demonstrated to it, and the imitation phase, where the robot performs regression on the model generated in the demonstration phase. The choice of algorithm for the trajectory encoding is a

widely researched topic. We choose to incorporate a GMM based approach for our learning algorithm in a similar manner to Chapter 3 (and so as to make use of the same advantages discussed there).

As a reminder, the formulaic expression for a GMM is given as

$$p(\xi) = \sum_{k=1}^{N_k} p(k)\, p(\xi|k)$$

with a total of N_k Gaussian components in the model, $p(k)$ being the priors, $p(\xi|k)$ being the conditional density functions, and ξ being a D-dimensional data vector containing both the input and output variables needed during regression. $p(k)$ and $p(\xi|k)$ are computed as functions of the model variables $\{\pi_k, \mu_k, \Sigma_k\}$, which represent the prior probabilities, mean vectors, and covariance matrices that define each Gaussian component. Further details can be found in [95]. In our experiments, we opt for a simpler characterization of the patient's gait cycle than what is typically seen in other gait therapy works; we record only the difference in the toe positions of each foot and the velocity of the unimpaired foot, as opposed to the joint rotations of the full leg. The data vector is then given by $\xi = [\Delta x_p, \dot{x}_{p,u}, x_r]^T$, with Δx_p representing the difference in patient foot position (e.g., $\Delta x_p = x_{p,right} - x_{p,left}$), $\dot{x}_{p,u}$ representing the patient's unaffected foot's velocity, and x_r representing the robot's end-effector position.

We then incorporate GMR during the imitation phase in order to extract the desired therapeutic behavior of the robot from our learned model. GMR leverages the Gaussian conditioning theorem and linear combination properties of Gaussian distributions to retrieve the mean output values ($\hat{\xi}_s$), referred to as the conditional expectation, from a GMM, as well as the variances in those values, referred to as the conditional covariance ($\hat{\Sigma}_s$). Again, further details can be found in [95].

Fig. 4.2 depicts the LfD procedure as described.

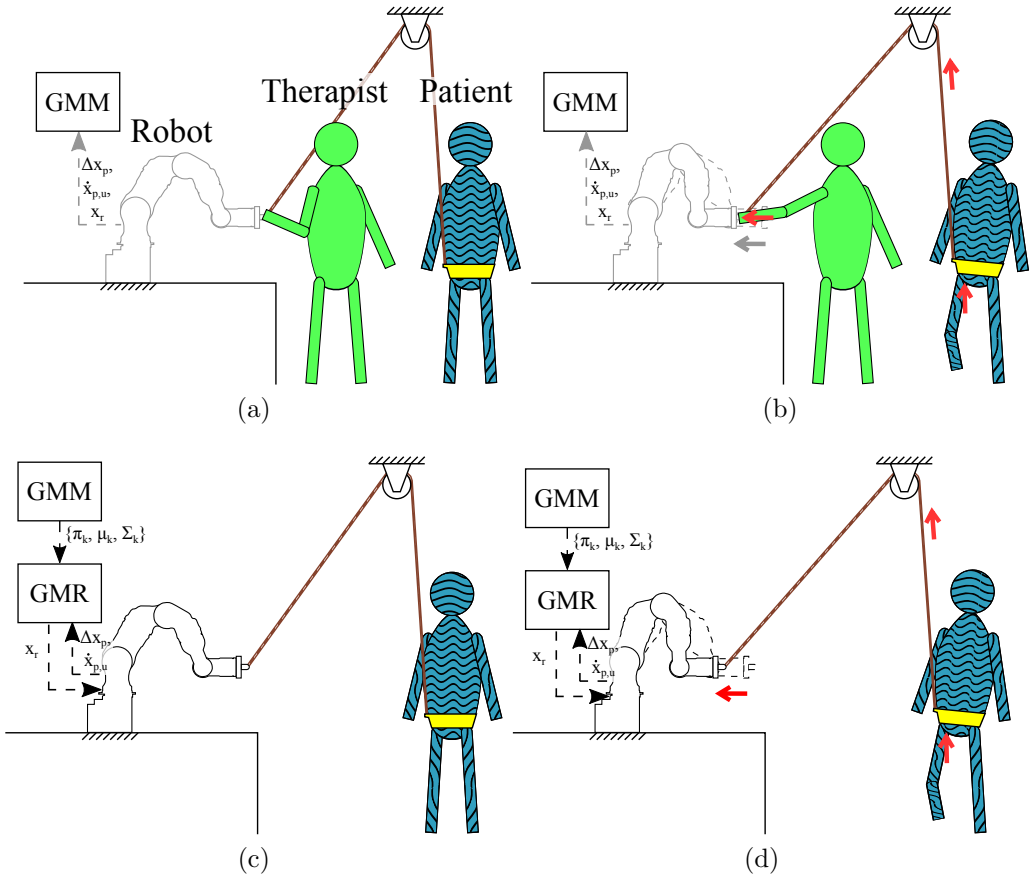


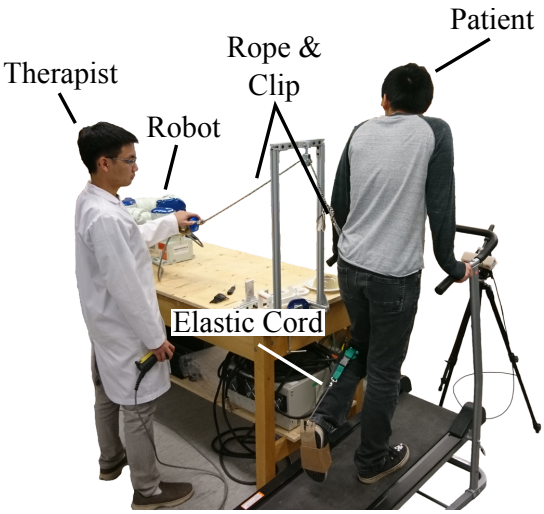
Figure 4.2: A generalized diagram of the LfD procedure employed in this work. In (a), the therapist and participant cooperatively interact while completing the walking task, with the therapist providing assistance by moving the robot as shown in (b). The learning system then learns the therapist's behavior from the provided demonstrations in phases (a) and (b), characterized as desired positions for the robot. Then, in diagrams (c) and (d), the robot replicates the learned behavior, allowing the participant to practice the gait therapy task in the therapist's absence while experiencing the therapist's assistance.

4.3 Evaluation

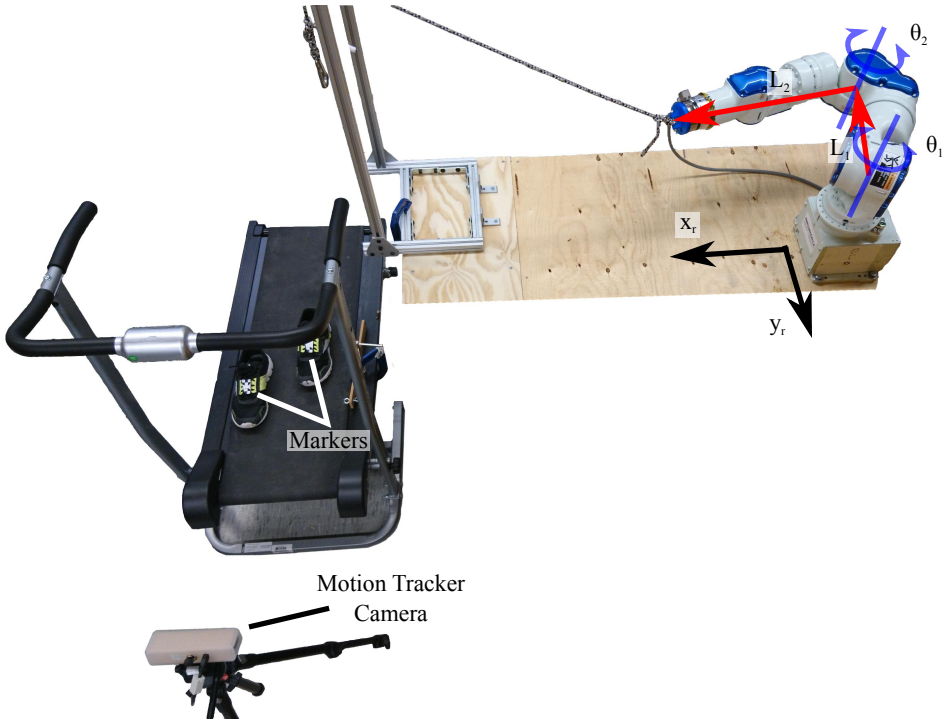
4.3.1 Experimental Setup

We evaluated the system with a standard locomotion task, where two able-bodied study participants (male, 23 years old, and male, 24 years old) walked on a manually powered treadmill. They wore an elastic cord attached between their heel and calf that emulated foot drop during locomotion. The elastic cord is stiff enough to ensure the toe fully drops during a step. A ClaroNav MicronTracker (ClaroNav, Inc., Toronto, Ontario, Canada) motion tracking camera was used to record the positions of both of the patient's feet. The participant wore a waist-level harness attached to a rehab robot by a rope and pulley system. The hip is chosen as the attachment site for simplicity, as it moves the least in the horizontal plane during gait. A Motoman SIA-5F (Yaskawa America, Inc., Miamisburg, Ohio, USA) seven DOF serial manipulator was used as the rehab robot, with a 6-DOF ATI Gamma Net force and torque sensor (ATI Industrial Automation, Inc., Apex, North Carolina, USA) attached at the robot's wrist joint before the end-effector. The robot was simplified to a 2-DOF RR planar robot, which moved in 1-DOF such that the waist harness was hoisted linearly upwards by the pulley. The therapist was represented by a third able-bodied individual (Fig. 4.3). Two additional participants were also tested, but due to the poor quality of their motion tracker data, these results were not included.

The impedance parameters in (4.5) were chosen experimentally. For demonstrations, M_d and K_d were given values permissive to free movement, while the damping parameter B_d was given a higher value and decreased until instability was observed. For imitations, K_d was adjusted to instead provide accurate trajectory tracking. Final values for the parameters were given as $M_d = 4.94$ N·s^2/m, $B_d = 80.52$ N·s/m, and $K_d = 0$ for demonstrations. For imitations, M_d and B_d were unchanged and $K_d = 311.29$ N/m, around a third of a value based on a previous study on measuring the impedance of a stiff upper arm (911.29 N/m) performed by our group. Desired accelerations and velocities were zero at all times, and desired positions were provided by regression dur-



(a)



(b)

Figure 4.3: Experiment setup. In (a) the robot is moved by the therapist by holding and pressing on its end-effector force sensor. This provides a lifting assistance to the patient who walks at their selected pace on the treadmill while harnessed to the robot through the rope and clip. During both the demonstration and imitation phases, the patient, played by a healthy participant, wears the elastic cord in order to simulate foot drop. In (b) the motion tracker camera is shown placed in front of the patient so as to capture the positions of their toes, which are registered to markers placed on the tops of their shoes. The simplified 2-DOF kinematics of the robot are also shown.

ing imitation. Estimates of the robot dynamics model nominal parameters $\bar{M}_r$ and $\bar{C}_r$ in (4.2) were performed as in [96] for a 2-DOF robot, while $\bar{g}_r$ and $\bar{f}_r$ are estimated to be negligible ($\bar{g}_r = 0$ as the robot is positioned in a gravity neutral orientation, i.e., where the joints of the robot that are controlled by the impedance controller are oriented to rotate in the horizontal plane only and are thereby not under the effects of gravity). The remaining joints of the robot were held static with a PID controller.

4.3.2 Results

Five demonstrations were first performed to train the system. While wearing the elastic cord on the left foot (representing the limb with foot drop), the participant was assisted by the therapist and completed 10 gait cycles per demonstration. A GMM of nine components ($N_k = 9$) was generated from these demonstrations. This selection was partially motivated by considering the common interpretation of the gait cycle as having eight phases; model generation was therefore tested for eight or more components. Using the generated model, five imitations were recorded in which the participant completed the same gait task wearing the elastic cord but with the assistance of the robot instead of the therapist. An additional five baseline datasets were recorded without therapist or robot intervention and without the elastic cord as a handicap, providing baseline data of an able-bodied individual for comparison later. These three sets of data are referred to as the "demonstration", "imitation", and "baseline" datasets or scenarios from hereon.

Motion tracker data of each foot and the treadmill surface were used to generate the toe clearance values for the assisted and normal datasets during the gait cycle for each foot. The trajectories were normalized to 5000 samples each to provide consistency in comparison, where the swing phase is found in the first portion of the data and the stance phase in the second (Fig. 4.4). The averages of the maximum toe clearances for each foot in each of the scenarios were also found (Table 4.1).

As we also aimed to examine the similarity between the toe clearance trajectories, the Kolmogorov-Smirnov hypothesis test was performed between the

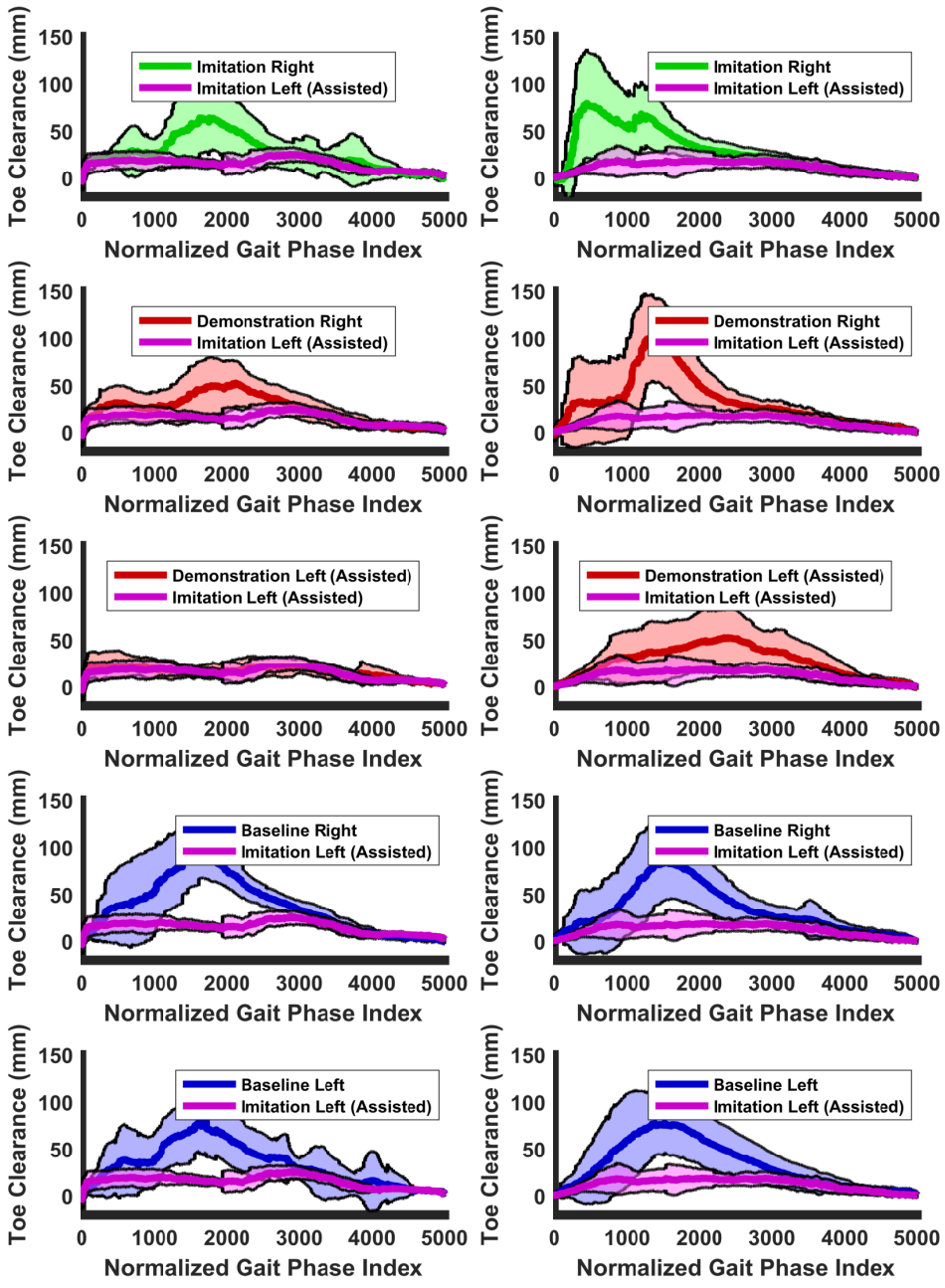


Figure 4.4: Comparison of toe clearance between the assisted left foot during imitation and each of the left and right feet in the other scenarios. Results for Participant 1 are shown in (a) and Participant 2 in (b). Mean values are represented by the solid colored lines, while one standard deviation is shown by the filled area around each trajectory. All trajectories are normalized to 5000 data points using spline interpolation, allowing for later comparison with the Kolmogorov-Smirnov test.

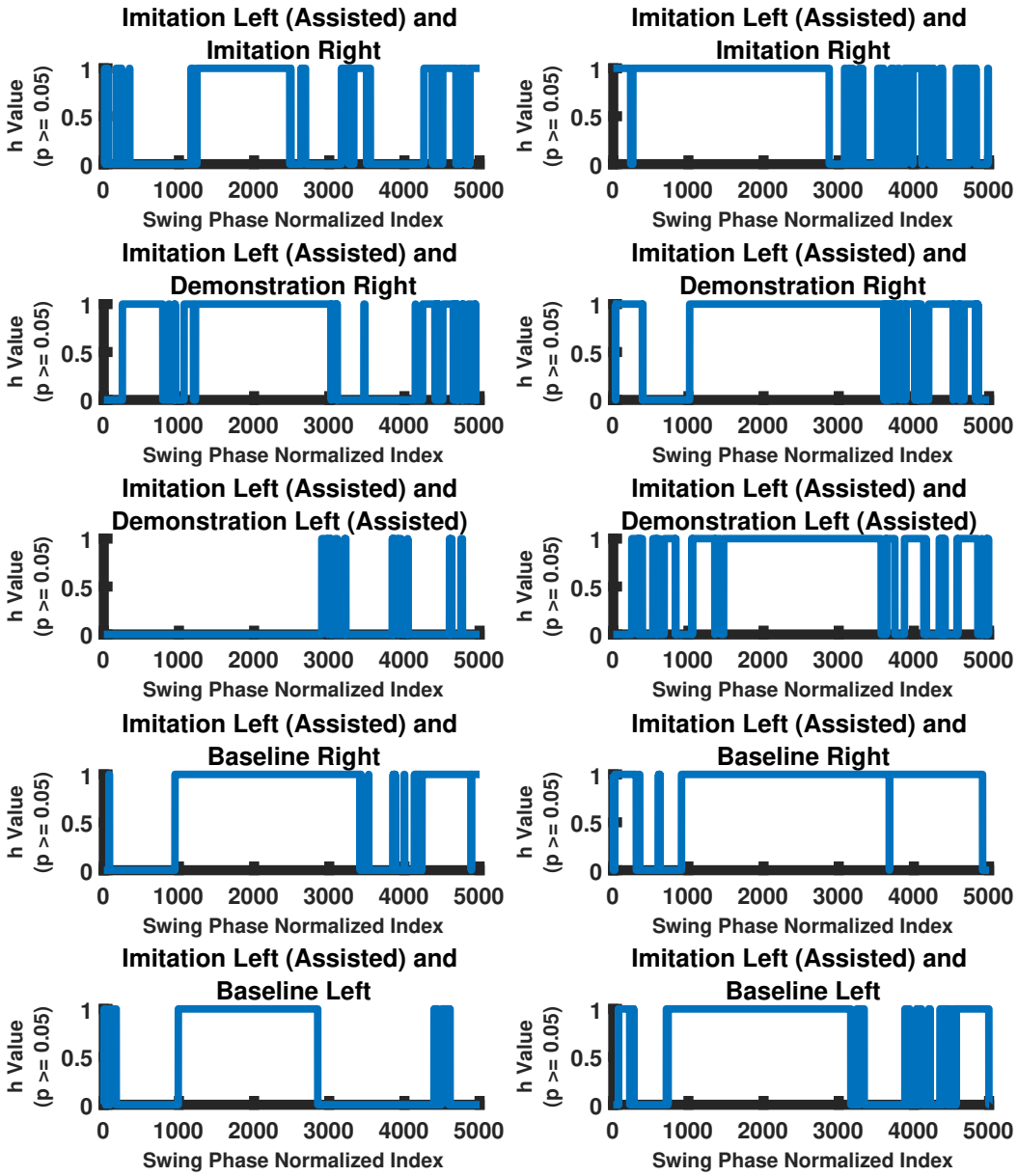
Table 4.1: Mean and standard deviation values of maximum toe clearances.

	Participant 1					
	Imitation		Demonstration		Baseline	
	Right	Left	Right	Left	Right	Left
Mean (mm)	92.65	33.26	78.44	39.12	113.71	106.90
SD (mm)	24.68	9.58	21.19	11.69	23.28	21.25
	Participant 2					
	Imitation		Demonstration		Baseline	
	Right	Left	Right	Left	Right	Left
Mean (mm)	127.97	32.44	140.19	82.36	131.72	118.74
SD (mm)	25.38	13.11	31.09	32.40	22.51	32.38

imitation (assisted) left foot dataset (representing the foot with foot drop) and each of the other datasets. The toe clearances of each scenarios' gait cycles at each normalized index of the trajectory were treated as individual distributions, across which the test was performed (Fig. 4.5).

4.4 Discussion

From Fig. 4.4 we see that the system would indeed be able to assist a patient with foot drop to achieve toe clearance (specifically the imitation (assisted) left foot), and in turn practice gait therapy on his/her own. No sudden, unsafe or unstable movements were experienced, showing that the proper interaction was learned and that the impedance controller functions adequately for the task. However, we see that even at each foot's maximum clearance (Table 4.1), the assisted left foot during imitation (33.26 ± 9.58 mm for Participant 1, 32.44 ± 13.11 mm for Participant 2) does not rise within a standard deviation of any of the other trajectories except for the demonstration (assisted) left foot for Participant 1 (39.12 ± 11.69 mm), motivating further examination of the similarity of the robot-assisted left foot trajectory to the other recorded foot trajectories. The Kolmogorov-Smirnov results confirm this observed dissimilarity; the null hypothesis ($h = 0$, $p > 0.05$), stating that the trajectories are statistically similar, is only confirmed for sparse segments of the gait cycle, nearer to the beginning of the other trajectories.



(a) (b)

Figure 4.5: Kolmogorov-Smirnov test performed for each normalized index. The diagrams depict the statistical similarities between the imitation (assisted) left foot's trajectories and each of the other scenarios. Results for Participant 1 are shown in (a) and Participant 2 in (b). The hypothesis measure (h) represents whether the two are statistically similar: $h = 0$ represents the null hypothesis (i.e., the signals are statistically similar at the point of comparison) and $h = 1$ represents statistical dissimilarity. The imitation (assisted) left foot achieves similar clearance values only much earlier in its swing phase to the other datasets, with the exception of Participant 1's demonstration scenario left foot measurements with which it matches throughout the entire cycle.

A qualitative examination of the toe clearance trajectories provides some insight into possible reasons behind these observations. The trajectories other than for the imitation and demonstration (assisted) left foot datasets distinctly peak around halfway through the swing phase ($50\% - 60\%$ of the swing phase). Both assisted left foot datasets, on the other hand, peak very late in the phase (80% of the swing phase). This phase shift could be attributed to a number of factors. First, it is highly plausible that the demonstration data provided by the therapist's assistance was timed incorrectly, and their intervention was phase shifted with respect to the participant's walking pattern; this is evident in the visual similarities between the demonstration and imitation data for the (assisted) left foot. A second cause could likely have been the stiffness of the robot during the imitation phase was too low ($K_d = 311.29 N/m$). Another possible reason could be that the intermittently applied change in the participant's center of gravity negatively affected his/her gait pattern, leading to a period in the cycle ($20\% - 50\%$ of the swing phase) where the participant resisted the robotic assistance until comfortable. Lastly, the generative properties of GMMs tend to pull the model's components away from curves in trajectories, leading to an observable "corner cutting" effect. When using GMR with the produced model, the desired output could then effectively have a damped behavior as compared to what was demonstrated. These same factors could also produce the reduced maximum toe clearances observed as well. One important caveat is that only toe clearance values were recorded. With the simulated foot drop, the toe is the lowest point of the foot during the swing phase; however, in normal gait, the heel is lowest when preparing for heel strike. It could be beneficial to record the positions of the heel as well, but this would require a more advanced motion capture system to capture the motion from behind the patient.

We suggest a number of possible improvements that could address these shortcomings. First and foremost would be to have experienced, actual rehabilitation practitioners perform the role of the therapist, as their expertise would likely produce improved results in the tracking of an appropriate gait pattern. In addition to this, results could also be improved if the compliance

of the robot during the demonstration phase was increased. The impedance controller is fundamentally limited by the design of the robot; compliance was increased as much as possible through lowering the terms M_d, B_d, and K_d, but M_d is lower bounded by the physical mass of the robot and B_d was required to be non-zero for stability. As a result, moving the robot required the therapist to exert as much force as the body weight percentage they were supporting, on top of the mass and damping the robot presented which could result in less accurate movements. An adaptive impedance control system could be a possible solution, where depending on some measure of sensed therapist intention the robot relaxes or increases its impedance parameters. With regards to the stiffness of the robot during the imitation phase, it may help to increase the value or to also implement an adaptive controller. The value was chosen by using one-third of a value for a stiff arm found in an earlier work by our group, in order to allow for safe and gentle guidance. However, it is likely that the system was too forgiving and thus had difficulty actually lifting the participant properly as is most clearly seen in Participant 2's left foot imitation results not resembling their left foot demonstration results. It would be beneficial to perform a future study focused on how to properly tune the impedance parameters or to implement an adaptive parameter tuning system. With regards to the learning algorithm, modifying Gaussian-based modeling methods to place more emphasis on curves in trajectories, as in [58], could help to address the hypothesized issues arising from the generative nature of the models. Lastly, increasing the participant population would benefit the study. Having more participants with a wider variation in gait patterns, as well as actual symptomatic patients would provide results more likely to be applicable to the intended population.

4.5 Conclusion

In this chapter, impedance control based teaching of a robot was used to teach a therapist's assistance to a robot during a treadmill-based therapy routine for a participant with foot drop. GMM and GMR were used as the learning and

regression algorithms needed to train the robot and have it imitate the therapist later. We show the system is able to successfully provide toe clearance during gait, although the imitation is not quite able to produce clearance values comparable to normal gait. Future work will focus on improving the learning algorithm to account for these inaccuracies, better tuning the impedance controller, and eventually testing the system with more participants and in clinical settings.

Chapter 5

Learning Therapists' Impedance-based Behaviors

5.1 Introduction

Rehabilitation medicine has recently come to focus on practicing functional tasks, also referred to as Activities of Daily Living (ADLs), as a means of facilitating therapy. ADLs take the form of day-to-day tasks such as opening doors, cooking, and dressing to name a few, and are practiced in order to provide gains in neuromuscular coordination that are directly translatable to daily life. Current practice is for therapists to either perform assessments of rehabilitation gains using ADLs in tests, or to perform in-home training for practicing ADLs. Use of ADLs as a focus for stroke rehabilitation has been shown to provide improved independence and quality of life outcomes [22]. This is opposed to the more traditional movement therapy, in which a patient simply moves and exercises their affected limbs. Robots have traditionally been preprogrammed to provide interactions appropriate to predefined tasks, which is sufficient in the simple case of movement therapy. However, ADLs are inherently more complex; the tasks are performed in unstructured environments where task parameters may vary greatly (e.g., the shape of a door handle, the location of the handle, etc.) and full knowledge of the task is unobtainable. As a result, ADL-based therapy has only recently begun to see computer implementation as in [97], and has seen little to no integration with robotics in particular. Despite these limitations, able-bodied humans can

perform ADLs robustly. In order to take advantage of this robustness, rehabilitation robots should be programmed to enable quick redefinition of therapy tasks and the therapeutic behavior by therapists that have minimal programming knowledge. This redefinition happens not by manipulating computer codes but by physically moving the rehabilitation robot as will be explained later.

