Haptic Teleoperation with Impedance Control based on Learned Inverse Dynamics with Application in Homecare Robotics

by

Muhammad Tufail

M.A.Sc., Asian Institute of Technology, Thailand, 2007

B.Sc., University of Engineering and Technology, Pakistan, 2004

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Mechanical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

December 2015

Abstract

Bilateral teleoperation allows a human operator to interact with a remote environment using the superior actuation and sensing skills of a robot and the unmatched cognitive skills of a human operator. It has shown promising results in applications such as telemedicine, telesurgery, and access to hazardous or remote environments. In all of these applications, the robot has to co-exist with humans and other delicate objects in the environment and therefore has to behave in a compliant ("soft") manner. Moreover, in order to improve the task performance, the interaction force must be fed back to the operator to feel. In this backdrop, the present thesis focuses on the application of bilateral teleoperation in a homecare environment.

In view of the underlying challenges involved with bilateral teleoperation, this dissertation focuses on the development of a complete teleoperation system that can effectively perform in real-time. A primary objective here is to use the impedance control approach to design local controllers for master and slave manipulators where the dynamic relationship between the applied forces and the resulting positions of the manipulators during interaction, is controlled. Impedance control requires the identification of the robot inverse dynamic model that can be computed in real-time and can adapt to changes in the actual dynamics of the robot. A complete data-driven learning-based technique called Locally Weighted Projection Regression (LWPR) is therefore used, which does not assume any a-priori knowledge of the inertial parameters of the robot. Performance of the system is improved by using online estimation of impedance of the unknown environment with which the slave manipulator interacts. A method of admittance control is designed. This method overcomes the shortcomings of the standard impedance control, as observed during experimentation. In the end, a method is developed to improve the transparency and position synchronization of the popular approach of wave-variables, which ensures stability under time delay that is induced by the communication channel during the exchange of information between the master and the slave ends. The effectiveness of the present developments is validated in an environment of homecare robotics, through simulation and experimentation, and the results are discussed.

Preface

The work presented in this thesis was conducted at the Industrial Automation Laboratory (IAL), Department of Mechanical Engineering, The University of British Columbia (UBC), Vancouver and was supervised by Prof. Clarence W. de Silva. The thesis topic, objectives, and the application were suggested by Prof. de Silva. My responsibility was to develop the system models and algorithms as well as experimentation with guidance and help from my supervisor, Prof. de Silva.

Part of this work has been published as:

1. M. Tufail, C.W. de Silva, "Haptic Teleoperation using Impedance Control with Application in Homecare Robotics," *Control and Intelligent Systems*, vol. 41, no. 3, 2013.
2. M. Tufail, C. W. de Silva, "Impedance Control Schemes for Bilateral Teleoperation", in *10th IEEE International Conference on Computer Science & Education (ICCSE)*, Vancouver, Canada, August 22-24 2014.
3. M. Tufail, C.W. de Silva, "Teleoperation using Impedance Control for Homecare Robotics," *The IASTED International Conference on Engineering and Applied Science (EAS 2012)*, Colombo, Sri Lanka, December 27-29, 2012.

In all the listed published work which is based on Chapter 3 of the thesis, I developed the control schemes and performed numerical simulation and experimentation. The use of impedance control in haptic teleoperation and their application in homecare were suggested by Prof. de Silva. He provided guidance and feedback in carrying out the activities. I wrote the manuscripts, which were edited and refined by Prof. de Silva.

Table of Contents

List of Tables

List of Figures

List of Symbols

a	Link Length
A	Homogeneous Transformation Matrix
b	Wave Impedance
b_e	Damping of the Environment Surface
B	Joint Space Friction Matrix
B_D	Desired Damping Matrix
C	Joint Space Coriolis/Centrifugal Matrix
C_x	Cartesian Space Coriolis/Centrifugal Matrix
d	Link Offset
D	Distance Metric of Gaussian Kernel
f_m^d	Desired Force for the Master Manipulator
F_{ext}	External Force
G	Joint Space Gravity Matrix
G_x	Cartesian Space Gravity Matrix
H	Hybrid Matrix
I_i	Inertia Tensor of Link *i*
J	Manipulator Jacobian Matrix
J_i	Jacobian Matrix of Link *i*
J^+	Jacobian Pseudo-Inverse
k_e	Stiffness of the Environment Surface
K_θ	Joint Stiffness Matrix
K_p	Proportional Gain in Cartesian PD control
K_D	Desired Stiffness Matrix
M	Joint Space Inertia Matrix
M_D	Desired Inertia Matrix

M_x	Cartesian Space Inertia Matrix
o_{i-1}^{i}	Origin of Frame *i*-1 Expressed in Frame *i*
P	Error Covariance Matrix
q	Joint Position
R	Rotation Matrix
S	Scattering Operator
T_{i-1}^{i}	Composite Transformation Matrix Representing Frame *i-1* in Frame *i*
τ	Joint Torque
u	Forward Wave Variable
v	