Learning from Demonstration (LfD) techniques [48] can be implemented to allow for hands-on kinesthetic demonstration-based reprogramming of robots for this purpose. Demonstration refers to the performance of a task which a robotic system observes either indirectly (e.g., through motion capture) or directly (i.e., the robot is moved by the demonstrator through the task trajectory). By statistically encoding behaviors learned through demonstrations, the robot can be programmed intuitively to imitate desired actions such as providing therapeutic forces to patients interacting with the robot. Learning methods making use of Gaussian Mixture Models (GMMs) [95] have become especially prevalent in the field of robotic automation in recent years. These models require relatively few demonstrations for recreation of the demonstrated behavior as opposed to other machine learning methods such as reinforcement learning, and so are ideal for the described scenario [48]. Research in the area of robotic rehabilitation has seen a rise in the incorporation of LfD techniques. LfD techniques have been used to succesfully teach a robot to guide a patient through specified trajectories for ADL training [58, 98]. Authors in [99] present an LfD approach that allows an assistive robot to cooperatively dress a user while being able to adapt to the highly unstructured nature of the task. LfD has also been employed to teach robots to assist a patient in completing a trajectory based specifically on the variance of the therapist's demonstrations [85, 87]. However, there is a lack of literature aiming to learn a therapist's impedance, and using the learned impedance to provide assistance to a patient practicing ADLs.

Providing demonstrations to the learning system itself should also be as intuitive as possible. Kinesthetic demonstration provides such an intuitive method. As a first step, this entails making the robot manipulator as compliant

as possible to an operator's physical input. For smaller lightweight robots, this can often be achieved simply through the operator's input overcoming the inertia and other dynamics of the robot's mechanical components, i.e., motors. For larger robots, however, this is typically impossible as internal gearing and friction make the structure non-backdrivable. Admittance control is a common technique for introducing compliance in such cases [100]. A force sensor is attached to the robot end-effector. Interaction forces sensed at the end-effector cause movements such that a pre-set dynamic relationship between the applied force and the ensued motion holds. This second method will be employed with the setup presented in this work.

We present a proof of concept system for kinesthetic teaching of rehabilitation robots based upon LfD principles. The system is intended to be kinesthetically programmed by users with little to no programming experience, e.g., physiotherapists. Provided with motion and force data from various demonstrations, the system aims to extract a data-driven model of the therapist's behavior (e.g., the levels of assistive/resistive forces) throughout the performance of an ADL task. This will require a set demonstrations involving both the therapist and the patient completing the task (successfully), and another set involving only the patient attempting to complete the task (unsuccessfully). Through this, a so-called "performance differential" [85] may be defined, inherently describing the therapist's behavior. Fig. 5.1 depicts a generalization of the system.

The chapter is organized as follows. Section 5.2 outlines the proposed components involved in the design of the system, and Section 5.3 describes the experimental evaluation and presents the results. Section 5.4 provides discussion of results and finally Section 5.5 offers concluding remarks and comments on future directions for the work.

5.2 Proposed Approach

We aim to produce a system that learns and replicates the impedance-based behavior of a therapist in 3-dimensional space, where a robot and two humans

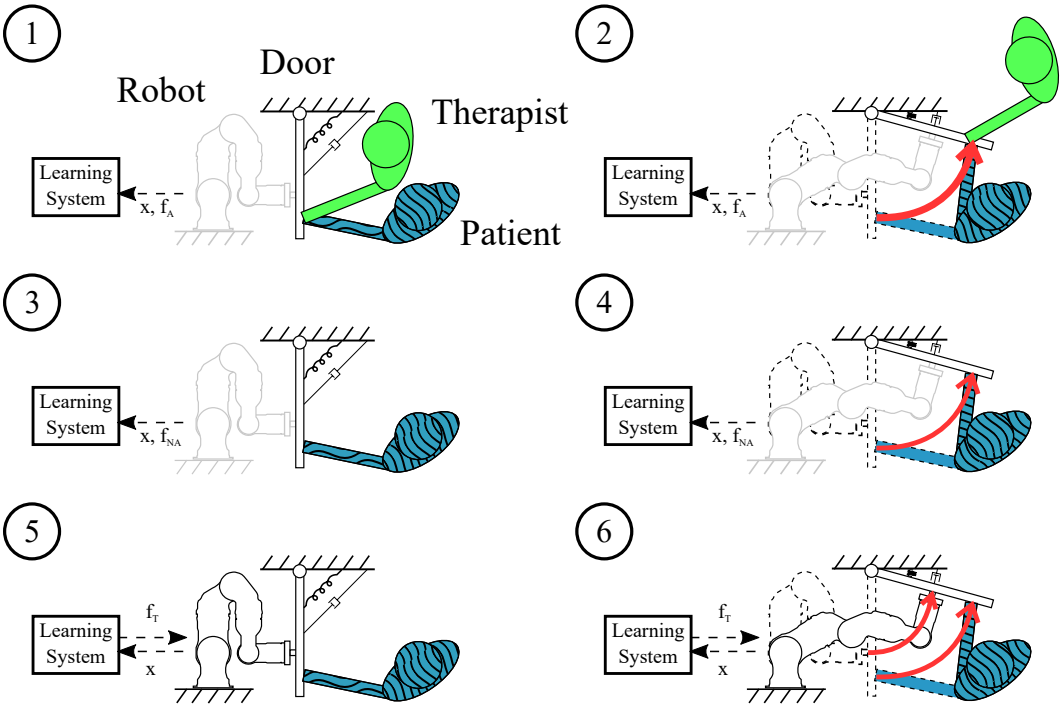


Figure 5.1: A generalized diagram for the LfD procedure employed in this work, where in this example the participants open a (self-closing) door. In diagram 1, the therapist and patient cooperatively interact while completing the task, moving the door to the position shown in diagram 2. In diagrams 3 and 4, the patient attempts to complete the task on their own. In phases 1-4, the robot is compliant and only passively observes and records the demonstrations. The learning system then learns the therapist's behavior from the provided demonstrations in phases 1-4. Then, in diagrams 5 and 6, the robot replicates the learnt behavior, allowing the patient to practice the therapy task in the therapist's absence while experiencing the therapist's interactions.

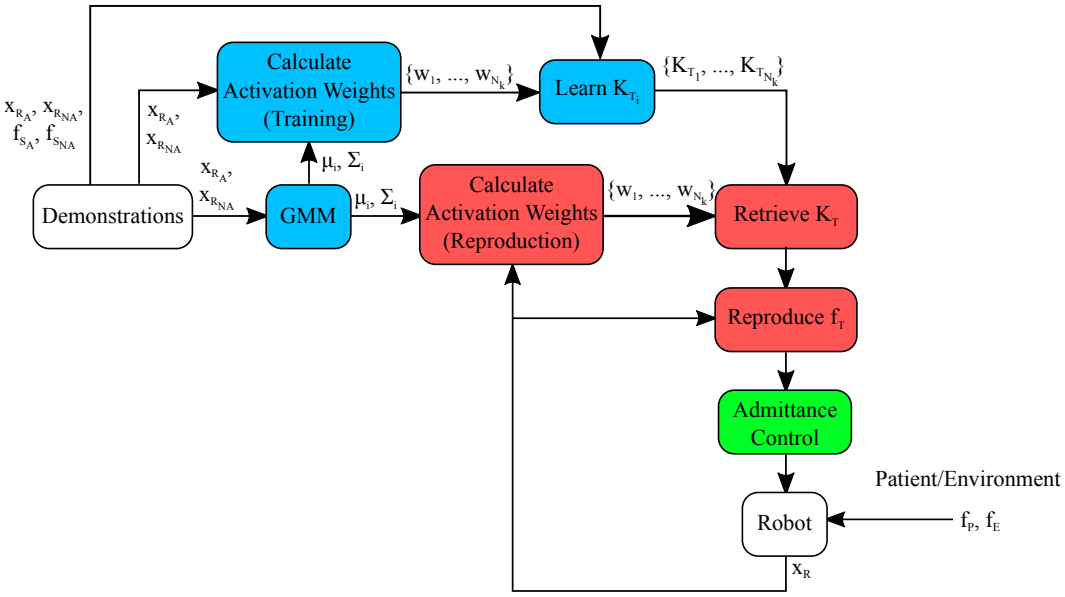


Figure 5.2: Process for reproducing the therapist's behavior learned through demonstrations. Demonstrations are provided to train the learning system (in blue). Then, in reproductions, the patient and task environment exert forces on the robot's force sensor. The admittance controller (in green) causes changes in the robot's end-effector position according to the measured forces. Reproduction of the therapist's behavior (in red), which in this scenario is an applied force, is determined using position feedback from the robot and the learned model.

(i.e., a therapist and a patient) will perform a collaborative task. The robot manipulator acts as a separate agent interacting with the task, much like how the therapist and the patient will contact the task environment. We fix the robot end-effector having an attached wrist force sensor to the task. We need three components: an admittance controller for making the robot compliant, an algorithm for learning the task trajectory, and an algorithm for learning and reproducing the therapist's impedance-based behaviors. Fig. 5.2 provides an overview of the system.

5.2.1 Admittance Control Scheme

A simple admittance control scheme is used to introduce compliance to the robot. Admittance controllers produce a desired displacement based on a predefined relationship with sensed forces. In implementation, this takes the

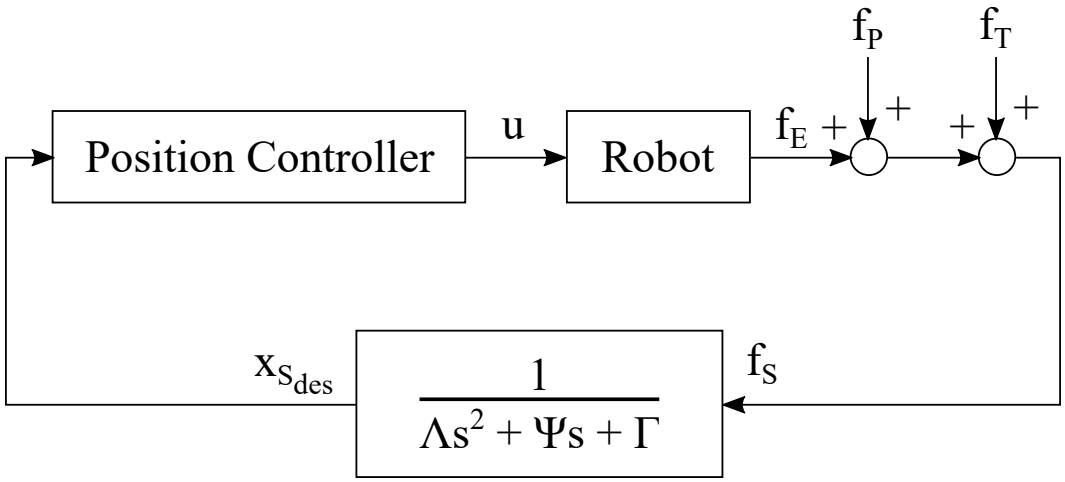


Figure 5.3: Admittance control block diagram. The force measured by the force sensor f_S produces the desired displacement $x_{S_{des}}$ through the control law in (5.1). A position controller produces the robot control torques u using the displacement.

form of the transfer function

$$G(s) = \frac{x_{S_{des}}(s)}{f_S(s)} = \frac{1}{\Lambda s^2 + \Psi s + \Gamma}$$

where f_S is the force exerted on the sensor, $x_{S_{des}}$ is the desired displacement of the robot, and Λ, Ψ, and Γ represent the inertia, damping, and stiffness constants, respectively. The control law is given by

$$f_S = \Lambda \ddot{x}_{S_{des}} + \Psi \dot{x}_{S_{des}} + \Gamma x_{S_{des}} \tag{5.1}$$

Fig. 5.3 provides a schematic of the admittance control loop. The admittance control adds the displacement $x_{S_{des}}$ calculated from (5.1) to the robot's current position. As the patient and therapist exert forces on the robot end-effector, the forces measured by the sensor can be expressed as

$$f_S = f_E + f_P + f_T \tag{5.2}$$

where f_E is the force presented by the task environment, f_P is the force exerted by the patient, and f_T is the therapeutic force exerted by the therapist on the robot end-effector.

5.2.2 Algorithm for Task Trajectory Encoding

LfD is employed in the system to generalize and learn the spatial movements necessary to complete the task. LfD typically involves two separate phases: a demonstration phase where trajectories are learned and statistically encoded, and a reproduction phase where the system performs regression using the generated model to provide a rendition of the earlier demonstrated behavior. A GMM is trained using all demonstrations, providing a probabilistic representation of the motion required to complete the task. The implementation of the GMM is similar again to Chapters 3 and 4; the expression for a GMM is given as

$$p\left(\xi\right) = \sum_{k=1}^{N_k} p\left(k\right) p\left(\xi|k\right)$$

with a total of N_k Gaussian components in the model, $p\left(k\right)$ being the priors, $p\left(\xi|k\right)$ being the conditional density functions, and ξ being a D-dimensional data vector. In this work, $\xi = x_R = [x, y, z]^T$, is the position of the robot end-effector expressed in the robot's base frame. The parameters $p\left(k\right)$ and $p\left(\xi|k\right)$ are computed through the use of each Gaussian's parameters $\{\pi_k, \mu_k, \Sigma_k\}$, representing the prior probabilities, mean vectors, and covariance matrices, respectively. For details, see [95].

The Gaussian parameters are trained using the Expectation-Maximization algorithm, iterating the parameters until the convergence of an optimization measure (typically the log-likelihood) is achieved. The E-step is of particular interest, where the likelihood or activation weight of each i^{th} Gaussian is computed for each data point ξ as follows:

$$w_i = \frac{\pi_i \mathcal{N}\left(\xi|\mu_i, \Sigma_i\right)}{\sum_k^{N_k} \pi_k \mathcal{N}\left(\xi|\mu_k, \Sigma_k\right)} \tag{5.3}$$

5.2.3 Algorithm for Encoding Therapist's Impedance-based Behavior

We propose that during performance of a task, the interaction forces exerted on the robot end-effector by each of the agents (task environment, patient, therapist) can be simplified as a set of spring forces, linearized about points

of the demonstration. We then rewrite (5.2) as

$$\begin{aligned} f_S &= f_E + f_P + f_T \\ &= (K_E + K_P + K_T)(x_f - x_R) \\ &= (K'_E + K_T)(x_f - x_R) \end{aligned} \tag{5.4}$$

where x_f is the approximate position of the task goal point (taken as the average of the demonstration endpoints), and K_E, K_P, and K_T represent the stiffnesses of the linearized task environment, patient, and therapist, respectively. As the system aims to learn specifically the therapist's force, we combine the spring constants for the patient and the task environment forces into K'_E, i.e., $K'_E = K_E + K_P$.

Demonstrations will be recorded for two cases: when the task is performed by the therapist and the patient together (assisted), and when the patient attempts to perform the task by themselves (not assisted). We define the spring constants linearized from the data associated with these cases as $K_A = K'_E + K_T$ and $K_{NA} = K'_E$, respectively. The spring constant for the therapist's force can then be estimated from the difference between the assisted and unassisted spring constants as follows

$$K_T = K_A - K_{NA} \tag{5.5}$$

Estimation of the linearized spring constants will be performed in a manner similar to [101] and [102]. Weighted Least Squares (WLS) estimation is used to compute the stiffness constant associated with each Gaussian component $K_i = \left[\left(X^T W_i X\right)^{-1} X^T W_i F_S\right]$, where by concatenating all N datapoints from every demonstration together, we have $X = [(x_f - x_{R_1}), \ldots, (x_f - x_{R_N})]^T$, $W_i = \text{diag}([w_{i_1}, w_{i_2}, \ldots, w_{i_N}])$ as calculated in (5.3), and $F_S = [f_{S_1}, f_{S_2}, \ldots, f_{S_N}]^T$. In this work, we assume no correlation exists between forces and positions across different Cartesian axes. As a result, the WLS estimation is performed for each axis and all spring constant matrices simplify to $K_i = \left[K_{i_x}, K_{i_y}, K_{i_z}\right]^T$. K_A and K_{NA} are estimated for each Gaussian component in this way, where position and force data from the assisted demonstrations are used in X and F_S to calculate K_A, and from the unassisted demonstrations to calculate K_{NA}.

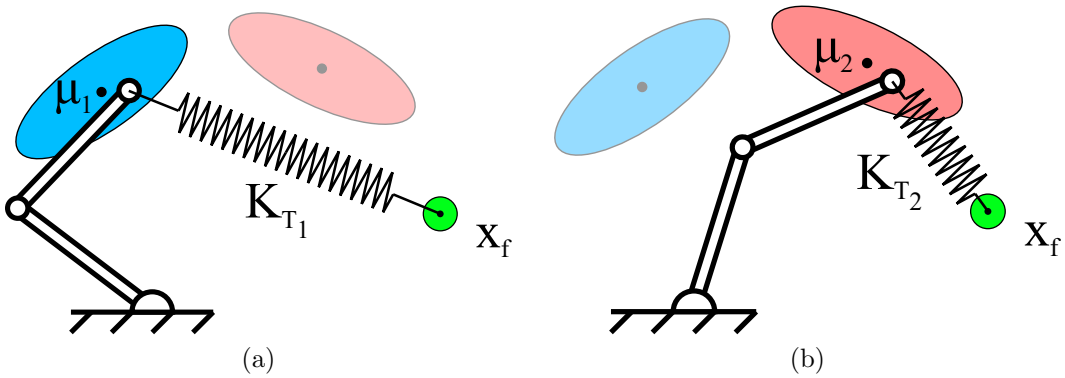


Figure 5.4: Simplified diagram of position-based impedance retrieval for reproduction of the therapist's behavior. In (a), activation weights for the first Gaussian component (colored blue) are highest when the robot is in close proximity to the component. A stiffness constant is retrieved for the corresponding Gaussian and used to generate the forces learned from the therapist. In (b), a different stiffness constant is used when the patient progresses into the spatial coordinates associated with a different Gaussian component (colored red). In actual reproduction, the retrieved stiffness constant may be a mixture of the learned stiffness constants influenced by multiple components, instead of a single constant from the influence of a single component as shown here.

Then, the estimated K_T for each Gaussian component is taken as the difference between K_A and K_{NA} as in (5.5).

In the reproduction phase, the estimated therapist force applied by the robot is given as

$$f_T = \sum_{i=1}^{N_k} w_i \left[K_{T_i} \left(x_f - x_R \right) \right] \tag{5.6}$$

where K_{T_i} is the therapist's stiffness associated with each i^{th} Gaussian component, calculated as described previously. The robot's position x_R is used to calculate the weights w_i of each component. The applied force f_T is then given by the mixture of the spring forces from all of the components according to the robot's distance from the target point. Fig. 5.4 depicts this concept.

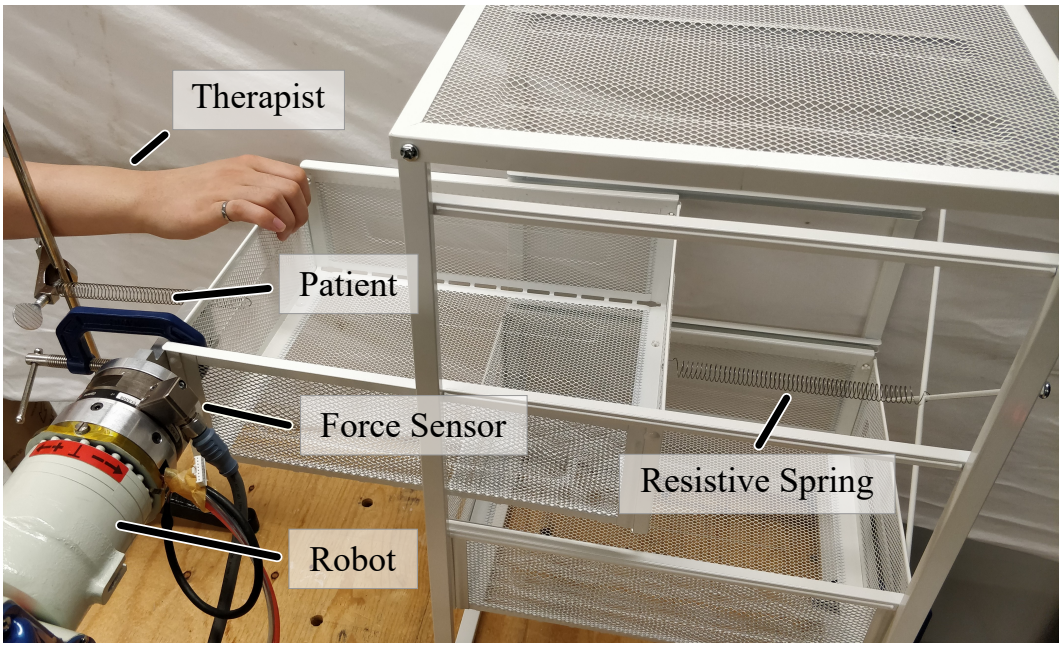


Figure 5.5: Experiment setup.

5.3 Evaluation

5.3.1 Experimental Setup

We evaluate the system on a simple cooperative task where the participants open a drawer fully. The drawer has a spring attached to its back, which resists the opening movement and tends to return the drawer to its closed position. In the experiments, the patient is emulated by a spring attached to the front of the drawer, which tends to open the drawer. The constants of the resistive and patient springs are equal, but the springs come to relaxation at a point before the drawer is fully opened. This means the emulated patient cannot complete the task alone. A Motoman SIA-5F (Yaskawa America, Inc., Miamisburg, Ohio, USA) seven Degrees-of-Freedom (DOF) serial manipulator is used as the rehab robot, with a 6-DOF ATI Gamma Net force and torque sensor (ATI Industrial Automation, Inc., Apex, North Carolina, USA) attached at the wrist joint before the end-effector. The therapist is represented by an able-bodied participant. Fig. 5.5 shows the experimental setup.

The admittance parameters in (5.1) were chosen experimentally, where Λ and Γ were given values permissive to free movement, while the damping

parameter Ψ was given a higher value and decreased until instability was observed. Final values for admittance parameters are given as $\Lambda = 0$, $\Psi = 5$ N·s/mm, and $\Gamma = 0.1$ N/mm.

Three demonstrations are performed for each of the assisted and unassisted scenarios, providing six demonstrations in total. In the assisted scenario, the user representing the therapist provides assistance when the motion of the patient alone begins to slow. In the unassisted scenario, the spring representing the patient is allowed to pull the drawer to equilibrium. To acquire more data, the drawer is moved to its fully extended position and released, moving against the patient back to equilibrium. Complete force profiles for the full motion trajectory are generated for the unassisted case in this manner.

5.3.2 Results

A model of 12 components ($N_k = 12$) was generated from the provided demonstrations. This selection is partially motivated by the biomechanics of human reaching movements; a person first accelerates their hand, travels towards the goal at velocity, and finally decelerates to accurately end their movement. As such, we choose the number of components to be a multiple of three. Fig. 5.6 shows the generated model against the training data. The learned model is then used to estimate the spring constant values K_{T_i} as described in Section 5.2.3.

The system is then evaluated in the real-world experimental setup. The drawer is released from its initial closed state with only the patient-emulating spring and the rehab robot acting against the resistive spring to open the drawer. Since the motions and forces associated with the task are almost completely in the y-axis, only those results are shown hereon. Resulting trajectory and force data is captured in Fig. 5.7.

We compare the resulting net forces from the reproduction to the mean of those obtained during demonstrations. The force profiles are arranged against their respective position profiles, shown in Fig. 5.8. Note that for plotting purposes, the net reproduction force data is calculated by summing the model's assistive force f_T, obtained from (5.6), with the sensor's perceived force f_S.

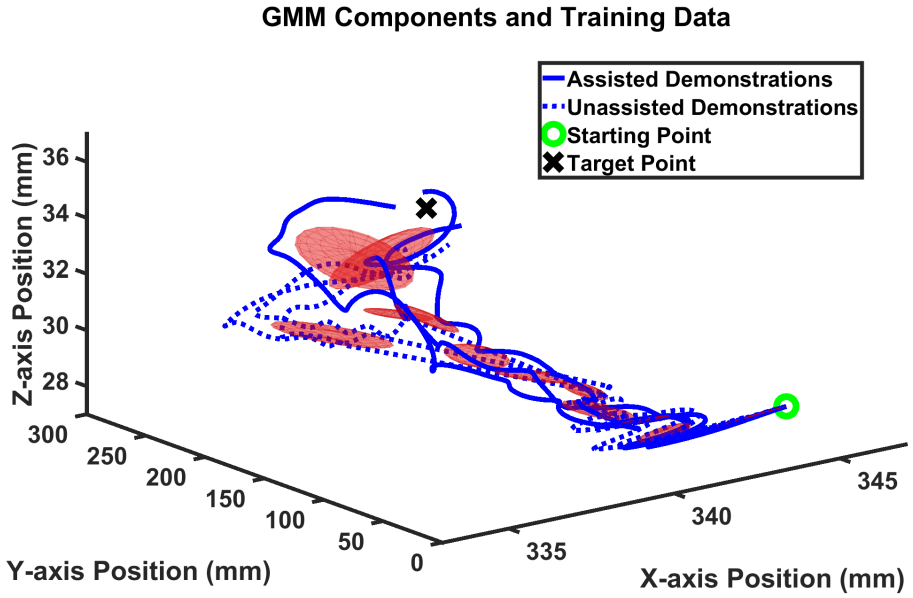


Figure 5.6: Decomposition of motion trajectory into $N_k = 12$ components using a GMM.

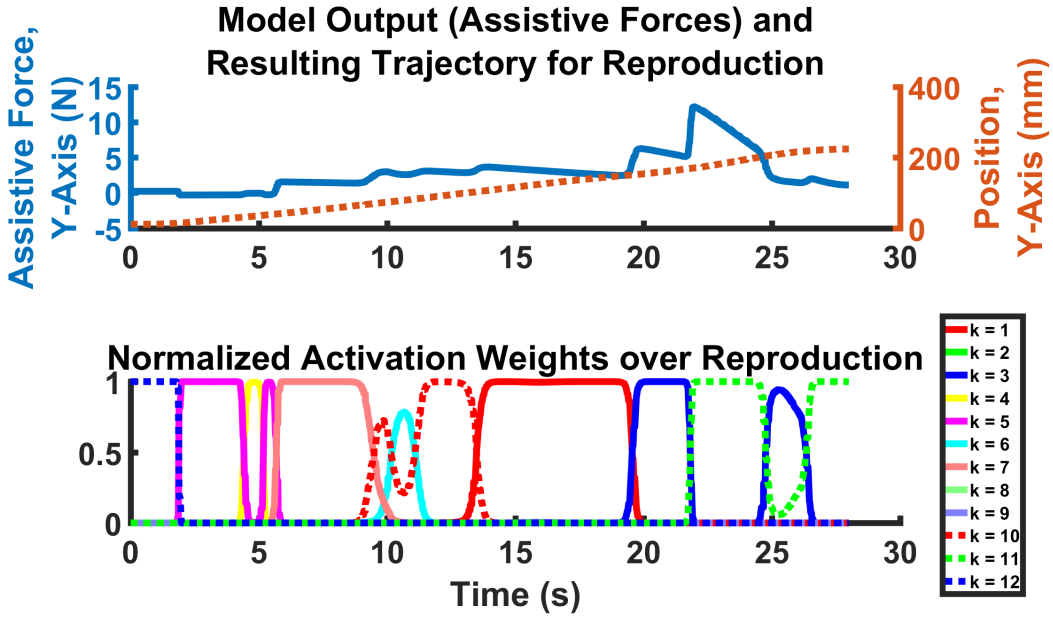


Figure 5.7: Reproduced assistive force output and component weights for a patient-only reenactment of the task. The model outputs a noticeably large force for the $k = 11$ Gaussian component.

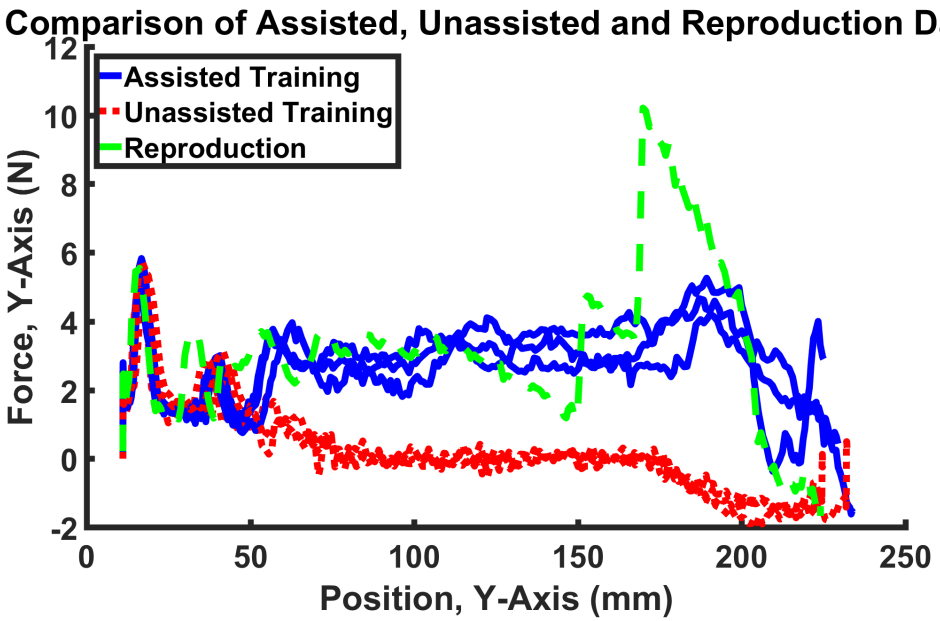


Figure 5.8: Net force profile comparisons across reproduction and training datasets. The unassisted case force data is in dotted red, the therapist-assisted case force data is in solid blue, and the robot-assisted data is in dashed green. The model force output closely matches that of the therapist demonstrations until nearer to the target point; the forces afterwards provide sufficient assistance to complete the task, but are noticeably higher.

Mean absolute error (MAE) for the reproduction is found to be $MAE = 1.8763$ N with the maximum instantaneous absolute error found as $AE_{max} = 8.012$ N. Lastly, the datasets have a correlation coefficient of $\rho = 0.4034$.

5.4 Discussion

Reproduction of the therapist's assistance is performed successfully. When releasing the drawer from its initial closed position, the system is able to command the robot to assist the patient in opening the drawer fully. No sudden, unsafe movements or moments of instability were observed. However, the MAE and correlation appear to indicate that the system produces only a moderately accurate reproduction of the therapist's assistive force. The results presented in Fig. 5.8 provide some insight on a possible source of the error. The system reproduces appropriate force output with minimal discrepancy for the majority of the period in which there is therapist intervention, roughly between $y = 50$

mm and $y = 170$ mm. However, forces after this point quickly diverge from the real therapist's and are responsible for the high $\mathrm{AE}_{\max}$. Relating this to Fig. 5.7 identifies the Gaussian component $k = 11$ as potentially problematic. This is likely because the demonstration data have greater variance near the end of their trajectories, as seen in Fig. 5.6. Gaussian $k = 11$, which is nearest to the target point, must cover a larger spatial volume than most of the other Gaussians. However, the linearization of the assistive stiffness constants may be too general as a result. A finer resolution is needed for the model, but simply adding more Gaussian components may be an impractical solution as it increases model complexity and computation time. The EM algorithm may also place the additional Gaussian components away from that portion of the trajectory even with a larger number of components. A possible solution would be to restructure the task in order to take advantage of more sophisticated Gaussian modeling methods such as the Stable Estimator of Dynamical Systems (SEDS), which provides a global-asymptotically stable task model [66].

With regards to the compliant nature of the system, the results are satisfactory. The system is kinesthetically movable, but not transparent to an ideal degree since the damping coefficient is high. Implementing an adaptive admittance controller, such as in [103] or [104] is a possible solution where the controllers adapt the admittance parameters online in response to parameters like the force tracking error or signal energy. Alternatively, an impedance controller can be used with torque control. The robot dynamics would be required however, which are not readily available in this case.

5.5 Conclusion

In this chapter, kinesthetic teaching of a robot was used for learning a therapist's behavior from recorded demonstrations. GMMs were used as the basis for learning the movements necessary for completing an activity of daily living task, and an assistive force was reproduced by the robot based on estimations of the therapist's impedance-based behavior in the therapist's absence. We

show that the system is able to properly reproduce the therapist's behaviors to assist a patient in completing the task. Future work will aim to incorporate improved learning algorithms that better generalize across demonstrations. We will also incorporate assistance-as-needed (AAN) features, in which the robot delivers assistance depending on the patient's performance.

Chapter 6

Robotics vs Telerobotics-mediated Hands-on Teaching for Rehabilitation Robots

6.1 Introduction

Rehabilitation robotics is an attractive solution to address the growing demand for rehabilitation services. The behaviours of existing robotic systems during rehabilitative therapies are typically pre-programmed, which is highly restrictive in the presence of unstructured task environments and given the variation in patients abilities and therapists' approaches. This is in contrast to the flexibility with which a skilled therapist can adjust the parameters of conventional non-robotic therapy based on years of experience. To directly incorporate the therapist's skills in robotic therapy for the purpose of providing patient-specific intervention, we propose the combination of Learning from Demonstration (LfD) algorithms and the therapist's experience. LfD is a paradigm focused on allowing a human user to program a robot through demonstration of desired behaviours, as opposed to explicit computer programming [48]. In general, the behaviours are actions or movements to be later imitated by the robot. The paradigm involves a machine learning algorithm that statistically encodes the demonstrations, and performs regression on the learned model at a later time to imitate the behaviours. Physically

moving the robot in order to teach it is referred to as kinesthetic teaching.

One important question is, in what manner should the therapist and patient interact with each other to best take advantage of the incorporated LfD paradigm? We explore two modalities here; robotics-mediated kinesthetic teaching (RMKT) where the therapist and the patient interact by using a single robot that learns their movements; and telerobotics-mediated kinesthetic teaching (TMKT), where the therapist and the patient interact using two robots, generally a master-slave system with force feedback. We hypothesize two outcomes:

Hypothesis 1: RMKT will allow a therapist to provide more consistent demonstrations of therapy tasks than TMKT. **Hypothesis 2:** RMKT and TMKT can be applied in similar scenarios, allowing for adequate learning and robotic imitation of the demonstrated therapeutic behaviors.

Although we hypothesize that RMKT will provide better demonstrations for LfD algorithms, we would like to show that TMKT, an unexplored concept, is a feasible alternative to RMKT, which has been previously researched and published. Therefore, the main contribution of this chapter is to study and develop the basis to support the feasibility of TMKT for rehabilitation with similar results as in RMKT. This chapter is organized as follows. Section 6.2 discusses previous works and Section 6.3 outlines the two robotic interaction modalities. Section 6.4 describes the impedance control scheme used and Section 6.5 provides a brief introduction to the LfD algorithms incorporated into the robot control system. Section 6.6 describes the experiments performed and their corresponding results are provided and discussed in Section 6.7. Lastly, Section 6.8 leaves off with closing remarks and possible future directions.

6.2 Related work

A selection of our group's previous works follow. In [94], the authors proposed a paradigm called *learn and replay* to build a bilateral telerehabilitation system that encompasses two distinct phases to save the time of a therapist. In the first phase, the system learns the therapist's arm impedance in performing

a task. Later, in the second phase, the system uses the learned impedance to imitate the therapist behaviour in his/her absence. Note that this system does not use LfD because it does not generalize the learned impedance for different scenarios using statistical encoding methods. In another work [85], the authors developed a robot-assisted rehabilitation system for co-operative therapy combining LfD and Assistance-as-Needed strategies. In the demonstration phase, the system learns the therapist's impedance using a statistical encoding algorithm and builds a model of the therapist's behaviour. Later, based on the difference between the patient's performance and the learned therapist's behaviour, the method determines whether to assist the patient in completing the task or not. In [86], the proposed system learned and imitated the therapist's force and motion behaviour using a different encoding algorithm designed to ensure global stability. Lastly, in [105] the authors implemented a neural-network-based system for upper-limb post-stroke motor disabilities. Aside from these articles, some authors such as [106], have proposed a similar system where a user interacts with a robot to complete a cooperative task, while [53, 101] proposed a similar cooperative task interaction using Machine Learning algorithms. However, these works do not explore implementation in the medical field.

RMKT has thus seen extensive implementation in our works. On the other hand, teleoperation systems have not been implemented in the rehabilitation field using LfD, i.e., TMKT. In [94], the potential of teleoperation-based systems are shown; now we aim to expand on our works and create the first TMKT systems. In this work, we will design a fair comparison between the two modalities, with the same therapy task used to record experimental data for. Ideally, we will be able to show that TMKT is as feasible and effective as RMKT.

6.3 Therapist-Patient Interaction Modalities in Robotic Rehabilitation

The first approach we consider is enabling demonstrations through RMKT. RMKT provides an intuitive method for users to teach the robot movements. This entails making the robot manipulator as compliant as possible to an operator's physical input, allowing a user to grasp and move the robot along the desired trajectory. Force or impedance controllers, which use readings from force and torque sensors, facilitate this. Fig. 6.1 depicts this concept.

The second proposed approach involves a TMKT system. In this approach, our focus is on telerehabilitation through a bilateral (haptics-enabled) TMKT system. Haptic feedback provides a human who operates a tele-robot with a sense of touching a virtual/physical environment. This system can simulate the so-called "hand-over-hand" therapy [107] over a distance, as shown in Fig. 6.2. Haptic tele-robots are also the ideal vehicles for moving the rehabilitation process to the home, as the therapist can train different patient-side robots in different houses without changing his/her location. Thus, tele-robots can increase access to and reduce costs of health care for patients living in remote areas [50]. One aspect of teleoperation to keep in mind is the possibility of delay. Given that TMKT incorporates LfD, the training process occurs offline. Therefore, during the imitation phase, there is no interaction between the two robots. As a result, the system does not present any delay.

Note that both approaches are performed with the therapist and patient interacting concurrently with the robot to perform demonstrations. It is possible to have the therapist and patient provide demonstrations of their guidance and capabilities in a sequential manner, where two sets of demonstrations would be recorded (one with the therapist performing the task alone, and one with the patient alone). While the sequential method makes establishing a performance differential easier, concurrent demonstration more closely resembles conventional, non-robotic rehabilitation in which the therapist and patient frequently interact to practice therapy tasks together. Also, for this work, we only require the robot to imitate the ideal task performance, without

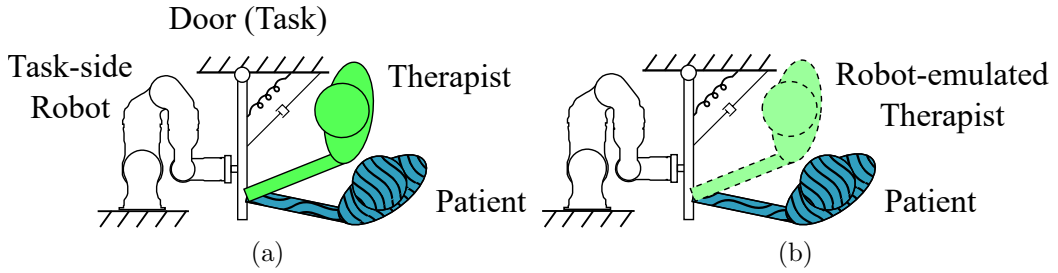


Figure 6.1: A generalized diagram for the LfD procedure combined with RMKT applied to a self-closing door task. Note that in this work we are using these concepts to a different task than depicted here. (a) depicts the patient interacting with the therapist and the robot. This demonstration can be taken as the ideal task performance, or used to establish a performance differential between the patient's capabilities and the therapist-patient combined capabilities. Whichever is chosen can be used later in (b), which depicts the patient interacting with the task-side robot at a later time. The robot emulates the therapist's behaviour learned in (a).

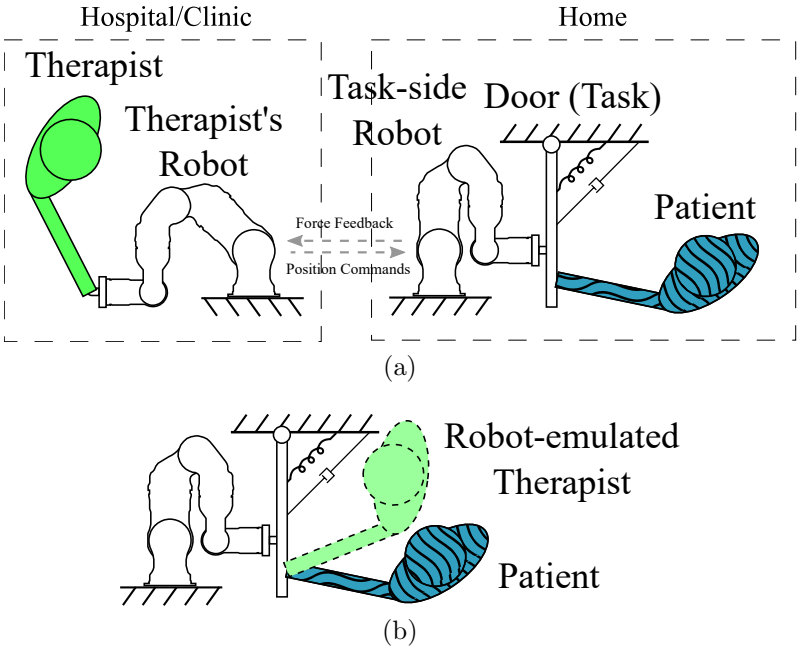


Figure 6.2: A generalized diagram of the LfD procedure combined with TMKT. (a) depicts the patient interacting with the therapist through telerobots. Similarly to the RMKT case, a performance differential could be established with these demonstrations, and is, in fact, easier to measure with two separate robots. (b) depicts the patient interacting with the task-side robot at a later time, where the robot emulates the therapist's behaviour.

the need for establishing a performance differential. Lastly, variation in the therapist's performance is akin to patient variation in concurrent demonstration as the robot sees a fusion of the therapist and patient, thereby making each demonstration patient-specific.

6.4 Impedance Control for Therapist-Patient-Robot Interaction

An impedance control scheme is selected to allow the therapist and patient to interact with one robot in the RMKT case safely, and for the therapist's robot to move the task-side robot in the TMKT case. Impedance controllers produce the desired force based on a predefined relationship with the robot's motion. We use a heavier industrial robot in this work, with internal gearing in the joints. These kinds of robots are typically impossible to move passively. Implementing impedance control allows for a user to move geared robots easily. Impedance controllers also remain stable when in contact with environments with high impedance, such as a human gripping and holding a robot in place [108]. Impedance controllers require the dynamics of the robot to be well modelled [92]. Using the same representation of the robot as in Chapter 4, the robot dynamics can be written as

$$M_r\left(\theta_s\right)\ddot{\theta}_r + C_r\left(\theta_r, \dot{\theta}_r\right)\dot{\theta}_r + g_r\left(\theta\right) + f_f\left(\theta_r, \dot{\theta}_r\right) - J_r F_e = \tau_r \tag{6.1}$$

where θ_r represents the robot joint angles, M_r the moment of inertia matrix, C_r the Coriolis and centrifugal matrix, g_r the gravity vector, J_r the robot's Jacobian, f_f the robot's joint friction vector, F_e the force exerted by the patient on the robot end-effector, and τ_r the controller's output motor torque. The dependence on θ_r will be dropped for brevity. The non-linear terms M_r, C_r, g_r and f_f can be roughly modelled, but will likely be inaccurate, potentially leading to undesired dynamics.

The Time Delay Estimation (TDE) method (as in [93, 94]) is again used here to reduce the inaccuracy when estimating these non-linear terms (see Chapter 4. The desired robot joint torque controller in Cartesian space can

again be given as:

$$\tau_r = \bar{M}_r J_r^{-1} \left\{ \ddot{x}_{r,d} - M_d^{-1} \left[B_d \left(\dot{x}_r - \dot{x}_{r,d} \right) + K_d \left(x_r - x_{r,d} \right) - F_e \right] - \dot{J}_r \dot{\theta}_r \right\} + \bar{M}_r \ddot{\theta}_r + \bar{C}_r \dot{\theta}_r + \bar{g}_r + \bar{f}_f + \tilde{N} - J_r F_e \quad (6.2)$$

We use this final representation of the controller in two different ways. For RMKT, the input comes from the therapist, patient, and environment acting on the force sensor. Therefore, F_e is used as the input signal for RMKT. For TMKT, the input comes from the desired motion of the master robot, given as velocities in this case. $\dot{x}_{r,d}$ is therefore used as the input signal instead. Note that the task performed with the robot will be solely in 1 Degree of Freedom (DOF), greatly simplifying the dynamics model estimation.

6.5 Learning from Demonstration

LfD is a paradigm focused on allowing a human user to program a robot through demonstration of desired behaviours [60, 61, 48]. In general, the behaviours are actions or movements to be later imitated by the robot.

A cornerstone and driver of our LfD-based approach is the assumption that programming know-how is limited in clinical settings. This requires that reprogramming the robotic system between different tasks must be made as simple and user-friendly as possible. State-of-the-art LfD techniques allow for this and facilitate robot learning based on only a few real demonstrations of the task by a human without any additional computer programming overhead.

LfD is divided into two phases, known as the demonstration and imitation phases. In the demonstration phase, a trainer interacts with the robot and performs an action that is to be learned by the robot. Multiple demonstrations of the task can be completed to provide a wider knowledge base for the robot. The imitation phase then imitates the learned behaviour based on the inputs the robot receives in real time.

In this chapter, Gaussian Mixture Models (GMM) and Gaussian Mixture Regression (GMR) are used as the underlying learning and imitation algorithms for the LfD paradigm. The GMM algorithm takes multiple demonstra-

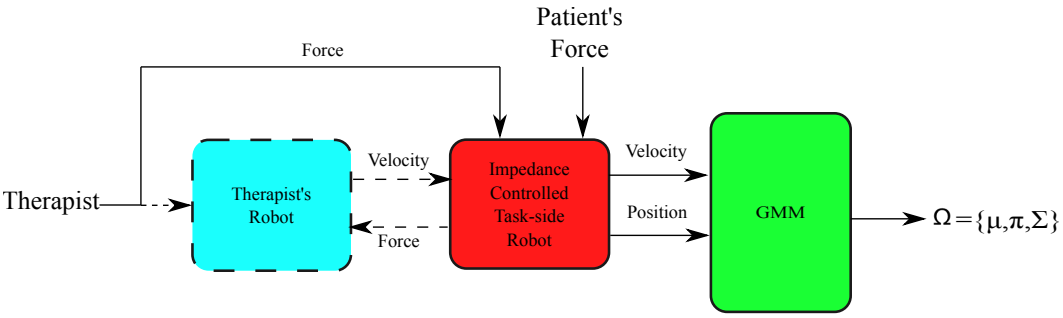


Figure 6.3: Block diagram of the system when used to provide demonstrations, which are used to train the GMM.

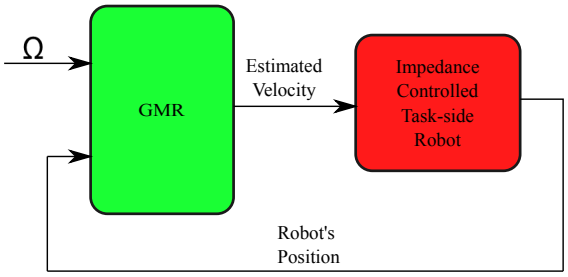
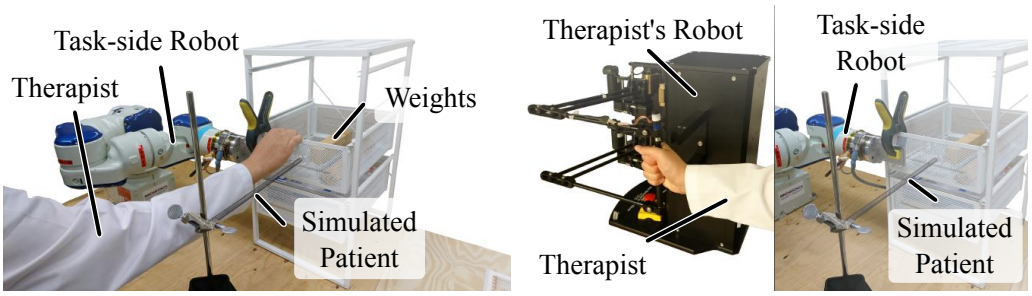


Figure 6.4: Block diagram of the system when imitating the therapist's demonstrated behavior, using the output of the GMR.

tions and extracts the necessary parameters to describe the data with Gaussian functions. This process avoids redundancy of data in memory. The GMR algorithm uses the stored data and, based on the regression input, retrieves the general form of the output. These concepts are discussed previously in Chapter 3, so details are not provided here. The GMM implementation in concert with the rest of the robotic control system is shown in Fig. 6.3, and a diagram of the GMR output being used in the task imitation phase is shown in Fig. 6.4.

6.6 Experiments

We implement each teaching modality using the same task. Comparing these two different ways of reprogramming a rehabilitation system using LfD shows us the strengths and weaknesses of each implementation, and where they perform similarly. In both experiments, we evaluate the system on a simple cooperative task where the participants open a drawer fully as shown in Fig.

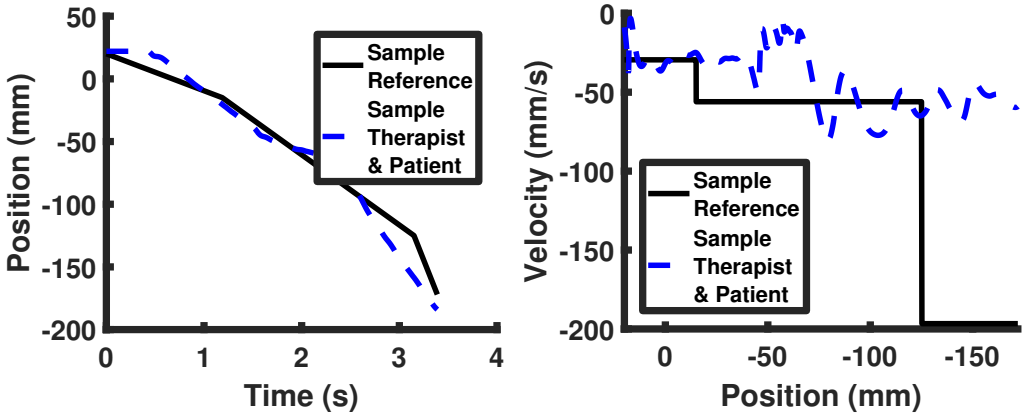


(a) (b)

Figure 6.5: Experimental setups. (a) shows the RMKT setup. The therapist, patient, and robot force sensor hold and open the drawer together. (b) shows the master robot that is added in TMKT. The therapist holds the master robot and moves the task-side robot through a direct force reflection control loop.

6.5 (a). The drawer contains objects with a small mass which creates friction between the drawer and the shelf's rails. Therefore, it resists the opening movement and tends to keep the drawer to its position. In the experiments, the patient is emulated by a weak (low stiffness) spring attached to the front of the drawer, which tends to open the drawer but cannot do so completely. This means the emulated patient cannot complete the task alone due to the simulated disability. The therapist (the role of which is played by our able-bodied human participants[1]) provides assistance to the simulated patient by helping to pull the drawer open while trying to follow a specific reference motion trajectory (Fig. 6.6 (a)). In all trials, the robot's end-effector position and velocity are recorded and later used to train the system as outlined in Fig. 6.3. Later, during the imitation phase, the GMR takes the robot's current end-effector position as query points to compute the desired velocity used by the controller to imitate the therapist's behaviour (Fig. 6.4). The position and velocity data of the robot end-effector are again collected, as well as the output variance of the GMR. The robot end-effector is attached to the front of the drawer. An impedance controller is used to provide robot compliance to participant input in 1 DOF.

[1]Ethics approval was granted by the University of Alberta Research Ethics Office under study ID MS10_Pro00033955.



(a) (b)

Figure 6.6: (a) shows an example of the trajectory data displayed to a participant during an experiment. The position and time data of the participant and patient's collaborative motion are plotted in real time as they attempt to follow a reference trajectory. (b) shows the extracted velocity-position trajectory performed by the participant, which is used to train the GMM.

6.6.1 Robotics-mediated Kinesthetic Teaching

In this experiment, the participant (i.e., therapist) trains the robot by holding its end-effector and assisting the simulated patient to complete the task. Each participant was asked to follow one given reference trajectory, which varied between each participant. The participants complete the task five different times following their given trajectory. Reference trajectories are randomized for the purpose of showing that the imitation results of the LfD algorithms are generalizable, and second to vary the difficulty of the task. A Motoman SIA5F 7 DOF industrial manipulator (Yaskawa America, Inc., Miamisburg, Ohio, USA) is used as the task-side robot for rehabilitation of the patient.

6.6.2 Telerobotics-mediated Kinesthetic Teaching

The second experiment requires the therapist and the patient to collaborate to complete the task while using a telerobotic system. As shown in Fig. 6.5 (b), the therapist interacts with the patient using a master-slave system, where the master robot is controlled by the therapist and the slave robot is the task-side robot with which the patient interacts. Once again, each participant helps the

Table 6.1: Average, minimum, and maximum variance for the GMR output corresponding to each participant's demonstration sets during RMKT and TMKT modalities.

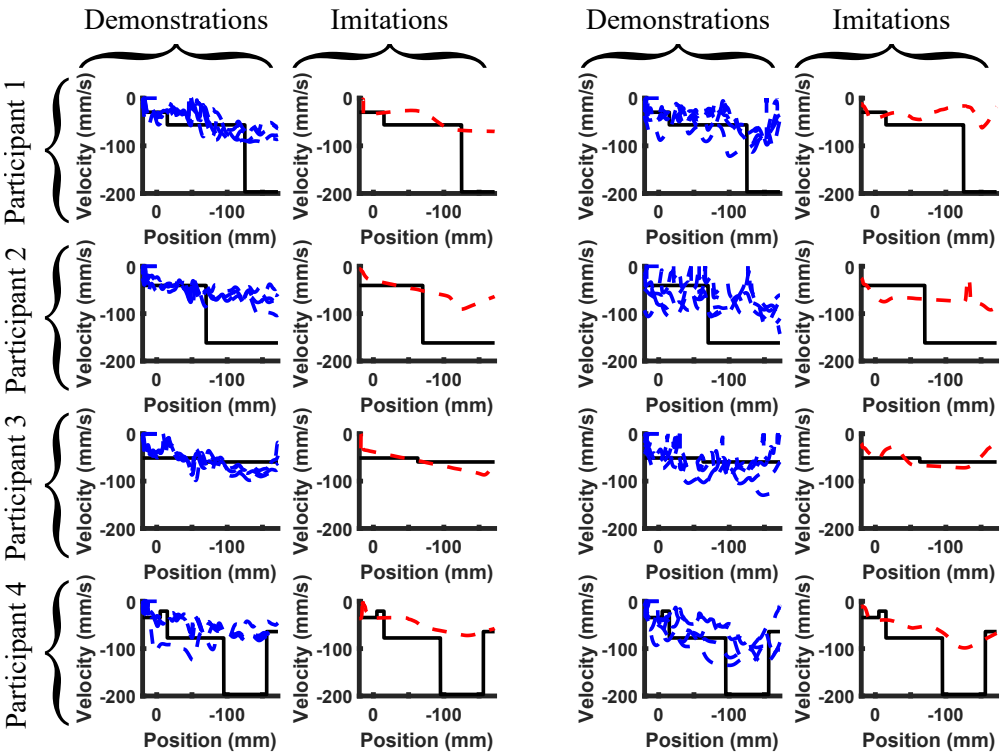
		Mean (mm/s)	Max (mm/s)	Min (mm/s)
Participant 1	RMKT	273.4	781.1	56.3
	TMKT	223.1	530.5	62.2
Participant 2	RMKT	151.5	714.6	75.2
	TMKT	686.8	1468.8	215.1
Participant 3	RMKT	170.5	760.3	44.0
	TMKT	596.8	1444.2	63.8
Participant 4	RMKT	258.9	660.3	72.8
	TMKT	489.1	1281.6	166.2

simulated patient to complete the task in a similar way as before for a total of five demonstrations. To improve the transparency between the therapist and the patient, force feedback is used on the therapist's side. The same Motoman SIA5F robot as in the RMKT case is used as the task-side (slave) robot here, while an HD^2 6 DOF robot (Quanser Inc., Markham, Ontario, Canada) is used as the therapist's user interface (master).

6.7 Results & Discussion

We present our results and analysis of the obtained data in three ways. First, we compare the participant-demonstrated velocity vs. position and the GMR-generated velocity vs. position for each experiment. Using the robot end-effector's recorded velocity and position data, we plot the results in Fig. 6.7 (a) and 6.7 (b). The figures provide a qualitative overview of how accurate the participants and the system trained by them were in following the reference trajectory. Second, we provide the variances of the GMR outputs for each imitation in order to quantitatively evaluate how repeatable and, by extension, how easy demonstrating the reference trajectories are for each modality (Table 6.1). Lastly, we perform a Student's T-Test is performed to compare the GMR output variances so as to provide a numerical evaluation of the modalities' similarity or difference. A box plot (Fig. 6.8) is used for visualization.

The results in Table 6.1 show a wide spread of GMR output variances,



(a) RMKT (b) TMKT

Figure 6.7: Velocity-position data from demonstrations (dashed blue, under demonstrations) and GMR imitations (dashed red, under imitations), plotted against their respective reference trajectories (black). (a) shows the trajectories recorded from the RMKT experiments while (b) shows the trajectories recorded from the TMKT experiments.

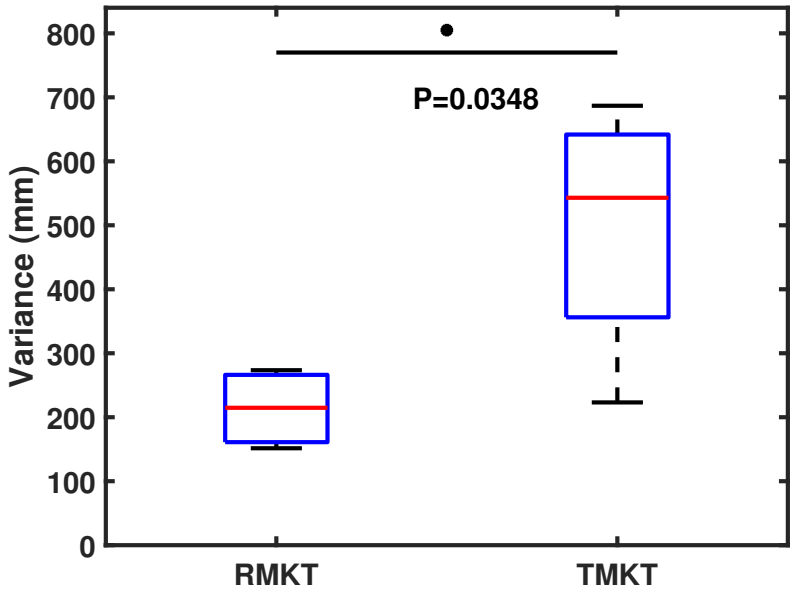


Figure 6.8: Box plot of average GMR trajectory variances for RMKT and TMKT.

differing across each participant. It can be noted that Participants 2, 3, and 4 exhibited larger variance results than Participant 1, meaning that their demonstrations were less consistent during the training phase. We can infer that the level of consistency in demonstrations, therefore, varies greatly on a user by user basis. A clear example of this is that Participant 1 shows smaller average and maximum variances when using TMKT while the rest of the participants show larger variances while using the TMKT modality. This observation may indicate that in general, it is more difficult to provide consistent demonstrations with the teleoperation setup, detracting from the feasibility of TMKT and corroborating Hypothesis 1. From a different perspective, minimum variances are in general consistently low, so for some participants, there are at least some portions of the trajectory that are repeatable for both modalities. Fig. 6.8 visualizes the T-Test results comparing the average velocity variance values obtained from the imitation GMR velocities. The results are not statistically similar ($p = 0.0348$), confirming the conclusions drawn from the data in Table 6.1.

As seen in Fig. 6.7 (a) and 6.7 (b), the participants sometimes experienced difficulties following the reference velocity depending on their reference

trajectory's level of difficulty. Therefore, the GMR results do not match the reference velocity accurately. However, the GMR output does closely resemble its training data, specifically in that portions of the trajectories that require changes in speed are properly conveyed and appear similar. These similarities can be interpreted as a good statistical reconstruction of the therapist's behaviour for both modalities. We can conclude that incorporating LfD techniques with both modalities is indeed possible, as in Hypothesis 2. This can be seen as an important step towards the introduction of TMKT as a research focus. We can summarize the conclusions in that TMKT may lead to less consistent demonstration data resulting in a less user-friendly interface, but does indeed allow for LfD algorithms to properly learn user demonstrations as in RMKT.

Several possible factors contributing to TMKT's lower performance (as well as general improvements to the experimental procedure) can be theorized. The most likely factor would be a lack of co-location between the user and the task. In RMKT, it is highly intuitive for the therapist to match their input to the resulting change in the task performance. However, with TMKT the spacial disconnect could have a negative effect on the user's perception. This in part leads to a possible second factor, in that the TMKT system was designed with ideal transparency as the target in mind, but without replicating the task-side robot's dynamics (e.g., inertia) on the therapist's robot's side. In this work, this meant the therapist's robot was much easier to move and could have resulted in difficulty in perceiving the degree of motion the task-side robot was undergoing. This work also focused on making TMKT as close to RMKT in performance, but one unused advantage of teleoperation is that workspace or force scaling is possible, which can be used to lower effort requirements for the master robot operator. More general factors could be that the task may have been too difficult for some participants and that the sample size was small. Further investigation into these factors could potentially produce more favourable results for TMKT, although the modality was still able to properly incorporate LfD regardless.

6.8 Conclusion

In this work, two different modalities for incorporating LfD techniques into robotic rehabilitation, RMKT and TMKT, were compared for feasibility and ease of use. A simple cooperative task of opening a drawer was used to represent a therapy task upon which to perform the comparison. The results indicated that both modalities were capable of providing demonstration data to the LfD algorithms to a satisfactory degree (validating Hypothesis 2), although TMKT demonstrations had a larger variance on average and were not statistically similar to their RMKT counterparts ($p = 0.0348$) (validating Hypothesis 1). Future works will focus on improving the user immersion in TMKT (e.g., mimicking task-side robot dynamics on the master robot), taking advantage of teleoperation-based techniques to make interaction easier, tuning experimental procedures, and using larger sample sizes. We would also like to test both approaches with and collect feedback from patients and therapists in an effort to properly validate the patient-specific aspect of the proposed system, for which we expect favorable results based on our previous works.

Chapter 7

Robotic System for Functional Capacity Evaluation

7.1 Introduction

The growing demand for rehabilitation services following a workplace injury has motivated the development of new technologies for robotics-assisted assessment and rehabilitation of motor function following injury. The standard practice in occupational (or vocational) rehabilitation is to first perform a functional assessment of the injured worker. Typically, this is done using a Functional Capacity Evaluation (FCE) that assesses a worker's performance in a set of standard tasks [109], where each task requires different sets of equipment. The tasks incorporated in the FCE may involve material-handling activities such as lifting, pushing, and pulling, and positional tolerance activities such as walking, reaching, and grasping.

The first problem with the above is that it needs a large amount of equipment for various functional tasks and the space to store them. While a small number of all-in-one computer-based assessment tools exist [110, 111], they are highly specialized in design and can replicate only specific rehabilitation tasks. A second problem emerges due to the current standardized assessments, where therapists qualitatively assess a patient's performance based on what they can observe. More complex, quantitative and objective assessments are desired. A third problem occurs when therapists increase the difficulty of a task or ask the injured workers to execute tasks that are considered boring;

the patients can become bored, unmotivated, or uncooperative.

To address the above issues, we propose a generalized robotics-based solution. Our solution incorporates a serial-manipulator and a projection-based Augmented Reality (AR) display in order to provide a unified tool for both FCE and rehabilitation that is immersive and device-independent. To evaluate the efficacy of the proposed system, the biomechanics of the user's arm while using the system is retrieved and compared against the biomechanics of their arm in an equivalent real-life performance of the same task. In this regard, we present the following hypothesis: The proposed system can be used as an alternative to traditional occupational rehabilitation exercise environments because *it does not significantly modify the biomechanics of the user's arm while performing functional tasks compared to the conventional task performance.*

The chapter is structured as follows: Section 7.2 is a brief overview of the work found in the literature that relates to our proposed approach. Section 7.3 describes the design of the rehabilitation exercise and experimental procedure. Section 7.4 presents the results and provides a discussion based on the performed data analysis. Finally, Section 7.5 concludes the findings and examines possible directions for future work.

7.2 Related Work

7.2.1 FCE

FCE is widely used to assess injured workers before, during and after rehabilitation. A number of studies have demonstrated the reliability and validity of FCE and correlation with future recovery and return to work. Peppers et al. showed that augmenting clinical evaluation with FCE improves physicians' assessments of the patient's skills and work capacities [112]. Gross et al. studied the impact and benefits of integrating FCE into rehabilitation for better outcomes for injured workers [113]. FCE has been found to significantly predict return to work [114] and is an integral component of graded activity and functional rehabilitation programs [115]. However, James et al. concluded that further research is needed in FCE, especially on the use of computer

technology (including robotics and digital sensors) [116].

7.2.2 Robot-assisted Assessment

The inclusion of robots in therapy is becoming more common thanks to robots' power, repetitive motion ability, reprogramming capacity and potential adaptability to new tasks. These features allow robots to be used in therapy fields such as emotional therapy and physical therapy. Yakub et al. provide a list of robots developed in the context of rehabilitation medicine [117]. The use of robots in occupational rehabilitation began in the early 1990s [118, 119], although they were employed mainly as assistive devices for workers with injury or disability. Recent developments in the area have culminated in devices such as BTE's EvalTech [110] and Simwork's Ergos II [111] systems, which simulate FCE assessment setups and can also be used for strength and movement coordination training. However, these devices are specifically designed to emulate a certain set of FCE tasks. Also, the performance of tasks with these systems are spatially constrained to their placement on the devices and the performance of tasks involving free-space motions is not an option. For instance, while a device may include a lock for practicing turning a key to open it, the more challenging task for painting a wall is not supported because it cannot be done at one point on the device. The tasks also remain limited by the need to have physical objects that the user holds during assessments (e.g., rotating handles and knobs).

7.2.3 Virtual Reality & Augmented Reality in Rehabilitation

Virtual reality (VR) and AR technology has been making its way into the rehabilitation field in recent years. It has been shown to increase the motivation of patients and keep them engaged since it uses games to disguise the repetitive movements of the rehabilitation exercises [120]. However, most of the VR and AR rehabilitation systems in the literature and on the market are targeted for those who have been affected by neurological injuries due to events such as stroke and spinal cord injury [26]. These systems cannot be used by injured

workers as-is due to the difference in challenge level and sophistication of the rehabilitation tasks between the two groups (i.e., stroke patients and injured workers).

For non-immersive VR, in which the game is displayed in a 2D screen in front of the patient, there exist systems like the BTE Eccentron [121] to improve lower-limb strength while providing an interactive game-like experience to guide the patient toward their objectives. To the best of the authors' knowledge, there are currently no immersive VR or AR systems that train injured workers to regain muscle strength to enable them to return to work. There is also no robotic system that is specifically developed for simulating the physical dynamics of functional tasks for the rehabilitation of injured workers. Our proposed system employs the use of a 3D spatial AR display to immerse the patient in a projected 3D virtual environment that is integrated with the physical environment including the robotic manipulator. Previous research from our group shows that the resultant colocation of visual and motor axes help improve user performance in rehabilitation exercises [122].

We propose an approach based on using a seven Degree-of-Freedom (DOF) serial manipulator for simulating the physical dynamics (i.e., haptic interaction) corresponding to functional tasks, eliminating the need for physical hardware of such tasks. Compared to rehabilitation facilities that allocate a large area for multiple tasks, this unified system can reduce the costs for equipment. Our approach also integrates an AR display to provide reconstructed visual feedback of the simulated task in an immersive environment. All types of motions can be performed on the robot due to its seven DOF design. This allows flexibility in movement that is not found in other systems. Furthermore, having a robotic system allows for masking the task parameters from the patient which can help prevent the loss of motivation from knowing about an increase in the difficulty level of the task.

The overall robot-AR system is useful for both FCE and rehabilitation of injured workers. Serial manipulators have been previously incorporated into rehabilitation medicine for both assessment and rehabilitation purposes [123]. Our group, in particular, has extensively applied serial rehabilitation robots

to target the neuromuscular rehabilitation of patients with stroke [86]. Likewise, we have also developed a robot-assisted AR system for simulated stroke patients, in which the effects of stroke (e.g., being distracted) is simulated by cognitively loading the user with a count down task. However, to the best of our knowledge, the use of robots and AR in the context of facilitating FCE and rehabilitation of injured workers remains unexplored.

7.3 Materials and Methods

7.3.1 Rehabilitation Task Design

The simplified movements found in rehabilitation tasks often involve reaching, grasping, and weight lifting. The task used in our robot-AR system implements these movements in their basic forms but can be further adapted to higher difficulty and complexity levels.

We chose a painting task that trains up-down hand movements by having the user paint a vertical wall. A fill indicator provides the user with information on the percentage of the wall that is already painted. Force feedback is provided by the robot when the virtual paint roller is in contact with the wall so that the haptic experience of painting on the wall in the real world is recreated. In the real-world condition, the user is given a physical paint roller to use on a portable physical wall positioned at the same spot the virtual wall was in the robot-AR condition. No paint is used in the real case; rather, the user is asked to "paint over" an area of the wall as much as they can. The user "paints" until the area they have covered encompasses the wall in the virtual task. Measurements such as time of completion and amount of force exerted by the user can potentially be retrieved and analyzed in the robot-AR setup, but are not within the focus of this chapter.

7.3.2 Robot Manipulator Choice and Control Strategy

Many standardized FCEs such as the WorkWell FCE and the Progressive Isoinertial Lifting Evaluation (PILE) place emphasis on an injured worker's ability to lift weighted objects (e.g., crates) as an important assessment, among

other physically strenuous tasks [109]. Therefore, it is desirable to use a robot capable of exerting enough force to realistically simulate heavy objects and interactions with environments typical of an injured worker's workplace. For this reason, the robot used in this work is a heavy-duty industrial robot; details are provided in Section 7.3.3. Internal gearing makes the structure of the robot non-back-drivable; however, the requirement of physical human-robot interaction (PHRI) in our experiments means a suitable robotic controller is needed to make the robot back-drivable.

Impedance controllers, which output a force for a robot to exert based on its motion, are ideal for providing stable PHRI when simulating environmental interactions. However, implementing such controllers typically requires full knowledge of the robot's dynamics parameters such as each joint's mass and center of mass [92], which are unavailable for the robot used in this work and difficult to accurately measure. Admittance controllers, which output a motion for the robot to execute based on a measured force input, are a common alternative for non-back-drivable, heavy-duty robots like the one used in this chapter. The general form of an admittance controller's transfer function is

$$G = \frac{\vec{V}_d(s)}{\vec{W}(s)} = \frac{1}{Ms + B} \tag{7.1}$$

Note that in this chapter, an internal velocity controller is used by the robot to perform movements in real-time, so the admittance controller is designed here to output a desired velocity rather than a desired position. The input to the controller, $\vec{W}(s) = \left[\vec{F}(s), \vec{\tau}(s)\right]^{\mathsf{T}}$, represents the wrench composed of input forces and torques, and the output, $\vec{V}_d(s)$, is the resulting desired velocity, composed of Cartesian and angular terms. M represents the desired mass and inertia matrix and B represents the desired Cartesian and angular damping matrix. These matrices affect the transparency of free motion experienced by the user and also the stability of the robot. A stiffness parameter is not used, similar to [104], because restoring forces are not desirable during co-

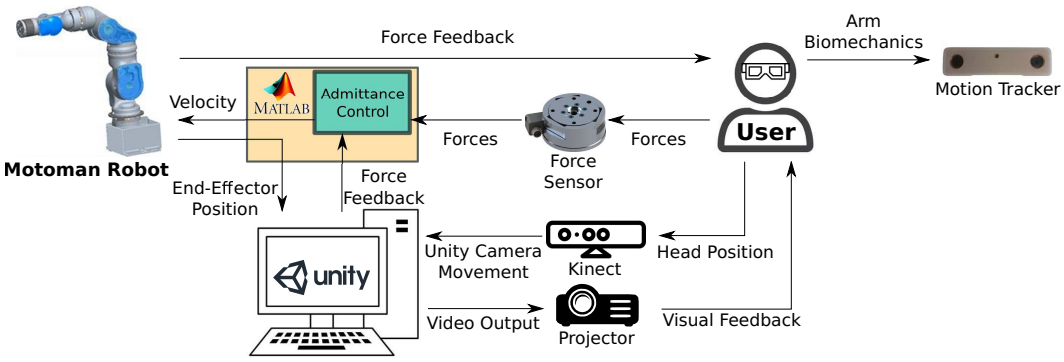


Figure 7.1: Flowchart of the communication between each system.

manipulation in free-space. It follows that G is given as

$$G = \begin{bmatrix} g_x & 0 & 0 & 0 & 0 & 0 \\ 0 & g_y & 0 & 0 & 0 & 0 \\ 0 & 0 & g_z & 0 & 0 & 0 \\ 0 & 0 & 0 & g_\alpha & 0 & 0 \\ 0 & 0 & 0 & 0 & g_\beta & 0 \\ 0 & 0 & 0 & 0 & 0 & g_\gamma \end{bmatrix}$$

where $\{g_x,\ g_y,\ \ldots,\ g_\gamma\}$ represent the admittance terms for each Cartesian direction and orientation angle. Large values for admittance terms result in greater allowed motions while small values result in more constrained movements. By changing the admittance parameters, allowed movements initiated by the user can be restricted to certain axes. This is used, for example, in our task where it is beneficial to restrict rotations in axes that are not of interest (e.g., small values for g_β and g_γ) while allowing free motion in the other axes (e.g., large values for g_x, g_y, g_z, and g_α).

7.3.3 Experimental Setup

As seen in Fig.7.1, the robot-AR system uses a Motoman SIA-5F (Yaskawa America, Inc., Miamisburg, Ohio, USA) seven DOF serial manipulator as the user interface to control the paint roller in the virtual environment. It is controlled using MATLAB, Simulink, and C++ in which the flow of communication between them is described in [94]. Attached to the robot's wrist joint before the end-effector is a 6-DOF ATI Gamma Net force/torque sensor (ATI Industrial Automation, Inc., Apex, North Carolina, USA). The AR subsystem

consists of an off-the-shelf InFocus IN116A projector mounted 3 m above the ground that projects to a screen on the table. A Microsoft Kinect V2 Sensor is positioned 1.2 m horizontally distant and 0.34 m vertically above the user's head to enable head tracking for displaying the correct perspective to the user. To properly view the 3D scene, active DLP-Link 3D shutter glasses are worn by the user. The development of the 3D environment is done using the Unity Game Engine [124] (Unity Technologies ApS, San Francisco, California, USA) where a virtual model of the workspace is created. This virtual model is created and calibrated to the world scale using Microsoft's RoomAlive Toolkit [125]. A ClaroNav MicronTracker (ClaroNav, Inc., Toronto, Ontario, Canada) motion tracking camera (MTC) is used to record the positions of the user's hand, elbow, and shoulder.

As mentioned earlier, the requirements of the more strenuous FCE and rehabilitation tasks imply a need for high force and torque haptic interactions. The admittance-type Motoman robot is used as the manipulator due to its heavy load capabilities compared to other impedance-type haptic interfaces. It has a payload limit of 5 kg for accurate movement and can generate joint torques up to a rated 300 Nm. These are the maximum values achievable by the robot and we do not use all of it. For safety, constraints are placed in the software to limit high velocities and position singularities. The attached force sensor records user force and torque inputs, which are used to facilitate the admittance control of the robot.

The painting task requires a specific setup of the projector around the robot due to the limited projection space and required robot configuration. The configuration uses a curved screen with dimensions 85 cm tall, 75 cm deep, and 56 cm wide situated on top of the table. The end-effector of the robot is positioned to the right of the screen. This configuration allows the user to have an intuitive feel of the simulated task and reduces the occlusions on the projection display caused by the user's arm and robot joints.

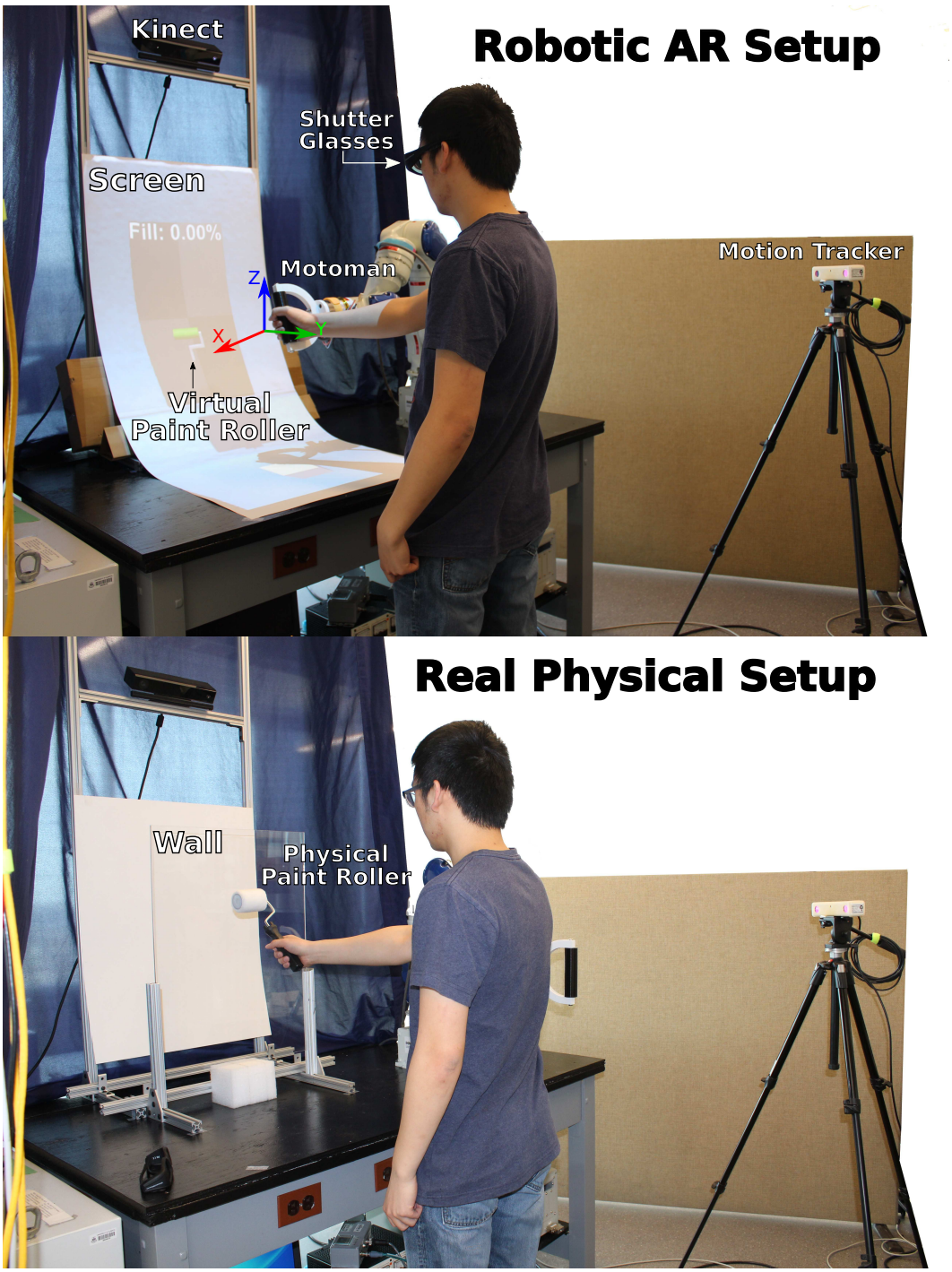


Figure 7.2: Painting task experimental setup for the robot-AR condition (top) and the real-life equivalent condition (bottom). The projector is not shown. Through AR, the paint roller will pop out in 3D from the perspective of the user in a geometrically correct position and orientation relative to the robot end-effector.

7.3.4 Experimental Procedure

5 trials for each condition (i.e., robot-AR or real-world) are carried out to have a total of 10 trials per person, lasting approximately 60 s per trial. The trials are performed by 2 able-bodied participants (both are male, 24 years old, and right-handed). Each participant is asked to stand in a comfortable position in front of the screen and to hold the Motoman robot's end-effector with his arm half extended. The participant is instructed to refrain from changing his standing location, which is marked on the floor, between the two experimental conditions. A chair is provided for the participant to take rests when needed. All 5 trials are recorded for a specific condition before moving onto the other one. The robot-AR condition for the painting task is presented to *Participant 1* as the first set of trials before doing the trials under the real-world condition. The opposite order is presented to *Participant 2*.

7.4 Results and Discussion

The hand, elbow, and shoulder positions recorded by the MTC form the data for the biomechanics analysis performed. These are recorded by placing fiducial markers on the back of the user's hand, and on the elbow and shoulder. To evaluate the similarity of the biomechanics between using the proposed system and the equivalent real-world task, we consider the hand position as the independent variable and the elbow and shoulder positions as the dependent variables. In other words, while the user's hand position changes depending on the goal of the task, the elbow and shoulder joint positions will change in order to best accommodate the desired hand pose. For a fixed hand position (independent variable), we will compare the distribution of the dependent variables between the two conditions.

The two-sample Kolmogorov-Smirnov (KS) test is a commonly used method of evaluating whether two one-dimensional distributions are statistically different (the null hypothesis is that they are similar). Since p_{H-E} and p_{E-S} are in three dimensions, we use the modified version of the KS test in three dimensions as described in [126] by Fasano and Franceschini. We make use of

the implementation of Fasano and Franceschini's work in [127] in conjunction with the Monte-Carlo simulations provided in the original work.

A preliminary comparison between the datasets for the two conditions is performed first to see if the distribution of elbow and shoulder joint positions for the same hand position as it traverses the entire surface of the wall being painted is statistically similar between the two conditions. Here, the joint position data from the real-world condition is taken as the baseline data; for a specific hand position, the elbow and shoulder positions for the robot-AR condition should resemble those measured in the real-world condition. If this happens, it can be concluded that the robot-AR system does not significantly modify how users perform the task compared to the real-world condition. The process of comparison is given as follows: for each recorded hand position in the robot-AR dataset, a similar hand position in the real-world data is found by using a nearest-neighbor (NN) search. For these similar hand positions, the associated hand-to-elbow (H2E) and elbow-to-shoulder (E2S) displacements can be calculated in each dataset. The result is a distribution of H2E and E2S displacements recorded in the robot-AR condition, and a distribution of H2E and E2S displacements that are associated with the real-world condition for the same hand positions. The modified KS test is then used to compare the H2E displacement distributions, and the E2S displacement distributions between the two conditions. Note that [126] only provides Monte Carlo simulations up to a maximum $n = 500$, where in a two sample KS test $n = \frac{n_1 n_2}{n_1 + n_2}$ where n_1 and n_2 are the number of points in the real-world condition and robot-AR condition, respectively. Knowing that the number of points in the distributions is the same, i.e., $n_1 = n_2$, we then restrict the number of points in the distributions to 1000 points or less. To do this, the collection of datapoints are downsampled to 1000 points for each condition, resulting in 200 points per trial. A significance value is returned by the test and is compared against an alpha value of $\alpha = 0.05$.

The results for the H2E and E2S comparisons produced a value of $p < 0.05$, indicating that the distributions are statistically different (i.e., rejecting the null hypothesis that two conditions have the same distribution) and there-

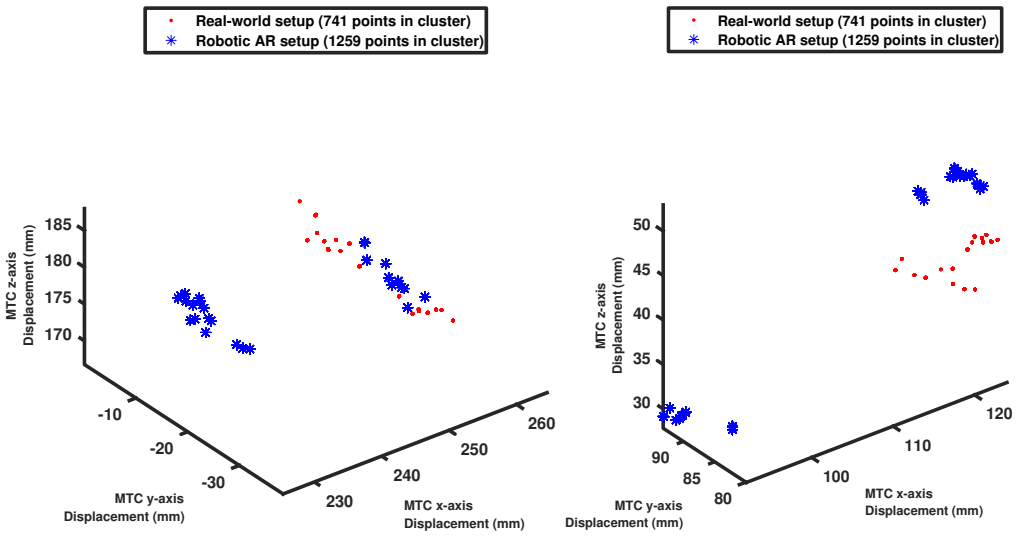


Figure 7.3: Joint position data for an example cluster. (a) shows the point cloud data for H2E displacements, and (b) shows E2S displacements.

fore suggests that the biomechanics of the two conditions are different. This motivates a closer inspection of the data.

Examining whether there are spatial trends in the similarity between the distributions may help explain the dissimilarity reported in the KS test for the full dataset. To do this, we propose to divide the data into spatial sections or voxels and performing the KS test for each voxel, looking for any that may be dissimilar. A grid of measurement points is first constructed by choosing points at evenly-spaced intervals to encompass the range of hand positions across all datasets in the three Cartesian dimensions. All recorded hand positions are then clustered to the nearest grid point using the NN search. We use an interval of 25 mm as the distance between grid points, as it provides a high resolution of voxels in our task space and allows most clusters to meet the requirements for n_1 and n_2. Fig. 7.3 shows the distributions of H2E displacements and E2S displacements for an example cluster.

For each cluster, the statistical similarity of the distributions for the associated data from the two conditions (robot-AR and real-world setups) is then evaluated with the modified KS test. Similar to before, the Monte Carlo simulations in [126] are only provided for a minimum of $n = 10$ between two

samples (and a maximum of $n = 500$). To ensure $10 \leq n \leq 500$, we impose conservative limits where $n_1 \geq 20, n_2 \geq 20$ and $n_1 + n_2 \leq 2000$. Graphical results of the modified KS tests are shown in Fig. 7.4.

The percentage of clusters that are statistically similar between the two conditions show that 43.85% of the measurable clusters were similar for H2E displacements and 28.46% were similar for E2S displacements during Participant 1's trials. Participant 2 achieved 46.67% similarity for H2E displacements and 29.33% similarity for E2S displacements.

At first glance, the fraction of clusters that produced similar results seems to be quite low, especially for the E2S displacements. However, a qualitative observation of the results in Fig. 7.4 shows that the statistically similar clusters are well spread across the entire workspace. There are a few possible reasons as to why some clusters may not show similar results. As the real-world condition experiments did not involve actual paint being laid on the physical wall, keeping track of the "painted" portion proved to be challenging. This could affect the fairness of the KS test performed. For example, if, for a specific cluster, $n_1 \gg n_2$, where $n_1, n_2 > 20$, then a comparison of the distributions would be valid according to the restrictions we placed on the comparison in a cluster, but it could suffer from the disparity in the quality of the distributions. The simplest way to address this issue would be to simply have more trials, which in turn would provide more data and a higher chance to better define the distributions for more clusters. It would also be beneficial in this situation to be able to remove the upper limit on datapoints to compare over, meaning running Monte Carlo simulations as in [126] for higher values of n.

Nevertheless, the results indicate there is a perceivable difference between using the robot-AR setup and performing the real-world equivalent task. The most likely cause would be that the damping and inertia of the robot were not low enough to properly convey full transparency during free motion. This could be the case, given the nature of the geared transmission system used in the robot and the admittance controller used to make it compliant. The implied difference in perceived weight during free motion would then be a likely cause in any changes in the observed biomechanics, as the user would

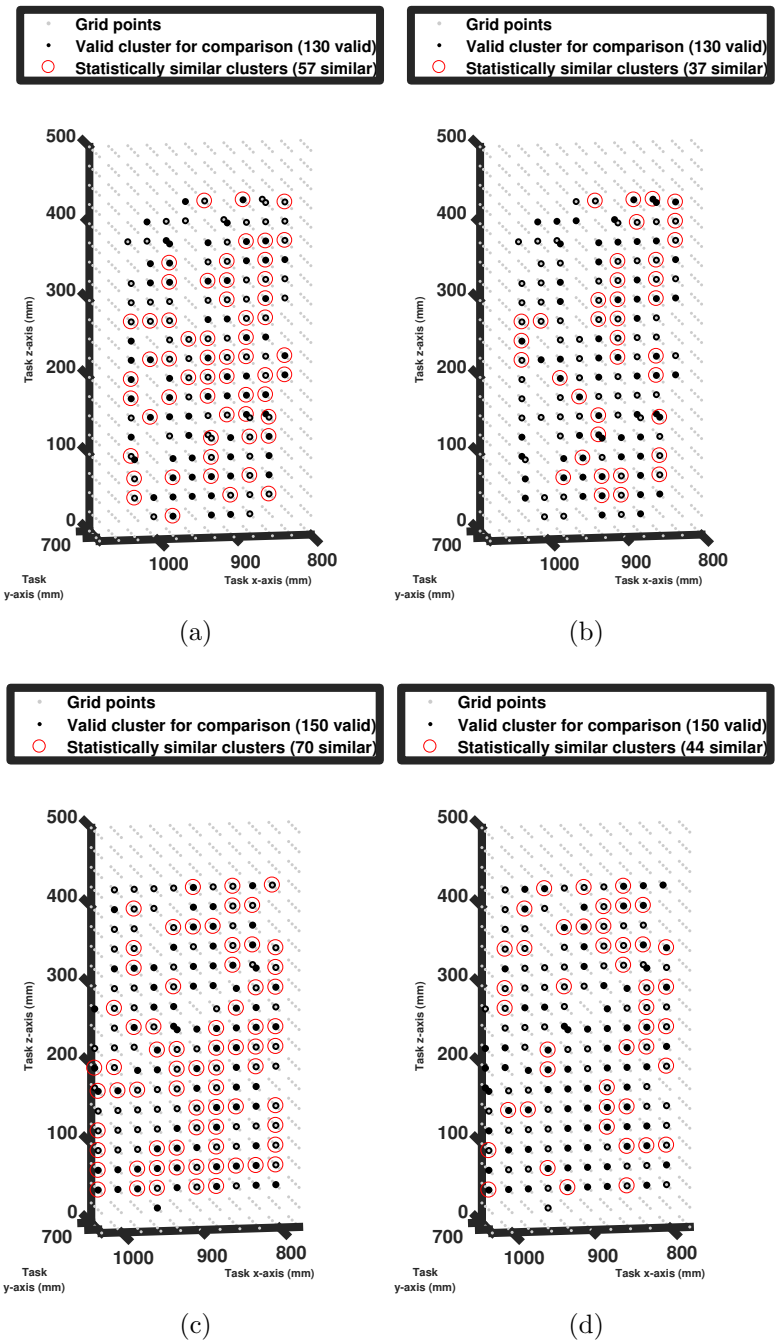


Figure 7.4: Three-dimensional KS results for the painting task with the data split into voxels for comparison. The grid points show points in space around the surface of the wall where H2E and E2S results were clustered and compared at. (a) and (b) represent H2E and E2S results for *Participant 1*, respectively, and (c) and (d) represent the same for *Participant 2*. Clusters with a sufficient number of datapoints for comparison with the KS test are shown with black points and those of statistical similarity are circled in red.

compensate for the heavier load, experienced in the robot-AR condition, by adjusting their joint positions accordingly. In Fig. 7.4, this may be the reason why the E2S distributions have a much lower overall similarity than the H2E distributions, as the upper arm may have moved more in order to compensate for the larger resistance to motion in the robot-AR condition while the lower arm remained the same in order to hold the brush handle comfortably. There is then a motivation for reexamining the results when performed using robot with similar load-bearing capabilities that is designed for the purpose of patient-safe interaction as well as built-in back-drivability.

7.5 Conclusion

In this chapter, a robot-AR system that aims to be a suitable alternative to existing FCE and rehabilitation environments for injured or disabled workers is developed and evaluated. A task that involves painting a wall is presented to the participants. To evaluate our approach, the task has a real-life equivalent condition in which the biomechanics of the participant's arm for both robotic AR and real-life conditions are recorded and compared to determine if the arm movements are similar. Our results show that the arm biomechanics for a painting task have significant differences in $\approx 50\%$ of the collected clusters for the hand-to-elbow (H2E) displacements, and $\approx 30\%$ for the elbow-to-shoulder (E2S) displacements for both participants. These initial findings show the potential of our robot-AR system to replicate upper-limb movements found in traditional FCE and rehabilitation exercises and it motivates us to further investigate better methods to simulate functional tasks. Future work includes implementing the system with a different robot and/or robot controller that is better suited for physical human-robot interaction (PHRI), expanding the projected area, developing more tasks, and testing the system with actual FCE tasks, or even in a clinical setting. By creating an all-in-one robotic AR occupational rehabilitation system, we hope to motivate and provide an efficient method for workers to recover from their injuries.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

This thesis presented an exploratory approach to semi-automation of physical rehabilitation medicine, focusing primarily on the introduction of Learning from Demonstration (LfD) algorithms to rehabilitation robotics. The use of these algorithms was shown to provide a quick and intuitive method for teaching therapeutic behaviors (i.e., guidance in the form of assistance or resistance) to robots. A generalized manipulator-based robotic system was developed throughout the thesis, culminating in an immersive system for presenting rehabilitation tasks through both haptic feedback and visual co-location. These developments represent a glimpse into what the future of the field of rehabilitation robotics, and possibly rehabilitation medicine, may hold: the potential to provide intensive and immersive therapy with minimal burden on healthcare providers while keeping specialists in the loop.

In Chapter 3, we proposed the use of LfD algorithms to facilitate robotic interactions provided to patients practicing a cooperative therapy task. A therapist and patient collaborated to lift an object, where the therapist either provided a constant amount of assistance or adjusted their assistance based on the patient's performance (represented by the speed of their motions). The therapist interacted with the task through a telerobotic medium while the patient interacted with the task directly (i.e., in the same physical space). Provided with demonstrations, the robotic system generated representations of the learned position-based assistance through the use of Gaussian Mixture

Models (GMMs). The ability of the model to imitate the therapeutic behaviors through Gaussian Mixture Regression (GMR) was later evaluated with varying degrees of success. A recommendation was formulated in that for learning behaviors in which demonstrations are subject to higher amounts of variance (i.e., fine motor or slower movements), more demonstrations should be provided.

In Chapter 4, the GMM and GMR-based implementation of LfD was applied to lower limb rehabilitation, namely gait therapy targeting foot drop. This work represented a departure from the telerobotic setup previously used in favor of a single-robot architecture. An impedance controller based on the Time-Delay Estimation (TDE) method of estimating dynamics parameters of the robot was used to enable hands-on movement of the robot by the therapist. A patient with simulated foot drop was assisted by a therapist who directly moved the rehabilitation robot in 1 DOF to provide assistance. The learned assistance was reproduced later, with the toe clearance achieved by the foot with simulated foot drop compared with the clearance achieved during therapist-assisted demonstrations, as well as baseline data of healthy gait. Minimum toe clearance was achieved although clearance levels were not statistically similar to healthy gait.

In Chapter 5, a different implementation of GMM-based LfD was used to learn impedance-based therapeutic behaviors for a 1-DOF upper-limb Activity of Daily Living (ADL) as the therapy exercise. Sequential demonstrations performed by a patient (simulated by a spring) when assisted by the therapist and when attempting the task alone provided a "performance differential" and allowed for learning the therapist's assistive impedance-based behaviors (interventions). An admittance controller was implemented for single-robot interaction as a substitute for the impedance controller used previously, allowing for movements in multiple DOFs without needing the robot's dynamics parameters. Evaluations of the reproduced forces based on the learned assistance showed satisfactory results, with the caveat that areas of the learned model with larger variance would benefit from having a higher resolution in terms of the distribution of GMM components.

In Chapter 6, a comparison between using telerobotic or single-robot setups to provide demonstrations for LfD was performed. The same 1-DOF task from Chapter 5 was used as the therapy task upon which to perform the comparison. The results indicated that both modalities were capable of providing demonstration data to the LfD algorithms to a satisfactory degree although telerobotic demonstrations had a larger variance on average and were not statistically similar to their single-robot counterparts. The results thereby provide motivation for focusing future work in the area on single-robot implementations of robotic rehabilitation.

In Chapter 7, the manipulator-based single robot rehabilitation system developed throughout the thesis was introduced as a medium for facilitating Functional Capacity Evaluation (FCE) in the field of occupational rehabilitation. The system focused on immersion as a primary objective, incorporating haptic feedback and an augmented-reality display for the practice of a workplace task in a virtual environment. The use of the admittance controller developed in Chapter 5 allowed for the simulation of a 3-DOF painting task. A comparison between a real-world version of the painting task and an equivalent version built in a virtual environment was performed where the similarity of the user's biomechanics provided an evaluation of the degree of realism conveyed by the robotic system. The recorded biomechanics were not statistically similar overall, but, when examined for spatial trends depending on the user's hand position, showed a general trend towards similarity across the entire task workspace. The system showed the potential to provide immersive virtual rehabilitation tasks and can be further refined to provide an innovative alternative to current occupational rehabilitation practices.

8.2 Future Work

8.2.1 Design of a full 7-DOF torque-based impedance controller and exploration of adaptive controllers

One of the most important conclusions derived throughout this thesis is the need for the robot controller, used for physical human-robot interaction (PHRI),

to allow for physical interactions that are as realistic as possible. This means maximum transparency during free motion of the robot (characterized by minimal apparent robot damping and mass as perceived by the user), and high fidelity to realistic environmental interactions (i.e., the ability to quickly exert realistic interaction forces in a stable manner). While the application of an admittance controller as in Chapters 5 and 7 conveniently avoids the need for the full 7-DOF robot dynamics parameters, the use of an impedance controller would still be preferable with regards to achieving realistic interactions. Ideally, a patient-safe robot with a built-in impedance controller would be used in future work. Alternatively, a mathematical formulation of the dynamics matrices needed for the controller can be computed as in Chapter 4. This would require the measurement of the joint masses and centers of gravity, and consideration of redundancy resolution in the controller to account for the seventh DOF (with 6 DOFs being sufficient for providing movement in all Cartesian and rotational axes). In order to achieve maximum transparency during free motion, less conservative controller parameters are needed, but doing so results in a trade-off with system stability that requires a thorough investigation. Adaptive impedance controllers may be considered to address this issue; research on adaptive controllers has itself seen and continues to see a significant amount of interest.

8.2.2 Design of a robotic system for bimanual interaction during therapy tasks

A significant portion of this thesis' work revolves around the development of the presented robotic system's ability to faithfully recreate therapeutic interactions for realistic tasks, whether they be ADLs or FCE tasks. A possible extension to the capabilities of the system would be to allow for the practice of bimanual tasks. Doing so would not only provide a wider array of tasks to practice with but could also create an avenue for therapists to observe and target compensatory motions initiated by the patient's unaffected limb. This would be especially relevant to upper limb post-stroke therapy, where hemiparesis often results in the manifestation of compensatory movements. The

proposed system may take the form of using two manipulator-style robots to interact with a real-world or virtual task but would likely depend on the cost-effectiveness of such a system.

8.2.3 Design of a robotic system for enabling group therapy

This thesis presents a robotic system that is suitable for interaction between a single therapist and a single patient. A common practice in conventional therapy is to engage multiple patients in rehabilitation exercises at the same time, which may be led by a single therapist. This type of rehabilitation session is known as group therapy and has been found to have additional benefits for some participants. Enabling interactions for this kind of therapy in robotic rehabilitation could also provide similar benefits as well as possible cost-savings.

8.2.4 Exploration of LfD algorithms that define models across the task workspace

The use of GMM-based LfD algorithms throughout the thesis was shown to be sufficient, but could certainly be improved upon. Limitations associated with the resolution of the model generated by the Gaussian components were brought to attention specifically in Chapters 3 and 5. A possible future direction would be to explore the use of alternative LfD algorithms that generate global models from demonstrations (i.e., that cover the entire task workspace). This could be performed through simple methods such as polynomial surface fitting, but could also be extended to explore more advanced concepts such as fitting Riemannian manifolds, or the Stable Estimator of Dynamical Systems (SEDS) algorithm (a variant of GMM-based LfD).

8.2.5 Design of a user-friendly editor for creating therapy tasks in virtual environments

The virtual environments presented to the patient in Chapter 7 are, as mentioned, created in the Unity game engine. Using this engine requires a signifi-

cant amount of computer programming knowledge, which runs contrary to one of the thesis' objectives in that programming of rehabilitation robots should be simple and intuitive enough to be used by any clinical therapist. It would be highly beneficial to create a simplified virtual environment creator (even based on the Unity engine), which would include assets (i.e., virtual objects and other resources) representative of common therapy tools and items. The incorporation of depth tracking cameras (such as the Microsoft Kinect used in 7) for recording environmental data of a real-world environment could also be used to create virtual environments quickly. For example, a real-life shelf used in standard FCE lifting tasks could be imaged and virtualized. The weighted crates used in the tasks could then be picked from the prebuilt asset library and placed in the environment as the therapist desires.

8.2.6 Creation of a library of learnt interactions

In this thesis, the LfD method presented requires that the robots be retrained by the therapist every time a patient's performance changes to a level unseen in previous sessions, or if the rehabilitation task changes. To help streamline this process, a library of learnt interactions could be created by compiling demonstrations and reproductions over every session and across any patient. Then if a patient's performance or the task changed, a therapist could determine the most similar set of demonstrations in the library and load the corresponding reproduction data to the robot. Such a system could potentially be developed with online capabilities, allowing therapists to draw upon the experiences and recommended interventions provided by other therapists from around the world for a variety of patient performances and rehabilitation tasks.

8.2.7 Application of the developed system to more clinically relevant tasks and clinical validation

The ultimate goal of this research is to develop a system suitable for clinical application. Towards this end, future developments of the robotic system proposed in this thesis should focus on evaluating the system's ability to provide

interactions appropriate to more basic ADLs such as dressing and grooming for post-stroke therapy, and standard FCE tasks for occupational rehabilitation. It is highly probable that the end-effector design of the robot will have to change in order to incorporate these tasks as a significant number of them revolve around fine motor control and dexterous (i.e., finger-based) movements. The final direction would then be to provide clinical validation with real post-stroke patients or injured workers, as opposed to the simulated patients or healthy participants seen in this thesis.

References

[1] P. Lum *et al.*, "Robotic devices for movement therapy after stroke: Current status and challenges to clinical acceptance," *Topics in Stroke Rehabilitation*, vol. 8, no. 4, pp. 40–53, 2002, pMID: 14523729. [Online]. Available: http://dx.doi.org/10.1310/9KFM-KF81-P9A4-5WW0

[2] Heart and Stroke Foundation of Canada, "Getting to the heart of the matter: solving cardiovascular disease through research," 2015. [Online]. Available: http://www.heartandstroke.ca/-/media/pdf-files/canada/2017-heart-month/heartandstroke-reportonhealth-2015.ashx?la=en

[3] M. Mimouni, K. Cismariu-Potash, M. Ratmansky, S. Shaklai, H. Amir, and A. Mimouni-Bloch, "Trends in physical medicine and rehabilitation publications over the past 16 years," *Archives of Physical Medicine and Rehabilitation*, vol. 97, no. 6, pp. 1030 – 1033, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003999315014100

[4] R. Voelker, "Rehabilitation medicine welcomes a robotic revolution," *JAMA*, vol. 294, no. 10, pp. 1191–1195, 2005. [Online]. Available: +http://dx.doi.org/10.1001/jama.294.10.1191

[5] L. E. Kahn, P. S. Lum, W. Z. Rymer, and D. J. Reinkensmeyer, "Robot-assisted movement training for the stroke-impaired arm: Does it matter what the robot does?" *Journal of rehabilitation research and development*, vol. 43, no. 5, 2014.

[6] M. Iosa, G. Morone, A. Cherubini, and S. Paolucci, "The three laws of neurorobotics: A review on what neurorehabilitation robots should do for patients and clinicians," *Journal of Medical and Biological Engineering*, vol. 36, no. 1, pp. 1–11, Feb 2016. [Online]. Available: https://doi.org/10.1007/s40846-016-0115-2

[7] P. Beckerle, G. Salvietti, R. Unal, D. Prattichizzo, S. Rossi, C. Castellini, S. Hirche, S. Endo, H. B. Amor, M. Ciocarlie, F. Mastrogiovanni, B. D. Argall, and M. Bianchi, "A human–robot interaction perspective on assistive and rehabilitation robotics," *Frontiers in Neurorobotics*, vol. 11, p. 24, 2017. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnbot.2017.00024

[8] E. J. Benjamin *et al.*, "Heart disease and stroke statistics—2017 update: A report from the american heart association," *Circulation*, 2017. [Online]. Available: http://circ.ahajournals.org/content/early/2017/01/25/CIR.0000000000000485

[9] Intercollegiate Stroke Working Party, *National clinical guideline for stroke*. Citeseer, 2012, vol. 20083.

[10] Heart and Stroke Foundation of Canada, "Mind the connection: Preventing stroke and dementia," 2016. [Online]. Available: http://www.heartandstroke.ca/-/media/pdf-files/canada/stroke-report/hsf-stroke-report-2016.ashx?la=en

[11] Public Health Agency of Canada, "Tracking heart disease and stroke in canada 2009." [Online]. Available: https://www.canada.ca/en/public-health/services/reports-publications/2009-tracking-heart-disease-stroke-canada.html

[12] S. C. Cramer, M. Sur, B. H. Dobkin, C. O'brien, T. D. Sanger, J. Q. Trojanowski, J. M. Rumsey, R. Hicks, J. Cameron, D. Chen *et al.*, "Harnessing neuroplasticity for clinical applications," *Brain*, vol. 134, no. 6, pp. 1591–1609, 2011.

[13] S. J. Albert and J. Kesselring, "Neurorehabilitation of stroke," *Journal of neurology*, vol. 259, no. 5, pp. 817–832, 2012.

[14] D. Bourbonnais and S. V. Noven, "Weakness in patients with hemiparesis," *The American journal of occupational therapy*, vol. 43, no. 5, pp. 313–319, 1989.

[15] A. McComas, R. Sica, A. Upton, and N. Aguilera, "Functional changes in motoneurones of hemiparetic patients," *Journal of Neurology, Neurosurgery & Psychiatry*, vol. 36, no. 2, pp. 183–193, 1973.

[16] A. Parton, P. Malhotra, and M. Husain, "Hemispatial neglect," *Journal of Neurology, Neurosurgery & Psychiatry*, vol. 75, no. 1, pp. 13–21, 2004.

[17] A. P. Yelnik *et al.*, "Perception of verticality after recent cerebral hemispheric stroke," *Stroke*, 2002.

[18] V. W. Mark and E. Taub, "Constraint-induced movement therapy for chronic stroke hemiparesis and other disabilities," *Restorative neurology and neuroscience*, vol. 22, no. 3-5, pp. 317–336, 2004.

[19] R. Bracewell, "Stroke: neuroplasticity and recent approaches to rehabilitation," *Journal of Neurology, Neurosurgery & Psychiatry*, vol. 74, no. 11, pp. 1465–1465, 2003.

[20] B. Langhammer and J. K. Stanghelle, "Bobath or motor relearning programme? a comparison of two different approaches of physiotherapy in stroke rehabilitation: a randomized controlled study," *Clinical rehabilitation*, vol. 14, no. 4, pp. 361–369, 2000.

[21] B. Williams, A. Chang, C. S. Landefeld, C. Ahalt, R. Conant, and H. Chen, *Current diagnosis and treatment: geriatrics 2E*. McGraw Hill Professional, 2014.

[22] L. Legg *et al.*, "Occupational therapy for patients with problems in personal activities of daily living after stroke: systematic review of randomised trials," *BMJ*, 2007. [Online]. Available: http://www.bmj.com/content/early/2006/12/31/bmj.39343.466863.55

[23] N. B. Alexander, A. T. Galecki, M. L. Grenier, L. V. Nyquist, M. R. Hofmeyer, J. C. Grunawalt, J. L. Medell, and D. Fry-Welch, "Task-specific resistance training to improve the ability of activities of daily living–impaired older adults to rise from a bed and from a chair," *Journal of the American Geriatrics Society*, vol. 49, no. 11, pp. 1418–1427, 2001.

[24] H. I. Krebs, J. J. Palazzolo, L. Dipietro, M. Ferraro, J. Krol, K. Rannekleiv, B. T. Volpe, and N. Hogan, "Rehabilitation robotics: Performance-based progressive robot-assisted therapy," *Autonomous robots*, vol. 15, no. 1, pp. 7–20, 2003.

[25] H. A. Yanco and K. Z. Haigh, "Automation as caregiver: A survey of issues and technologies," *Am. Assoc. Artif. Intell*, vol. 2, pp. 39–53, 2002.

[26] H. M. Van der Loos, D. J. Reinkensmeyer, and E. Guglielmelli, *Rehabilitation and Health Care Robotics*. Cham: Springer International Publishing, 2016, pp. 1685–1728. [Online]. Available: https://doi.org/10.1007/978-3-319-32552-1_64

[27] H. I. Krebs and N. Hogan, "Therapeutic robotics: A technology push," *Proceedings of the IEEE*, vol. 94, no. 9, pp. 1727–1738, 2006.

[28] D. J. Reinkensmeyer, L. E. Kahn, M. Averbuch, A. McKenna-Cole, B. D. Schmit, and W. Z. Rymer, "Understanding and treating arm movement impairment after chronic brain injury: progress with the arm guide." *Journal of rehabilitation research and development*, 2014.

[29] P. S. Lum, C. G. Burgar, and P. C. Shor, "Evidence for improved muscle activation patterns after retraining of reaching movements with the mime robotic system in subjects with post-stroke hemiparesis," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 12, no. 2, pp. 186–194, 2004.

[30] M. Guidali, A. Duschau-Wicke, S. Broggi, V. Klamroth-Marganska, T. Nef, and R. Riener, "A robotic system to train activities of daily living in a virtual environment," *Medical & Biological Engineering & Computing*, vol. 49, no. 10, p. 1213, July 2011. [Online]. Available: https://doi.org/10.1007/s11517-011-0809-0

[31] J. Mehrholz, A. Hädrich, T. Platz, J. Kugler, and M. Pohl, "Electromechanical and robot-assisted arm training for improving generic activities of daily living, arm function, and arm muscle strength after stroke," *Cochrane database of systematic reviews*, no. 6, 2012.

[32] I. Söderback, *International handbook of occupational therapy interventions*. Springer, 2009.

[33] C. Gowland, P. Stratford, M. Ward, J. Moreland, W. Torresin, S. Van Hullenaar, J. Sanford, S. Barreca, B. Vanspall, and N. Plews, "Measuring physical impairment and disability with the chedoke-mcmaster stroke assessment." *Stroke*, vol. 24, no. 1, pp. 58–63, 1993.

[34] D. J. Gladstone, C. J. Danells, and S. E. Black, "The fugl-meyer assessment of motor recovery after stroke: a critical review of its measurement properties," *Neurorehabilitation and neural repair*, vol. 16, no. 3, pp. 232–240, 2002.

[35] A. Pandyan, G. Johnson, C. Price, R. Curless, M. Barnes, and H. Rodgers, "A review of the properties and limitations of the ashworth and modified ashworth scales as measures of spasticity," *Clinical rehabilitation*, vol. 13, no. 5, pp. 373–383, 1999.

[36] A. A. Blank, J. A. French, A. U. Pehlivan, and M. K. O'Malley, "Current trends in robot-assisted upper-limb stroke rehabilitation: promoting patient engagement in therapy," *Current physical medicine and rehabilitation reports*, vol. 2, no. 3, pp. 184–195, 2014.

[37] A. U. Pehlivan, D. P. Losey, and M. K. O'Malley, "Minimal assist-as-needed controller for upper limb robotic rehabilitation," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 113–124, 2016.

[38] V. Squeri, A. Basteris, and V. Sanguineti, "Adaptive regulation of assistance 'as needed'in robot-assisted motor skill learning and neurorehabilitation," in *2011 IEEE International conference on rehabilitation robotics*. IEEE, 2011, pp. 1–6.

[39] E. T. Wolbrecht, V. Chan, D. J. Reinkensmeyer, and J. E. Bobrow, "Optimizing compliant, model-based robotic assistance to promote neurorehabilitation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 3, pp. 286–297, 2008.

[40] J. H. Ricker, M. Rosenthal, E. Garay, J. DeLuca, A. Germain, K. Abraham-Fuchs, and K.-U. Schmidt, "Telerehabilitation needs: a survey of persons with acquired brain injury," *The Journal of Head Trauma Rehabilitation*, vol. 17, no. 3, pp. 242–250, 2002.

[41] S. F. Atashzar, N. Jafari, M. Shahbazi, H. Janz, M. Tavakoli, R. V. Patel, and K. Adams, "Telerobotics-assisted platform for enhancing interaction with physical environments for people living with cerebral palsy," *Journal of Medical Robotics Research*, vol. 2, no. 02, p. 1740001, 2017.

[42] M. Shahbazi, S. F. Atashzar, M. Tavakoli, and R. V. Patel, "Position-force domain passivity of the human arm in telerobotic systems," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pp. 552–562, 2018.

[43] M. Sharifi, S. Behzadipour, H. Salarieh, and M. Tavakoli, "Cooperative modalities in robotic tele-rehabilitation using nonlinear bilateral impedance control," *Control Engineering Practice*, vol. 67, pp. 52–63, 2017.

[44] C. R. Carignan and H. I. Krebs, "Telerehabilitation robotics: Bright lights, big future?" *Journal of Rehabilitation Research and Development*, vol. 43, no. 5, pp. 695–710, August 2006, copyright - Copyright Superintendent of Documents Aug/Sep 2006; Document feature - Diagrams; Photographs; Illustrations; Last updated - 2017-11-09; CODEN - JRRDDB.

[45] M. J. Johnson, R. C. Loureiro, and W. S. Harwin, "Collaborative tele-rehabilitation and robot-mediated therapy for stroke rehabilitation at home or clinic," *Intelligent Service Robotics*, vol. 1, no. 2, pp. 109–121, 2008.

[46] D. Kairy, P. Lehoux, C. Vincent, and M. Visintin, "A systematic review of clinical outcomes, clinical process, healthcare utilization and costs associated with telerehabilitation," *Disability and rehabilitation*, vol. 31, no. 6, pp. 427–447, 2009.

[47] C. G. Atkeson and S. Schaal, "Robot learning from demonstration," in *ICML*, vol. 97. Citeseer, 1997, pp. 12–20.

[48] B. D. Argall *et al.*, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889008001772

[49] S. F. Atashzar, M. Shahbazi, M. Tavakoli, and R. V. Patel, "A new passivity-based control technique for safe patient-robot interaction in haptics-enabled rehabilitation systems," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, September 2015, pp. 4556–4561.

[50] F. Amirabdollahian, S. Ates, A. Basteris, A. Cesario, J. Buurke, H. Hermens, D. Hofs, E. Johansson, G. Mountain, N. Nasr, and et al., "Design, development and deployment of a hand/wrist exoskeleton for home-based rehabilitation after stroke - script project," *Robotica*, vol. 32, no. 8, pp. 1331–1346, 2014.

[51] R. Sainburg, D. Good, and A. Przybyla, "Bilateral synergy: A framework for post-stroke rehabilitation," *Journal of neurology & translational neuroscience*, vol. 1, no. 3, October 2013. [Online]. Available: http://europepmc.org/articles/PMC3984050

[52] J. R. Beard and D. E. Bloom, "Towards a comprehensive public health response to population ageing," *Lancet (London, England)*, vol. 385, no. 9968, p. 658, 2015.

[53] S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, and A. Kheddar, "Learning collaborative manipulation tasks by demonstration using a haptic interface," in *Advanced Robotics, 2009. ICAR 2009. International Conference on*. IEEE, 2009, pp. 1–6.

[54] E. Gribovskaya, S. M. Khansari-Zadeh, and A. Billard, "Learning non-linear multivariate dynamics of motion in robotic manipulators," *The International Journal of Robotics Research*, vol. 30, no. 1, pp. 80–117, 2011.

[55] L. Peternel, T. Petrič, E. Oztop, and J. Babič, "Teaching robots to cooperate with humans in dynamic manipulation tasks based on multi-modal human-in-the-loop approach," *Autonomous Robots*, vol. 36, no. 1-2, pp. 123–136, January 2014. [Online]. Available: https://doi.org/10.1007/s10514-013-9361-0

[56] M. Maaref, A. Rezazadeh, K. Shamaei, R. Ocampo, and T. Mahdi, "A bicycle cranking model for assist-as-needed robotic rehabilitation therapy using learning from demonstration," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 653–660, July 2016.

[57] A. Lydakis, Y. Meng, C. Munroe, Y.-N. Wu, and M. Begum, "A learning-based agent for home neurorehabilitation," in *Rehabilitation Robotics (ICORR), 2017 International Conference on*. IEEE, July 2017, pp. 1233–1238.

[58] C. Lauretti *et al.*, "Learning by demonstration for planning activities of daily living in rehabilitation and assistive robotics," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1375–1382, July 2017.

[59] M. Najafi, K. Adams, and M. Tavakoli, "Robotic learning from demonstration of therapist's time-varying assistance to a patient in trajectory-following tasks," in *Rehabilitation Robotics (ICORR), 2017 International Conference on*. IEEE, July 2017, pp. 888–894.

[60] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 358, no. 1431, pp. 537–547, 2003. [Online]. Available: http://rstb.royalsocietypublishing.org/content/358/1431/537

[61] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys*, vol. 50, no. 2, pp. 21:1–21:35, Apr. 2017. [Online]. Available: http://doi.acm.org/10.1145/3054912

[62] D. Reynolds, "Gaussian mixture models," *Encyclopedia of Biometrics*, pp. 827–832, 2015.

[63] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Robot Programming by Demonstration*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1371–1394. [Online]. Available: https://doi.org/10.1007/978-3-540-30301-5_60

[64] A. Roby-Brami, A. Feydy, M. Combeaud, E. V. Biryukova, B. Bussel, and M. F. Levin, "Motor compensation and recovery for reaching in stroke patients," *Acta Neurologica Scandinavica*, vol. 107, no. 5, pp. 369–381, 2003. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1034/j.1600-0404.2003.00021.x

[65] C. Kisner, L. A. Colby, and J. Borstad, *Therapeutic exercise: Foundations and techniques*. Fa Davis, 2017.

[66] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, Oct 2011.

[67] C. M. Sackley, "The relationships between weight-bearing asymmetry after stroke, motor function and activities of daily living," *Physiotherapy Theory and Practice*, vol. 6, no. 4, pp. 179–185, 1990.

[68] K.-H. Mauritz, "Gait training in hemiplegia," *European Journal of Neurology*, vol. 9, no. s1, pp. 23–29, 2002. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1468-1331.2002.0090s1023.x

[69] M. R. Schindl *et al.*, "Treadmill training with partial body weight support in nonambulatory patients with cerebral palsy," *Archives of Physical Medicine and Rehabilitation*, vol. 81, no. 3, pp. 301 – 306, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003999300900753

[70] N. E. Mayo *et al.*, "A new approach to retrain gait in stroke patients through body weight support and treadmill stimulation," *Stroke*, 1998.

[71] C. Werner *et al.*, "Treadmill training with partial body weight support and an electromechanical gait trainer for restoration of gait in subacute stroke patients," *Stroke*, 2002.

[72] S. Hussain, S. Q. Xie, and G. Liu, "Robot assisted treadmill training: Mechanisms and training strategies," *Medical Engineering & Physics*, vol. 33, no. 5, pp. 527 – 533, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1350453310002973

[73] I. Díaz, J. J. Gil, and E. Sánchez, "Lower-limb robotic rehabilitation: literature review and challenges," *Journal of Robotics*, vol. 2011, 2011.

[74] W. Meng *et al.*, "Recent development of mechanisms and control strategies for robot-assisted lower limb rehabilitation," *Mechatronics*, vol. 31, pp. 132 – 145, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957415815000501

[75] S. Jezernik *et al.*, "Adaptive robotic rehabilitation of locomotion: a clinical study in spinally injured individuals," *Spinal Cord*, vol. 41, no. 12, p. 657, 2003.

[76] J. Cao *et al.*, "Control strategies for effective robot assisted gait rehabilitation: The state of art and future prospects," *Medical Engineering & Physics*, vol. 36, no. 12, pp. 1555 – 1566, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1350453314002136

[77] R. Garate *et al.*, "Experimental validation of motor primitive-based control for leg exoskeletons during continuous multi-locomotion tasks," *Frontiers in Neurorobotics*, vol. 11, p. 15, 2017.

[78] K. Gui, H. Liu, and D. Zhang, "Toward multimodal human–robot interaction to enhance active participation of users in gait rehabilitation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 11, pp. 2054–2066, 2017.

[79] M. F. Eilenberg, H. Geyer, and H. Herr, "Control of a powered ankle–foot prosthesis based on a neuromuscular model," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 2, pp. 164–173, 2010.

[80] A. R. Wu *et al.*, "An adaptive neuromuscular controller for assistive lower-limb exoskeletons: A preliminary study on subjects with spinal cord injury," *Frontiers in Neurorobotics*, vol. 11, p. 30, 2017.

[81] H. Vallery *et al.*, "Reference trajectory generation for rehabilitation robots: complementary limb motion estimation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 17, no. 1, pp. 23–30, 2009.

[82] V. Rajasekaran, J. Aranda, and A. Casals, "Adaptive walking assistance based on human-orthosis interaction," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on.* IEEE, 2015, pp. 6190–6195.

[83] J. A. Blaya and H. Herr, "Adaptive control of a variable-impedance ankle-foot orthosis to assist drop-foot gait," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 12, no. 1, pp. 24–31, 2004.

[84] C. C. Chen and R. K. Bode, "Factors influencing therapists' decision-making in the acceptance of new technology devices in stroke rehabilitation," *American journal of physical medicine & rehabilitation*, vol. 90, no. 5, pp. 415–425, 2011.

[85] M. Maaref *et al.*, "A gaussian mixture framework for co-operative rehabilitation therapy in assistive impedance-based tasks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 5, pp. 904–913, Aug 2016.

[86] C. Martínez and M. Tavakoli, "Learning and robotic imitation of therapist's motion and force for post-disability rehabilitation," in *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on.* IEEE, 2017, pp. 2225–2230.

[87] M. Najafi *et al.*, "Robotic assistance for children with cerebral palsy based on learning from tele-cooperative demonstration," *International Journal of Intelligent Robotics and Applications*, 01 2017.

[88] R. Begg and J. Kamruzzaman, "A machine learning approach for automated recognition of movement patterns using basic, kinetic and kinematic gait data," *Journal of Biomechanics*, vol. 38, no. 3, pp. 401 – 408, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021929004002258

[89] M. Hansen *et al.*, "Real time foot drop correction using machine learning and natural sensors," *Neuromodulation: Technology at the Neural Interface*, vol. 5, no. 1, pp. 41–53, 2002. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1046/j.1525-1403.2002._2008.x

[90] M. Peshkin *et al.*, "Kineassist: a robotic overground gait and balance training device," in *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, June 2005, pp. 241–246.

[91] J. D. Stewart, "Foot drop: where, why and what to do?" *Practical Neurology*, vol. 8, no. 3, pp. 158–169, 2008. [Online]. Available: https://pn.bmj.com/content/8/3/158

[92] N. Hogan and S. P. Buerger, "Impedance and interaction control," in *Robotics and automation handbook*, T. R. Kurfess, Ed. CRC press, 2004, ch. 19.

[93] J. W. Jeong, P. H. Chang, and K. B. Park, "Sensorless and modeless estimation of external force using time delay estimation: application to impedance control," *Journal of mechanical science and technology*, vol. 25, no. 8, p. 2051, 2011.

[94] R. Tao, "Haptic teleoperation based rehabilitation systems for task-oriented therapy," Master's thesis, University of Alberta, Edmonton, Canada, 2014.

[95] S. Calinon, *Robot programming by demonstration*. EPFL Press, 2009.

[96] M. W. Spong and M. Vidyasagar, *Robot dynamics and control*. John Wiley & Sons, 2008.

[97] R. J. Adams *et al.*, "Assessing upper extremity motor function in practice of virtual activities of daily living," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 2, pp. 287–296, Mar. 2015.

[98] Y. Meng *et al.*, "A learning from demonstration framework to promote home-based neuromotor rehabilitation," in *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2016, pp. 1126–1131.

[99] E. Pignat and S. Calinon, "Learning adaptive dressing assistance from human demonstration," *Robotics and Autonomous Systems*, vol. 93, pp. 61 – 75, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889016303669

[100] W. S. Newman, "Stability and performance limits of interaction controllers," *Journal of Dynamic Systems, Measurement, and Control*, vol. 114, no. 4, p. 563, 1992.

[101] L. Rozo *et al.*, "Learning collaborative impedance-based robot behaviors," in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, ser. AAAI'13. AAAI Press, 2013, pp. 1422–1428. [Online]. Available: http://dl.acm.org/citation.cfm?id=2891460.2891659

[102] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011. [Online]. Available: http://dx.doi.org/10.1163/016918611X558261

[103] H. Seraji, "Adaptive admittance control: an approach to explicit force control in compliant motion," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, May 1994, pp. 2705–2712 vol.4.

[104] F. Dimeas and N. Aspragathos, "Online stability in human-robot cooperation with admittance control," *IEEE Transactions on Haptics*, vol. 9, no. 2, pp. 267–278, April 2016.

[105] S. F. Atashzar *et al.*, "A computational-model-based study of supervised haptics-enabled therapist-in-the-loop training for upper-limb post-stroke robotic rehabilitation," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pp. 563–574, April 2018.

[106] R. Ikeura and H. Inooka, "Variable impedance control of a robot for cooperation with a human," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 3, May 1995, pp. 3097–3102 vol.3.

[107] N. Hogan, H. I. Krebs, J. Charnnarong, P. Srikrishna, and A. Sharon, "Mit-manus: a workstation for manual therapy and training. i," in *[1992] Proceedings IEEE International Workshop on Robot and Human Communication*. IEEE, 1992, pp. 161–165.

[108] T. C. Hsia and R. G. Bonitz, "Force tracking impedance control of robot manipulators under unknown environment," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 3, pp. 474–483, May 2004.

[109] D. P. Gross and M. F. Reneman, *Functional Capacity Evaluation*. New York, NY: Springer New York, 2017, pp. 1–4. [Online]. Available: https://doi.org/10.1007/978-1-4614-6439-6_101935-1

[110] BTE EvalTech. Https://www.btetech.com/product/evaltech/.

[111] Ergos II Work Simulator. http://www.simwork.com/Products/ErgosIIWorkSimulator.aspx.

[112] D. Peppers *et al.*, "Influence of functional capacity evaluation on physician's assessment of physical capacity of veterans with chronic pain: a retrospective analysis," *PM&R*, vol. 9, no. 7, pp. 652–659, 2017.

[113] D. P. Gross *et al.*, "A cluster randomized clinical trial comparing functional capacity evaluation and functional interviewing as components of occupational rehabilitation programs," *Journal of occupational rehabilitation*, vol. 24, no. 4, pp. 617–630, 2014.

[114] D. P. Gross, M. C. Battié, and J. D. Cassidy, "The prognostic value of functional capacity evaluation in patients with chronic low back pain: part 1: timely return to work," *Spine*, vol. 29, no. 8, pp. 914–919, 2004.

[115] F. Schaafsma, E. Schonstein, K. M. Whelan, E. Ulvestad, D. T. Kenny, and J. H. Verbeek, "Physical conditioning programs for improving work outcomes in workers with back pain," *Cochrane Database Syst Rev*, vol. 1, 2010.

[116] C. James, M. Reneman, and D. Gross, "Functional capacity evaluation research: Report from the second international functional capacity evaluation research meeting," *Journal of occupational rehabilitation*, vol. 26, no. 1, pp. 80–83, 2016.

[117] F. Yakub, A. Z. M. Khudzari, and Y. Mori, "Recent trends for practical rehabilitation robotics, current challenges and the future," *International Journal of Rehabilitation Research*, vol. 37, no. 1, pp. 9–21, 2014.

[118] B. Taylor, M. E. Cupo, and S. J. Sheredos, "Workstation robotics: a pilot study of a desktop vocational assistant robot," *American Journal of Occupational Therapy*, vol. 47, no. 11, pp. 1009–1013, 1993.

[119] J. L. Schuyler and R. M. Mahoney, "Assessing human-robotic performance for vocational placement," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 3, pp. 394–404, 2000.

[120] D. J. Reinkensmeyer and S. J. Housman, ""If I can't do it once, why do it a hundred times?": Connecting volition to movement success in a virtual environment motivates people to exercise the arm after stroke," in *2007 Virtual Rehabilitation*, Sept 2007, pp. 44–48.

[121] BTE Eccentron. Https://www.btetech.com/product/eccentron/.

[122] R. Ocampo and M. Tavakoli, "Improving user performance in haptics-based rehabilitation exercises by colocation of user's visual and motor axes via a three-dimensional augmented-reality display," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 438–444, 2019.

[123] H. I. Krebs, N. Hogan, M. L. Aisen, and B. T. Volpe, "Robot-aided neurorehabilitation," *IEEE Transactions on Rehabilitation Engineering*, vol. 6, no. 1, pp. 75–87, 1998.

[124] Unity. Https://unity3d.com/.

[125] B. Jones, R. Sodhi, M. Murdock, R. Mehra, H. Benko, A. Wilson, E. Ofek, B. MacIntyre, N. Raghuvanshi, and L. Shapira, "Roomalive: Magical experiences enabled by scalable, adaptive projector-camera units," in *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '14. New York, NY, USA: ACM, 2014, pp. 637–644.

[126] G. Fasano and A. Franceschini, "A multidimensional version of the kolmogorov–smirnov test," *Monthly Notices of the Royal Astronomical Society*, vol. 225, no. 1, pp. 155–170, 1987.

[127] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, "Numerical recipes in c," *Cambridge University Press*, vol. 1, p. 3, 1988.

Appendix A

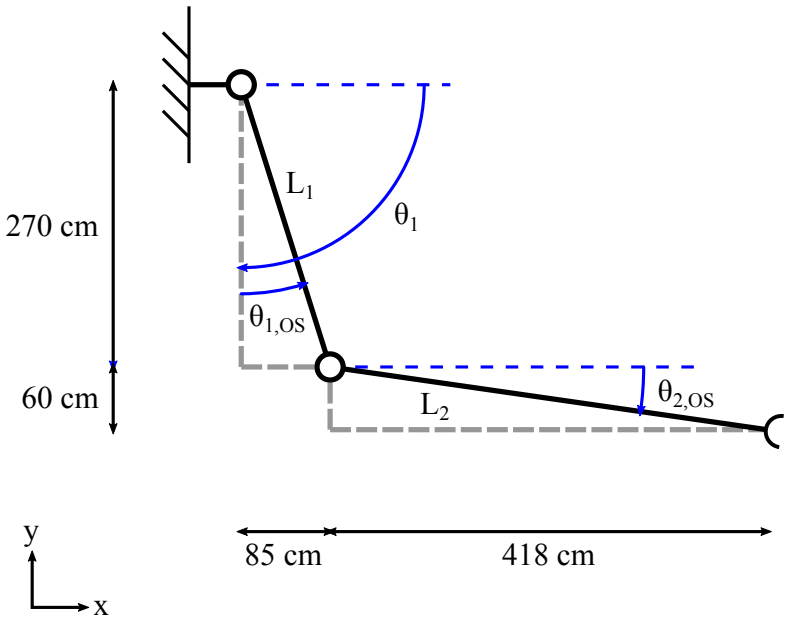
Formulation of 2 DOF Robot Dynamics Parameters for Impedance Controller

Details regarding the formulation of the 2 DOF robot Jacobian J_r and dynamics parameters $\bar{M}_r$ and $\bar{C}_r$ used for the impedance controller in 4 are provided in this Appendix. Fig. A.1 depicts the dimensions and kinematics of the simplified 2 DOF version of the Motoman which serves as the basis for the dynamics calculations.

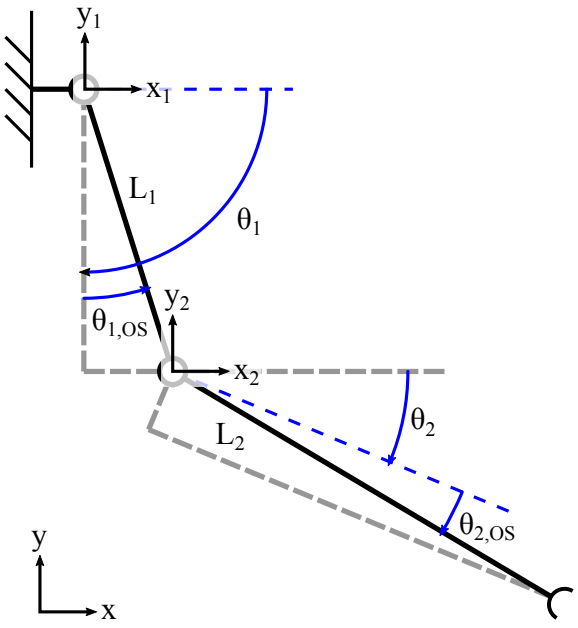
The process described in [96] is then used as follows. The forward kinematics of the centers of the links and their derivatives (i.e., velocities) are given as

$$\begin{aligned}
\begin{bmatrix} x_{c,1} \\ y_{c,1} \end{bmatrix} &= \begin{bmatrix} L_{c,1}\cos(\theta_1 + \theta_{1,OS}) \\ L_{c,1}\sin(\theta_1 + \theta_{1,OS}) \end{bmatrix} \\
\begin{bmatrix} x_{c,2} \\ y_{c,2} \end{bmatrix} &= \begin{bmatrix} L_1\cos(\theta_1 + \theta_{1,OS}) + L_{c,2}\sin(\theta_2 + \theta_{2,OS} - \theta_1) \\ L_1\sin(\theta_1 + \theta_{1,OS}) + L_{c,2}\cos(\theta_2 + \theta_{2,OS} - \theta_1) \end{bmatrix} \\
\begin{bmatrix} \dot{x}_{c,1} \\ \dot{y}_{c,1} \end{bmatrix} &= \begin{bmatrix} -L_{c,1}\sin(\theta_1 + \theta_{1,OS})\frac{d\theta_1}{dt} \\ L_{c,1}\cos(\theta_1 + \theta_{1,OS})\frac{d\theta_1}{dt} \end{bmatrix} \\
\begin{bmatrix} \dot{x}_{c,2} \\ \dot{y}_{c,2} \end{bmatrix} &= \begin{bmatrix} L_1\sin(\theta_1 + \theta_{1,OS}) - L_{c,2}\cos(\theta_2 + \theta_{2,OS} - \theta_1)\frac{d\theta_1}{dt} \\ \quad + L_{c,2}\cos(\theta_2 + \theta_{2,OS} - \theta_1)\frac{d\theta_2}{dt} \\ L_1\cos(\theta_1 + \theta_{1,OS}) + L_{c,2}\sin(\theta_2 + \theta_{2,OS} - \theta_1)\frac{d\theta_1}{dt} \\ \quad - L_{c,2}\sin(\theta_2 + \theta_{2,OS} - \theta_1)\frac{d\theta_2}{dt} \end{bmatrix}
\end{aligned} \tag{A.1}$$

where $L_{c,1}$ and $L_{c,2}$ describe the centers of mass for the 2 links. Computations of the end-effector positions and velocities can be performed by simply



(a)



(b)

Figure A.1: Robot dimensions and kinematics diagram for 2 DOF simplification. (a) provides dimensions associated with links L_1 and L_2, which are rigid combinations of the Motoman robot's links 1, 2, and 3, and links 4, 5, 6, and 7 respectively. (b) depicts the joint angles θ_1 and θ_2 as reported by the robot's built-in controller. The angle offsets $\theta_{1,OS}$ and $\theta_{2,OS}$ describe the difference between the reported joint angles and those of the 2 DOF version of the robot. The frames shown in (b) indicate the joint frames used by the built-in controller to report the current joint angles. For example, the configuration shown in (a) corresponds to when $\theta_1 = -90°$ and $\theta_2 = 0$.

repeating the same calculations for $x_{c,2}$ and $y_{c,2}$, but with $L_{c,1}$ and $L_{c,2}$ replaced with L_1 and L_2:

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} L_1\cos(\theta_1+\theta_{1,OS}) + L_2\sin(\theta_2+\theta_{2,OS}-\theta_1) \\ L_1\sin(\theta_1+\theta_{1,OS}) + L_2\cos(\theta_2+\theta_{2,OS}-\theta_1) \end{bmatrix}$$
$$\begin{bmatrix} \dot{x}_2 \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} L_1\sin(\theta_1+\theta_{1,OS}) - L_2\cos(\theta_2+\theta_{2,OS}-\theta_1)\frac{d\theta_1}{dt} \\ \quad +L_2\cos(\theta_2+\theta_{2,OS}-\theta_1)\frac{d\theta_2}{dt} \\ L_1\cos(\theta_1+\theta_{1,OS}) + L_2\sin(\theta_2+\theta_{2,OS}-\theta_1)\frac{d\theta_1}{dt} \\ \quad -L_2\sin(\theta_2+\theta_{2,OS}-\theta_1)\frac{d\theta_2}{dt} \end{bmatrix} \tag{A.2}$$

The relationship between Cartesian and joint space velocity is given by the Jacobian through the equation $\vec{v} = J\dot{\vec{\theta}}$. Using (A.1), the Jacobians for each link's center of mass can be found as

$$J_{c,1} = \begin{bmatrix} -L_{c,1}\sin(\theta_1+\theta_{1,OS}) & 0 \\ L_{c,1}\cos(\theta_1+\theta_{1,OS}) & 0 \end{bmatrix}$$
$$J_{c,2} = \begin{bmatrix} -L_1\sin(\theta_1+\theta_{1,OS}) & L_{c,2}\cos(\theta_2+\theta_{2,OS}-\theta_1) \\ -L_{c,2}\cos(\theta_2+\theta_{2,OS}-\theta_1) & \\ L_1\cos(\theta_1+\theta_{1,OS}) & -L_{c,2}\sin(\theta_2+\theta_{2,OS}-\theta_1) \\ +L_{c,2}\sin(\theta_2+\theta_{2,OS}-\theta_1) & \end{bmatrix} \tag{A.3}$$

The Jacobian describing the end-effector's movement J_r can be computed in the same way using (A.2), but is almost equivalent to $J_{c,2}$ computed in (A.3) and is therefore not included for brevity. The kinetic energy from translational components of the robot's movement is then given by

$$K_v = \frac{1}{2}\dot{\vec{\theta}}(m_1 J_{c,1}^T J_{c,1} + m_2 J_{c,2}^T J_{c,2})\dot{\vec{\theta}} \tag{A.4}$$

where m_1 and m_2 refer to the masses of each of the 2 (combined) links. In the experimental configuration shown in Chapter 4 (similar to Fig. A.1) and with respect to how the controller reports velocities, the kinetic energy from angular components of the robot's movement is given by

$$K_\omega = \frac{1}{2}\dot{\vec{\theta}}^T\left(I_1\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix} + I_2\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\right)\dot{\vec{\theta}} \tag{A.5}$$

where I_1 and I_2 refer to the I_{33} moment of inertia term (since rotation is in the x-y plane only) of the inertia tensors for the 2 links. The robot inertia matrix can then be computed as

$$\bar{M}_r(\vec{\theta}) = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} = m_1 J_{c,1}^T J_{c,1} + m_2 J_{c,2}^T J_{c,2} + \begin{bmatrix} I_1 + I_2 & -I_2 \\ -I_2 & I_2 \end{bmatrix}$$
$$= \begin{bmatrix} m_2 L_1^2 + 2 m_2 L_1 L_{c,2} \sin(\theta_2' + \theta_{1,OS}) & -m_2 L_1 L_{c,2} \sin(\theta_2' + \theta_{1,OS}) \\ +m_1 L_{c,1}^2 + m_2 L_{c,2}^2 + I_1 + I_2 & +m_2 L_{c,2}^2 - I2 \\ -m_2 L_1 L_{c,2} \sin(\theta_2' + \theta_{1,OS}) & m_2 L_{c,2}^2 + I2 \\ -m_2 L_{c,2}^2 - I2 & \end{bmatrix} \tag{A.6}$$

where $\theta_2' = \theta_2 + \theta_{2,OS}$ is introduced for brevity. The Christoffel symbols are then computed as follows:

$$\begin{aligned} c_{111} &= \frac{1}{2}\frac{\partial m_{11}}{\partial \theta_1} \\ c_{121} = c_{211} &= \frac{1}{2}\frac{\partial m_{11}}{\partial \theta_2} \\ c_{221} &= \frac{\partial m_{12}}{\partial \theta_2} - \frac{1}{2}\frac{\partial m_{22}}{\partial \theta_1} \\ c_{112} &= \frac{\partial m_{21}}{\partial \theta_1} - \frac{1}{2}\frac{\partial m_{11}}{\partial \theta_2} \\ c_{122} = c_{212} &= \frac{1}{2}\frac{\partial m_{22}}{\partial \theta_1} \\ c_{222} &= \frac{1}{2}\frac{\partial m_{22}}{\partial \theta_2} \end{aligned} \tag{A.7}$$

The elements of the Coriolis and centrifugal matrix $\bar{C}_r$ can then be found, with the k, j-th element of the matrix computed as

$$c_{kj} = \sum_{i=1}^{n} c_{ijk}(\dot{\vec{\theta}})\dot{\theta}_i \tag{A.8}$$

where, in the experimental setup used in 4, the matrix is given as

$$\bar{C}_r(\vec{\theta}, \dot{\vec{\theta}}) = \begin{bmatrix} m_2 L_1 L_{c,2} \cos(\theta_2' + \theta_{1,OS})\dot{\theta}_2 & m_2 L_1 L_{c,2} \cos(\theta_2' + \theta_{1,OS})(\dot{\theta}_1 - \dot{\theta}_2) \\ -m_2 L_1 L_{c,2} \cos(\theta_2' + \theta_{1,OS})\dot{\theta}_1 & 0 \end{bmatrix} \tag{A.9}$$

Appendix B

Overall Layout of Presented Robotic Systems

A high-level block diagram of all electronic or robotic components used in this thesis is presented in this appendix. Fig. B.1 depicts the interconnected components and the data passed between each component.

The base system is comprised of two PCs (named the Quanser PC, which contains the QUARC Simulink software, and the Agile Planet PC, which contains a pre-installed WinCE OS to interface with the Agile Planet servo controller) and the Yaskawa Motoman SIA5F 7 DOF manipulator. The admittance controller in Chapters 5 and 7, as well as other control or diagnostic components (shown in Appendix C) are implemented in the QUARC Simulink environment on the Quanser PC. The impedance controller used in Chapters 4 and 6 is implemented as a C++ application on the Agile Planet PC's WinCE installation. The Agile Planet PC, ATI Gamma Net force sensor, ClaroNav Micron Tracker camera, and Unity environment incorporated throughout the thesis all communicate with the Quanser PC through a built-in QUARC UDP communication application. Note that the Quanser HD^2 used in Chapters 3 and 6 does not need to interface with the UDP application, as its controller is integrated directly in the QUARC software.

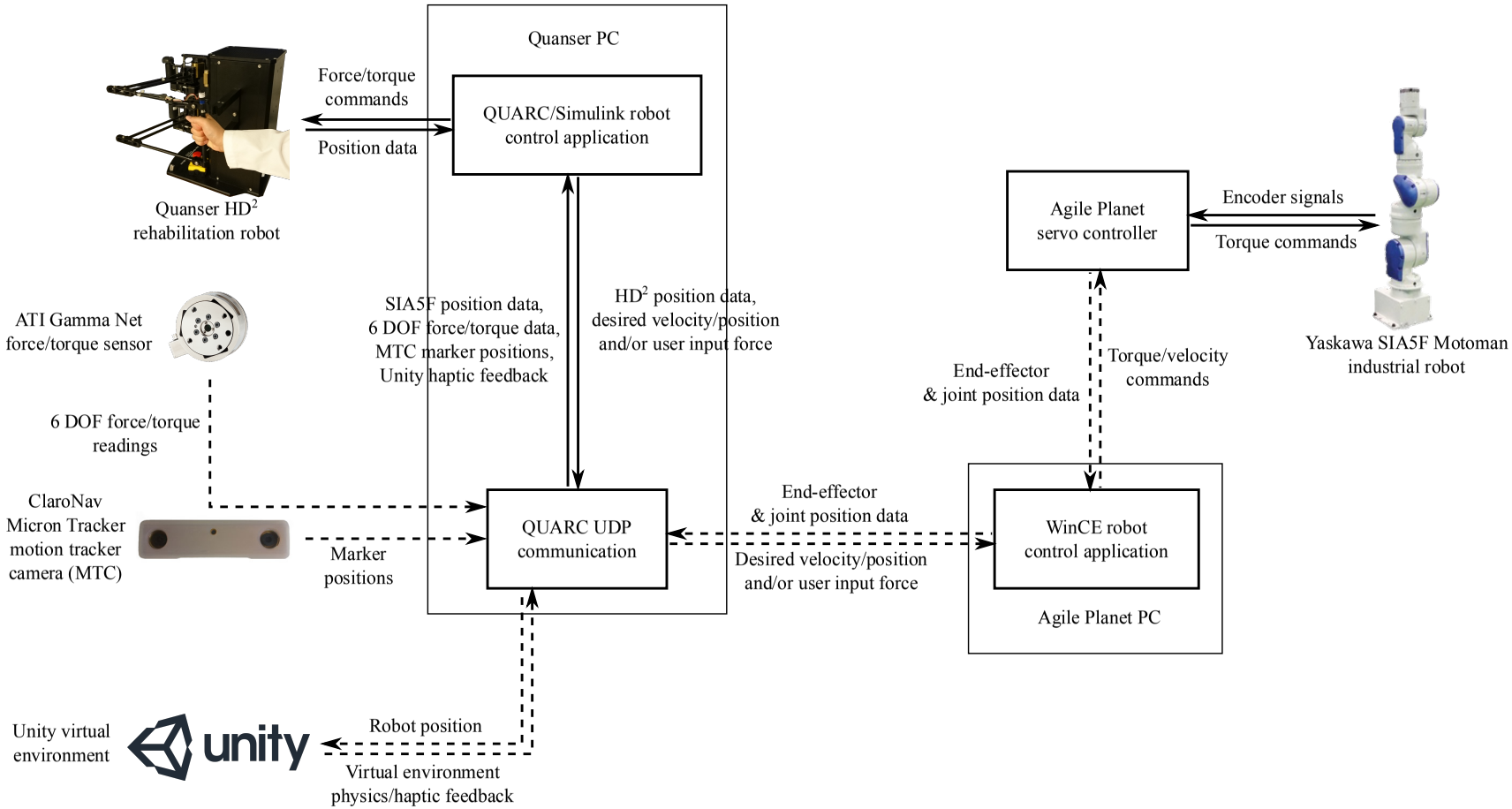


Figure B.1: Overall system layout of sensors, robots, and data signal transmissions. Solid lines indicate direct communication lines (i.e., through software or through attached wiring) while dashed lines indicate communication over ethernet.

Appendix C

Simulink Model Component Architecture

Example Simulink models for the impedance and admittance control schemes, as well as important model components, are shown in this appendix. Fig. C.1 shows the overall model used for the impedance control of the robot in Chapter 4, and Fig. C.2 shows the overall model used for admittance control in Chapter 7. The Laplace domain representation of the admittance controller transfer function is shown in Fig. C.3, and lastly the Simulink-side UDP communications implementation is shown in Fig. C.4.

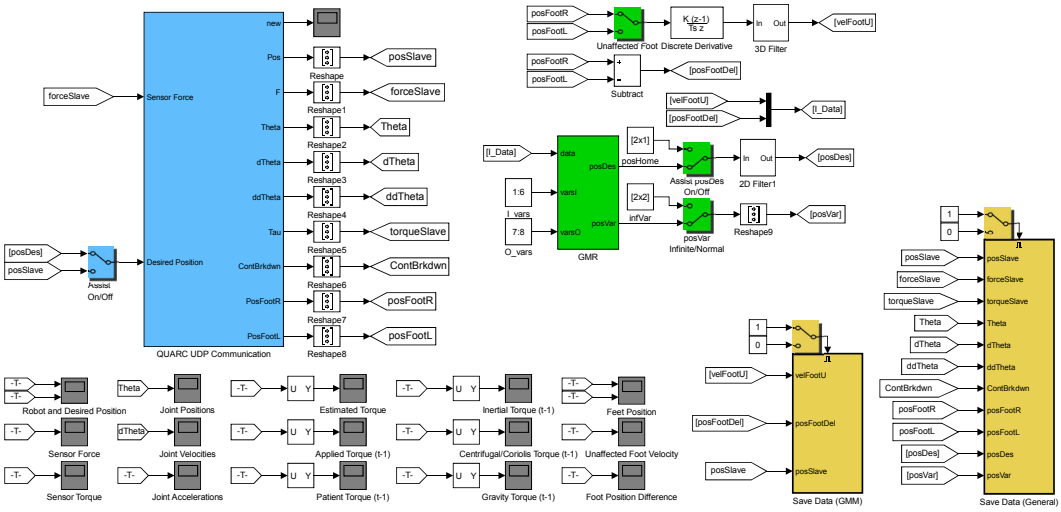


Figure C.1: Simulink block diagram used for impedance control in Chapter 4. Communications architecture for UDP data transmission between Simulink and either the Motoman robot or the ClaroNav MTC is colored blue, and is contained in a subsystem. The Gaussian Mixture Regression algorithm is colored green, as well as the blocks that configure its input data. Data recording blocks are yellow and signal displays (for the purpose of troubleshooting) are grey.

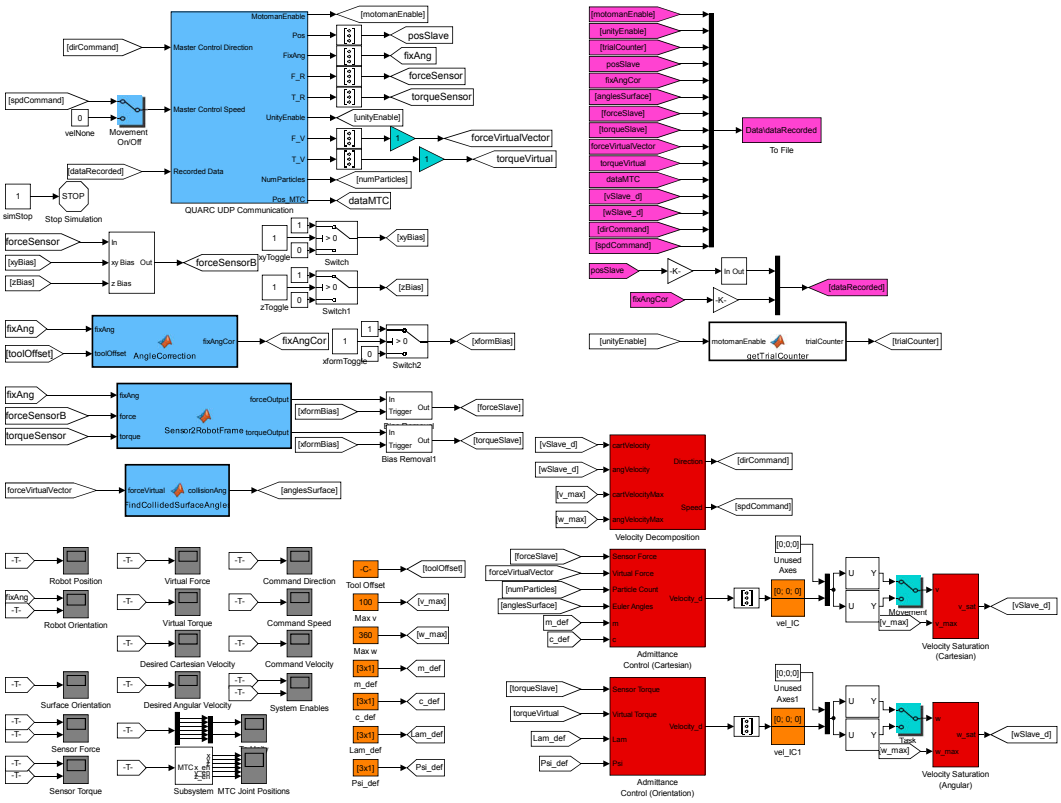
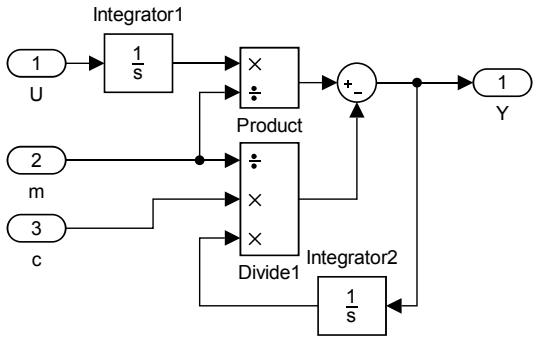


Figure C.2: Simulink block diagram used for admittance control in Chapter 7. Communications architecture for UDP data transmission between Simulink and the Motoman robot, ClaroNav MTC, and Unity environment is colored blue and is contained in the top left subsystem. Transformations of the force sensor readings from the sensor frame to the robot (world) frame are performed in the blue function blocks below the communications subsystem. The admittance control architecture is colored red and contained within separate subsystems for Cartesian and angular movements. Parameters for the admittance controller are colored orange. Data recording blocks are purple, and signal displays are again grey.



H(s) = 1 / (ms + c)
Y(s) * (1 + c/ms) = U(s) * (1/ms)
Y(s) = U(s) * (1/ms) - Y(s) * (c /ms)

Figure C.3: Transfer function that facilitates admittance control as in Chapters 5 and 7. Derivation of the LTI system is provided below the model implementation.

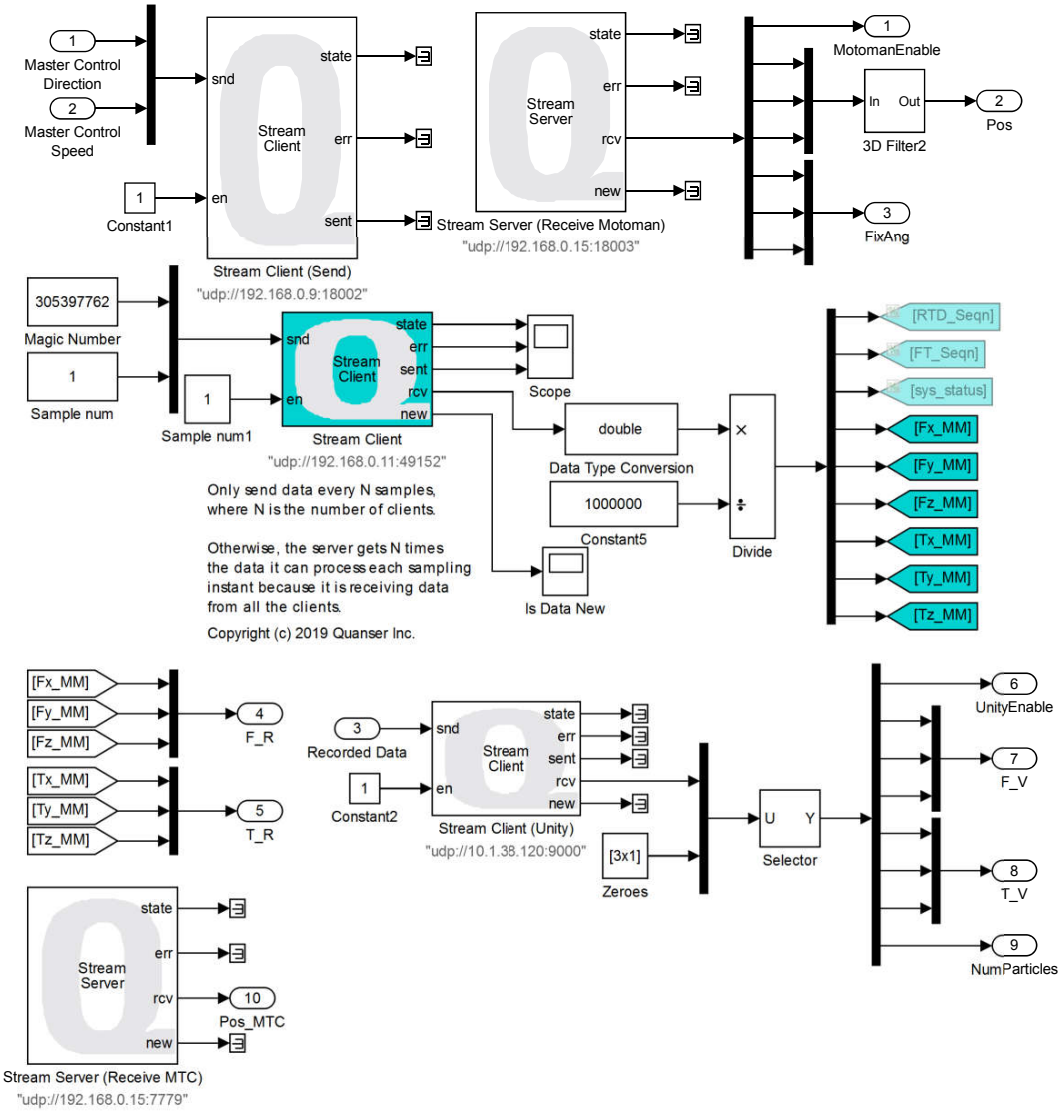


Figure C.4: Communications architecture for UDP data transmission, in this example specifically for Chapter 7. All UDP streams are facilitated through the use of blocks provided by the QUARC software (Quanser Inc., Markham, Ontario, Canada).

Appendix D

Matlab Codes for Learning from Demonstration and C++ Code for Motion Tracker Camera

Samples of the Matlab codes used to implement the LfD algorithms presented in this thesis are provided in this appendix. An example of the C++ code used to interface with the ClaroNav MTC is also provided.

D.1 Basic GMM and GMR Implementation

The following section of code shows how the GMM algorithm, publicly provided in [95], is implemented in Chapter 4. In this code, the demonstration datasets are first loaded and downsampled, before being used as the input to train the GMM (performed with the functions *init_GMM_kmeans* and *EM_GMM*).

```
%% Initialization
clear;
clc;
close all;
addpath('./m_fcts/');

%% Parameters
model.nbStates = 9; % Number of states in the GMM
model.nbVar = 8; % Number of variables
nbData = 100; % Length of each trajectory
nbSamples = 3; % Number of demonstrations

%% Data preparation and GMM parameters
loadstr = {
    'Training 1\',...
    'Training 2\',...
    'Training 3\',...
    'Training 4\',...
    'Training 5\'
```

```
    };
Data = zeros(model.nbVar, nbSamples * nbData);
for i = 1:nbSamples
    % Load individual demo
    load(['.\Data\', char(loadstr(i)), 'DataGMM (Cut)']);
    % Downsample and filter data
    trDataRS = spline(1:size(DataGMMCut, 2), DataGMMCut(2:1 + model.nbVar, :)
        ,...
        linspace(1, size(DataGMMCut, 2), nbData));
    % Concatenate demonstration data
    Data(:, nbData * (i - 1) + 1:nbData * i) = trDataRS;
end

%% Mixture model parameters estimation
model = init_GMM_kmeans(Data, model);
model = EM_GMM(Data, model); % Rows of h correspond to ith
    % Gaussian component
% Plot GMM and training data
figure; view(3); hold on;
plotGMM3D(model.Mu(1:3, :), model.Sigma(1:3, 1:3, :), [0.8, 0, 0], 0.3);
plot3(Data(1, :), Data(2, :), Data(3, :), 'b');

%% Cleanup
clearvars -EXCEPT model
save('Data\GMM_Current');
```

After the GMM has been generated offline, GMR is performed online (i.e., during sessions of human-robot-interaction) in the Simulink environment by calling the following code:

```
function [outputs, output_var] = GMR_2DOF (data, I_vars, O_vars, model)
%#codegen
    [outputs, output_var, ~] = GMR(model, data, I_vars, O_vars);
end
```

D.2 Weighted Least Squares Estimation

In Chapter 5, the statistical encoding of the learned impedance is performed using Weighted Least Squares (WLS) estimation. The code provided below shows how the data is first imported and downsampled, and then used with WLS estimation.

```
%% Assisted Task Data and Modelling

%% Data preparation and GMM parameters
loadstr = {'Test1\', 'Test2\', 'Test3\'};
Data = zeros(6, nbSamples * nbData);
xEnd = zeros(3, nbSamples);
for i = 1:nbSamples
    % Load individual demo
    load(['.\Data\', char(loadstr(i)), 'kinSlave']);
    load(['.\Data\', char(loadstr(i)), 'forcetorqueSlave']);
    trData = [kinSlave(2:4, :); forcetorqueSlave(2:4, :)];
    % Downsample and filter data
    trDataRS = spline(1:size(trData, 2), trData, linspace(1, ...
        size(trData, 2), nbData));
    % Concatenate demonstration data
```

```
    Data(:, nbData * (i - 1) + 1:nbData * i) = trDataRS;
    xEnd(:, i) = trDataRS(1:3, end);
end
xEndAve = mean(xEnd, 2);
DataA = Data;

%% Compute activation weights using previous model
[~, h] = computeImpGamma(Data(1:3, :), model);
% Plot GMM and training data
figure; view(3); hold on;
plotGMM3D(model.Mu(1:3, :), model.Sigma(1:3, 1:3, :), [0.8, 0, 0], 0.3);
plot3(Data(1, :), Data(2, :), Data(3, :), 'b');
plot3(xEndAve(1), xEndAve(2), xEndAve(3), 'go');

%% Gaussian component impedance (stiffness) estimation for
X_A = (repmat(xEndAve, 1, nbSamples * nbData) - Data(1:3, :))';
F_A = Data(4:6, :)';
% K_A_est = zeros(3, 3, model.nbStates);
K_A = zeros(3, 3, model.nbStates);
for i = 1:model.nbStates
    for j = 1:3
        % Weighted Least Squares (WLS) estimation
        W = diag(h(i, :));
        K_A(j, j, i) = (X_A(:, j)' * W * X_A(:, j)) \ X_A(:, j)' * W * F_A(:,
            j);
    end
end

%% Unassisted Task Data and Modelling

%% Data preparation and GMM parameters
loadstr = {'Test7\', 'Test8\', 'Test9\'};
Data = zeros(6, nbSamples * nbData);
for i = 1:nbSamples
    % Load individual demo
    load(['.\Data\', char(loadstr(i)), 'kinSlave']);
    load(['.\Data\', char(loadstr(i)), 'forcetorqueSlave']);
    trData = [kinSlave(2:4, :); forcetorqueSlave(2:4, :)];
    % Downsample and filter data
    trDataRS = spline(1:size(trData, 2), trData, linspace(1, ...
        size(trData, 2), nbData));
    % Concatenate demonstration data
    Data(:, nbData * (i - 1) + 1:nbData * i) = trDataRS;
end
DataNA = Data;

%% Compute activation weights using previous model
[~, h] = computeImpGamma(Data(1:3, :), model);
% Plot GMM and training data
figure; view(3); hold on;
plotGMM3D(model.Mu(1:3, :), model.Sigma(1:3, 1:3, :), [0.8, 0, 0], 0.3);
plot3(Data(1, :), Data(2, :), Data(3, :), 'b');
plot3(xEndAve(1), xEndAve(2), xEndAve(3), 'go');

%% Gaussian component impedance (stiffness) estimation for
X_NA = (repmat(xEndAve, 1, nbSamples * nbData) - Data(1:3, :))';
F_NA = Data(4:6, :)';
% K_NA_est = zeros(3, 3, model.nbStates);
K_NA = zeros(3, 3, model.nbStates);
for i = 1:model.nbStates
    for j = 1:3
        % Weighted Least Squares (WLS) estimation
        W = diag(h(i, :));
        K_NA(j, j, i) = (X_NA(:, j)' * W * X_NA(:, j)) \ X_NA(:, j)' * W *
            F_NA(:, j);
    end
end

```

```
%% Linear estimate of therapist's assistive force (spring constant)
K_T = zeros(3, 3, model.nbStates);
for i = 1:model.nbStates
    K_T(:, :, i) = K_A(:, :, i) - K_NA(:, :, i);
end

%% Cleanup
clearvars -EXCEPT nbData nbSamples model xEndAve K_A K_NA K_T
save('GMM_Current');
```

The learned impedance is then used online to generate assistive forces in the Simulink environment by calling the following code:

```
function forceT = ImpForce(Data)
%#codegen
    paramGMM = coder.load('GMM_Current');
    f = zeros(3, 3, paramGMM.model.nbStates); % Need to change this depending
         on # axes
    h = computeImpGamma(Data, paramGMM.model);
    for i = 1:paramGMM.model.nbStates
        f(:, :, i) = h(i) * paramGMM.K_T(:, :, i) / sum(h);
    end
    forceT = sum(f, 3) * (paramGMM.xEndAve - Data);
end
```

D.3 Motion Tracker Code

The C++ code used to interface with the ClaroNav MTC is provided below:

```
#include "stdafx.h"
#include "data_transmission.h"
#include "config.h"
#include <conio.h>
#include <math.h>
#include <time.h>
#include <MTC.h> // MTC.h need to be in the local directory or include path

// Macro to check for and report MTC usage errors.
#define MTC(func) { int r = func; };

#ifdef WIN32
        int getMTHome(char *sMTHome, int size); // Forward declaration
#endif


int _tmain(int argc, _TCHAR* argv[], char* envp[])
{
        printf("\n-----------------------------------------------");
        printf("\n");
        printf("\nTelerobotic and Biorobotic Systems Lab");
        printf("\nMicronTracker UDP streaming server");
        printf("\n");
        printf("\n-----------------------------------------------");
        printf("\n");

        // Connect to the available cameras, and report on what was found
        // The first camera is designated as the "current" camera - we will
            use
        // its coordinate space in reporting pose measurements.
        char MTHome[512];
        char calibrationDir[512];
        char markerDir[512];
```

```
33
34          #ifdef WIN32
35                  if (getMTHome(MTHome, sizeof(MTHome)) < 0) {
36                          // No Environment
37                          printf("MTHome environment variable is not set!\n");
38                          return -1;
39                  }
40                  else {
41                          sprintf_s(calibrationDir, "%s\\CalibrationFiles",
                                MTHome);
42                          sprintf_s(markerDir, "%s\\Markers", MTHome);
43                  }
44          #else  // Linux & Mac OSX
45                  sprintf(calibrationDir, "../CalibrationFiles");
46                  sprintf(markerDir, "../Markers");
47          #endif
48
49          // Create an instance of data_transmission class
50          data_transmission transmission;
51
52          // Read first command-line argument (name of config file)
53          string name_config_st = "MicronTracker_udp_streaming_server.cfg";
54          if (argc > 1) {
55                  name_config_st = argv[1];
56          }
57
58          Config config(name_config_st, envp);
59
60          string tmp = config.pString("ip_local").c_str();
61          char ip_local_scp[1024];
62          strcpy_s(ip_local_scp, tmp.c_str());
63          tmp = config.pString("ip_remote").c_str();
64          char ip_remote_scp[1024];
65          strcpy_s(ip_remote_scp, tmp.c_str());
66          short port_remote_ss = (short)config.pInt("port_remote");
67          short port_local_ss = (short)config.pInt("port_local");
68          tmp = config.pString("name_markers").c_str();
69          char marker_names_scp[1024];
70          strcpy_s(marker_names_scp, tmp.c_str());
71          short num_markers = (short)config.pInt("num_markers");
72
73          printf("\nLocal IP address: %s:%d", ip_local_scp, port_local_ss);
74          printf("\nRemote IP address: %s:%d", ip_remote_scp, port_remote_ss);
75          printf("\nSupplied markers: %s", marker_names_scp);
76
77          int comm_error = 0;
78          comm_error = transmission.init_transmission(ip_local_scp,
                port_local_ss,
79                  ip_remote_scp, port_remote_ss);
80
81          if (comm_error) {
82                  printf("\nTransmission init failed with error code %d",
                        comm_error);
83                  printf("\n");
84                  system("Pause");
85
86                  return -1;
87          }
88
89          // Back to camera initialization
90          MTC(Cameras_AttachAvailableCameras(calibrationDir)); // Path to
                directory where the calibration files are
91          if (Cameras_Count() < 1) {
92                  printf("\nNo camera found!");
93                  //return 0;
94          }
95          mtHandle CurrCamera, IdentifyingCamera;
96          int      CurrCameraSerialNum;
```

```
97          // Obtain a handle to the first/only camera in the array
98          MTC(Cameras_ItemGet(0, &CurrCamera));
99          // Obtain its serial number
100         MTC(Camera_SerialNumberGet(CurrCamera, &CurrCameraSerialNum));
101         printf("\nAttached %d camera(s). Curr camera is %d", Cameras_Count(),
                CurrCameraSerialNum);
102
103         // Load the marker templates (with no validation).
104         MTC(Markers_LoadTemplates(markerDir)); // Path to directory where the
                marker templates are
105         printf("\nLoaded %d marker templates", Markers_TemplatesCount());
106
107         // Create objects to receive the measurement results
108         mtHandle IdentifiedMarkers = Collection_New();
109         mtHandle t2m = Xform3D_New(); // tooltip to marker xform handle
110         mtHandle m2c = Xform3D_New(); // marker to camera xform handle
111         mtHandle PoseXf = Xform3D_New();
112
113         //-=-=-=- DATA CAPTURE AND TRANSMISSION -=-=-=-//
114         const short len_data_ss = 4;
115         double *data = (double *)malloc(num_markers * len_data_ss * sizeof(
                double));
116         if (data == NULL) exit(1);
117         for (int i = 0; i < num_markers * len_data_ss; i++) {
118                 data[i] = 0.0;
119         }
120
121         int i = 0;
122
123         char marker_names_tmp[1024];
124         char *marker_names_tok;
125         const char *delim = " ,";
126         char *next_token;
127
128         clock_t start;
129         clock_t end;
130         float seconds;
131         float sample_time = (1 / (float)20);
132
133         printf("\nPreparing to stream to %s:%d... \nPress any key to quit.",
                ip_remote_scp, port_remote_ss);
134         while ((!_kbhit()) && (!comm_error)) {
135                 start = clock();
136
137                 MTC(Cameras_GrabFrame(NULL)); // Grab a frame (all cameras
                        together)
138                 MTC(Markers_ProcessFrame(NULL)); // Process the frame(s) to
                        obtain measurements
139
140                 if (i < 40) {
141                         i++;
142                         if (i == 40) {
143                                 printf("\nStreaming now.");
144                         }
145                         continue; // The first 20 frames are auto-adjustment
                                frames, and would be ignored here
146                 }
147
148                 // Here, MTC internally maintains the measurement results.
149                 // Those results can be accessed until the next call to
                        Markers_ProcessFrame, when they
150                 // are updated to reflect the next frame's content.
151                 // First, we will obtain the collection of the markers that
                        were identified.
152                 MTC(Markers_IdentifiedMarkersGet(NULL, IdentifiedMarkers));
153                 // printf("%d: identified %d marker(s)\n", i,
                        Collection_Count(IdentifiedMarkers));
154
```

```
155                 // Now we iterate on the identified markers (if any), and
                        report their name and their pose
156                 for (int j = 1; j <= Collection_Count(IdentifiedMarkers); j
                        ++) {
157                     // Obtain the marker's handle, and use it to obtain
                            the pose in the current camera's space
158                     // using our Xform3D object, PoseXf.
159                     mtHandle Marker = Collection_Int(IdentifiedMarkers, j
                            );
160                     MTC(Marker_Marker2CameraXfGet(Marker, CurrCamera,
                            PoseXf, &IdentifyingCamera));
161
162                     // We check the IdentifyingCamera output to find out
                            if the pose is, indeed,
163                     // available in the current camera space. If
                            IdentifyingCamera==0, the current camera's
164                     // coordinate space is not registered with any of the
                             cameras which actually identified
165                     // the marker.
166                     if (IdentifyingCamera != 0) {
167                         char MarkerName[MT_MAX_STRING_LENGTH];
168                         double  Position[3], Angle[3];
169                         // We will also check and report any
                                measurement hazard
170                         mtMeasurementHazardCode Hazard;
171                         MTC(Marker_NameGet(Marker, MarkerName,
                                MT_MAX_STRING_LENGTH, 0));
172
173                         // Get m2c
174                         MTC(Marker_Marker2CameraXfGet(Marker,
                                CurrCamera, m2c, &IdentifyingCamera));
175                         // Get t2m
176                         MTC(Marker_Tooltip2MarkerXfGet(Marker, t2m));
177                         // Transform both to PoseXf
178                         MTC(Xform3D_Concatenate(t2m, m2c, PoseXf));
179
180                         // Get position
181                         MTC(Xform3D_ShiftGet(PoseXf, Position));
182                         // Here we obtain the rotation as a sequence
                                of 3 angles. Often, it is more convenient
183                         // (and slightly more accurage) access the
                                rotation as a 3x3 rotation matrix.
184                         MTC(Xform3D_RotAnglesDegsGet(PoseXf, &Angle
                                [0], &Angle[1], &Angle[2]));
185                         MTC(Xform3D_HazardCodeGet(PoseXf, &Hazard));
186
187                         // Send position data if identified marker
                                matches specified one
188                         strcpy_s(marker_names_tmp, marker_names_scp);
189                         marker_names_tok = strtok_s(marker_names_tmp,
                                 delim, &next_token);
190                         int ind_marker = 0;
191                         while (marker_names_tok != NULL) {
192                             if (!strcmp(MarkerName,
                                    marker_names_tok)) {
193                                 for (int k = 0; k <
                                        len_data_ss - 1; k++) {
194                                     data[k + ind_marker *
                                            len_data_ss] =
                                        Position[k];
195                                 }
196                                 data[(ind_marker + 1) *
                                        len_data_ss - 1] = 1.0;
197                                 break;
198                             }
199                             ind_marker++;
200                             marker_names_tok = strtok_s(NULL,
                                    delim, &next_token);
```

```
201                                 }
202                         }
203                 }
204                 comm_error = transmission.send(data, num_markers *
                        len_data_ss);
205                 for (int j = 0; j < num_markers; j++) {
206                         data[(j + 1) * len_data_ss - 1] = 0.0;
207                 }
208
209                 end = clock();
210                 seconds = (float)(end - start) / CLOCKS_PER_SEC;
211                 if (seconds > sample_time)
212                         printf("\nSample time > %2.4f s: %2.4f", sample_time,
                                seconds);
213         }
214
215         //-=-=-=- CLEANUP -=-=-=-//
216         // Free up all resources taken
217         free(data);
218
219         Collection_Free(IdentifiedMarkers);
220         Xform3D_Free(PoseXf);
221         Cameras_Detach(); // Important - otherwise the cameras will continue
                capturing, locking up this process.
222
223         printf("\n");
224         system("Pause");
225
226         return 0;
227 }
```

Appendix E

Matlab scripts for Kolmogorov-Smirnov Test-based Analyses

Analysis methods involving the Kolmogorov-Smirnov analysis are provided in this appendix. The simpler 1D KS test performed in Chapter 4 is shown below, where the built-in Matlab function *kstest2* performs the 2 sample version.

```
%% Initialization
clear;
clc;
close all;
addpath('./export_fig_m_fcts/');

%% Parameters
nbSamples = 3;
nbData = 5000;

%% Load and organize toe clearance data
toeclrRD = []; % Right foot during therapist-assisted demonstrations
toeclrLD = []; % Left foot during therapist-assisted demonstrations
toeclrRB = []; % Right foot during baseline unassisted sessions
toeclrLB = []; % Left foot during baseline unassisted sessions
toeclrRI = []; % Right foot during robot-assisted reproductions
toeclrLI = []; % Left foot during robot-assisted reproductions

% Code for loading data removed for brevity...

%% Perform iterative Kolmogorov-Smirnov test
ksLTRD = zeros(nbData, 2);
ksLTLD = zeros(nbData, 2);
ksLTRB = zeros(nbData, 2);
ksLTLB = zeros(nbData, 2);
ksLTRI = zeros(nbData, 2);
% 1D KS test to compare between left foot robot-assisted reproduction data
% and all other recorded datasets for each foot
for i = 1:nbData
        [ksLTRD(i, 1), ksLTRD(i, 2)] = kstest2(toeclrLI(:, i), toeclrRD(:, i)
            );
        [ksLTLD(i, 1), ksLTLD(i, 2)] = kstest2(toeclrLI(:, i), toeclrLD(:, i)
            );
        [ksLTRB(i, 1), ksLTRB(i, 2)] = kstest2(toeclrLI(:, i), toeclrRB(:, i)
            );
```

```
        [ksLTLB(i, 1), ksLTLB(i, 2)] = kstest2(toeclrLI(:, i), toeclrLB(:, i)
            );
        [ksLTRI(i, 1), ksLTRI(i, 2)] = kstest2(toeclrLI(:, i), toeclrRI(:, i)
            );
end
```

The 2 sample 3D KS test performed in Chapter 7 is also provided below.

```
%% 3D KS test over all MAR data and NN physical data
disp('3D KS test (whole dataset)');
[ZnHE, DHE, rrHE] = kstest3d(dataHENNPhys', dataHEMAR');
[ZnES, DES, rrES] = kstest3d(dataESNNPhys', dataESMAR');
n1 = size(dataHENNPhys, 2);
n2 = size(dataHEMAR, 2);
n = (n1 * n2) / (n1 + n2);
% Compute significance levels
disp(['HE:', num2str(kstest3dSL(n, ZnHE, rrHE))]);
disp(['ES:', num2str(kstest3dSL(n, ZnES, rrES))]);

%% 3D KS test (voxel based)
% Generate ranges of hand positions in each axis
minx = min(dataHandleProj(1, :));
maxx = max(dataHandleProj(1, :));
miny = min(dataHandleProj(2, :));
maxy = max(dataHandleProj(2, :));
minz = min(dataHandleProj(3, :));
maxz = max(dataHandleProj(3, :));

xrange = floor(minx / clusterRes) * clusterRes:clusterRes:ceil(maxx /
    clusterRes) * clusterRes;
yrange = floor(miny / clusterRes) * clusterRes:clusterRes:ceil(maxy /
    clusterRes) * clusterRes;
zrange = floor(minz / clusterRes) * clusterRes:clusterRes:ceil(maxz /
    clusterRes) * clusterRes;

% Generate voxel center points with desired resolution
refGrid = zeros(3, length(xrange) * length(yrange) * length(zrange));
for k = 1:length(zrange)
    for j = 1:length(yrange)
        for i = 1:length(xrange)
            pointInd = i + length(xrange) * ((j - 1) + length(yrange) * (k -
                1));
            refGrid(:, pointInd) = [xrange(i); yrange(j); zrange(k)];
        end
    end
end

% Biomechanics data
dataJoints = [dataHandleProj;
        dataHandle;
    dataElbow;
    dataShoulder;
    knnsearch(refGrid', dataHandleProj')';
    dataSetup];
% Output matrix for KS tests
dataSim = zeros(3, size(refGrid, 2));

for i = 1:size(refGrid, 2)
        % Select all datapoints within cluster i
    clustData = dataJoints(:, dataJoints(13, :) == i);
        % If over 2000 points in cluster, select 2000 points with hand
        % positions closest to voxel center
        if size(clustData, 2) > 2000
                clustData = clustData(:, knnsearch(clustData(1:3, :)',
                    refGrid(:, i)', 'k', 2000));
        end
        % Separate data into Motoman-AR data and real world task data
```

```
        clustData1 = clustData(:, clustData(14, :) == 1);
        clustData2 = clustData(:, clustData(14, :) == 2);
        % Determine if cluster meets minimum number of datapoints for
            comparison
        if (size(clustData1, 2) > 19) && (size(clustData2, 2) > 19)
                dataHE1 = clustData1(4:6, :) - clustData1(7:9, :);
                dataHE2 = clustData2(4:6, :) - clustData2(7:9, :);
                dataES1 = clustData1(7:9, :) - clustData1(10:12, :);
                dataES2 = clustData2(7:9, :) - clustData2(10:12, :);
                if (size(unique(dataHE1', 'rows'), 1) > 4 && size(unique(
                    dataHE2', 'rows'), 1) > 4 && ...
                                size(unique(dataES1', 'rows'), 1) > 4 && size
                                    (unique(dataES2', 'rows'), 1) > 4)
                        [ZnHE, DHE, rrHE] = kstest3d(dataHE1', dataHE2');
                        [ZnES, DES, rrES] = kstest3d(dataES1', dataES2');
                        n1 = size(dataHE1, 2);
                        n2 = size(dataHE2, 2);
                        n = (n1 * n2) / (n1 + n2);
                        % Compute significance levels
                        dataSim(1, i) = kstest3dSL(n, ZnHE, rrHE);
                        dataSim(2, i) = kstest3dSL(n, ZnES, rrES);
                        dataSim(3, i) = 1;
                else
                        % Invalid clusters are regarded as dissimilar
                        dataSim(1, i) = 100;
                        dataSim(2, i) = 100;
                end
        else
                % Invalid clusters are regarded as dissimilar
                dataSim(1, i) = 100;
                dataSim(2, i) = 100;
        end
end
```

The implementation provided in [126] is split into two functions here. The first, *kstest3d*, computes the average (required for the 2 sample KS test) of the largest absolute difference between the cumulative frequency distributions of the entire dataset and the data for each condition (i.e., robotic vs real-world experimental setups). The correlation coefficient of the entire dataset is also computed.

```
function [Zn, D, rr] = kstest3d(s1, s2)

assign_point = false; % Set true to assign center point to maximizing
    quadrant
                      % Leave this false if you want FF's original procedure

[n1, m1] = size(s1);
[n2, m2] = size(s2);

if ~all([m1, m2] == 3)
   error('# of columns in X and Y must equal 3');
end

D1 = zeros(n1, 8);
count1 = 0;
for i = 1:n1
   count1 = count1 + 1;
   [a1, b1, c1, d1, e1, f1, g1, h1] = ...
            octCount(s1(i, 1), s1(i, 2), s1(i, 3), s1, n1 - 1);
   [a2, b2, c2, d2, e2, f2, g2, h2] = ...
```

```
            octCount(s1(i, 1), s1(i, 2), s1(i, 3), s2, n2);

    temp = abs([a1 - a2, b1 - b2, c1 - c2, d1 - d2, ...
            e1 - e2, f1 - f2, g1 - g2, h1 - h2]);
    if assign_point
       % Assign point to quadrant where it maximizes difference
       ind = find(max(temp));
       if length(ind) >= 1
          ind = ind(1); % Take first maximum
          temp(ind) = temp(ind) + 1 / length(s1);
       end
    end
    D1(count1, :) = temp;
end
D2 = zeros(n2, 8);
count2 = 0;
for i = 1:n2
    count2 = count2 + 1;
    [a1, b1, c1, d1, e1, f1, g1, h1] = ...
            octCount(s2(i, 1), s2(i, 2), s2(i, 3), s1, n1);
    [a2, b2, c2, d2, e2, f2, g2, h2] = ...
            octCount(s2(i, 1), s2(i, 2), s2(i, 3), s2, n2 - 1);

    temp = abs([a1 - a2, b1 - b2, c1 - c2, d1 - d2, ...
            e1 - e2, f1 - f2, g1 - g2, h1 - h2]);
    if assign_point
       % Assign point to quadrant where it maximizes difference
       ind = find(max(temp));
       if length(ind) >= 1
          ind = ind(1); % take first maximum
          temp(ind) = temp(ind) + 1/length(s2);
       end
    end
    D2(count2, :) = temp;
end

D1 = max(max(D1));
D2 = max(max(D2));
D = mean([D1, D2]);

% Average correlation coefficients
r1 = corrcoef(s1); rxy1 = r1(1, 2); rxz1 = r1(1, 3); ryz1 = r1(2, 3);
r2 = corrcoef(s2); rxy2 = r2(1, 2); rxz2 = r2(1, 3); ryz2 = r2(2, 3);
r1 = mean([rxy1, rxz1, ryz1]);
r2 = mean([rxy2, rxz2, ryz2]);
rr = sqrt(0.5 * (r1 * r1 + r2 * r2));

Zn = probks(n1, n2, D);

%%
% Count fractions of points in s in quadrants defined around point (x,y).
%       s is a nx3 matrix
%       Output in x-y plane for top/bottom half of z axis respectively:
%        a|b   e|f
%       ----- -----
%        c|d   g|h
%
%       Currently, the point x, y is not counted in any fraction
function [a, b, c, d, e, f, g, h] = octCount(x, y, z, s, dnorm)

slx = s(:, 1) < x;
sgx = s(:, 1) > x;
sly = s(:, 2) < y;
sgy = s(:, 2) > y;
slz = s(:, 3) < z;
sgz = s(:, 3) > z;

inda = slx & sgy & sgz;
```

```
indb = sgx & sgy & sgz;
indc = slx & sly & sgz;
indd = sgx & sly & sgz;
inde = slx & sgy & slz;
indf = sgx & sgy & slz;
indg = slx & sly & slz;
indh = sgx & sly & slz;

a = sum(inda) / dnorm;
b = sum(indb) / dnorm;
c = sum(indc) / dnorm;
d = sum(indd) / dnorm;
e = sum(inde) / dnorm;
f = sum(indf) / dnorm;
g = sum(indg) / dnorm;
h = sum(indh) / dnorm;

%----- Asymptotic Q-function to approximate the 3-sided Zn value
function lambda = probks(n1, n2, D)

% Numerical Recipes in C, section 14.7
N = (n1 * n2) / (n1 + n2);
lambda = (sqrt(N) * D);
```

These parameters are then used as the inputs to *kstest3dSL*, which performs a lookup into the tables of Monte Carlo simulations provided by [126] in order to determine the KS test's statistical significance level for the compared distributions.

```
function SL = kstest3dSL(nExp, ZnExp, CCExp)

SLind = [30, 40, 50, 60, 70, 80, 90, 95, 99];
nind = [10, 20, 50, 100, 200, 500];
CCind = [0, 0.5, 0.6, 0.7, 0.8, 0.9];
Zn3D = zeros(length(nind), length(SLind), length(CCind));
Zn3D(:, :, 1) = ...
        [0.992 1.05 1.10 1.16 1.22 1.3 1.42 1.53 1.75;
    1.02 1.07 1.13 1.19 1.26 1.34 1.47 1.57 1.8;
    1.06 1.12 1.18 1.24 1.31 1.39 1.52 1.63 1.84;
    1.12 1.17 1.23 1.29 1.36 1.44 1.56 1.68 1.88;
    1.15 1.21 1.27 1.33 1.41 1.48 1.6 1.71 1.88;
    1.22 1.28 1.33 1.39 1.46 1.54 1.66 1.76 1.95];
Zn3D(:, :, 2) = ...
        [0.96 1.01 1.07 1.12 1.19 1.27 1.39 1.51 1.74;
    0.98 1.03 1.09 1.15 1.22 1.30 1.43 1.55 1.78;
    1.02 1.08 1.14 1.2 1.27 1.37 1.5 1.61 1.86;
    1.07 1.12 1.18 1.24 1.32 1.41 1.54 1.67 1.9;
    1.11 1.17 1.22 1.29 1.36 1.45 1.58 1.7 1.93;
    1.17 1.24 1.29 1.36 1.42 1.5 1.62 1.72 1.94];
Zn3D(:, :, 3) = ...
        [0.94 0.99 1.05 1.11 1.17 1.26 1.38 1.49 1.72;
    0.96 1.01 1.07 1.13 1.2 1.29 1.42 1.53 1.78;
    1 1.06 1.12 1.18 1.25 1.34 1.47 1.58 1.83;
    1.04 1.1 1.16 1.22 1.3 1.38 1.52 1.63 1.86;
    1.09 1.15 1.2 1.26 1.33 1.42 1.56 1.66 1.91;
    1.15 1.20 1.26 1.32 1.39 1.47 1.61 1.72 1.91];
Zn3D(:, :, 4) = ...
        [0.914 0.97 1.02 1.09 1.15 1.24 1.36 1.48 1.72;
    0.927 0.985 1.04 1.1 1.17 1.26 1.39 1.51 1.76;
    0.97 1.02 1.08 1.14 1.22 1.31 1.44 1.55 1.8;
    1.01 1.06 1.12 1.19 1.25 1.34 1.48 1.59 1.83;
    1.05 1.1 1.16 1.22 1.29 1.37 1.52 1.62 1.86;
    1.11 1.16 1.21 1.27 1.33 1.42 1.54 1.64 1.9];
Zn3D(:, :, 5) = ...
```

```
        [0.89 0.642 1 1.06 1.12 1.21 1.33 1.45 1.66;
    0.893 0.946 1 1.06 1.13 1.22 1.36 1.48 1.74;
    0.929 0.982 1.04 1.1 1.17 1.26 1.4 1.52 1.79;
    0.96 1.01 1.07 1.13 1.2 1.29 1.43 1.56 1.83;
    0.99 1.05 1.16 1.23 1.32 1.32 1.46 1.59 1.83;
    1.06 1.11 1.16 1.22 1.29 1.37 1.48 1.6 1.85];
Zn3D(:, :, 6) = ...
        [0.846 0.895 0.953 1.01 1.08 1.16 1.3 1.41 1.63;
    0.844 0.896 0.947 1.01 1.08 1.17 1.31 1.44 1.7;
    0.867 0.917 0.968 1.03 1.1 1.19 1.33 1.47 1.75;
    0.89 0.943 0.997 1.06 1.13 1.22 1.37 1.51 1.77;
    0.926 0.974 1.03 1.09 1.15 1.24 1.4 1.52 1.83;
    0.958 1.01 1.07 1.13 1.19 1.28 1.41 1.53 1.84];

[SLMesh, nMesh, CCMesh] = meshgrid(SLind, nind, CCind);
Znq = interp3(SLMesh, nMesh, CCMesh, Zn3D, ...
        SLind, nExp * ones(size(SLind)), CCExp * ones(size(SLind)), 'spline')
            ;
SL = interp1(Znq, SLind, ZnExp, 'spline', 'extrap');
```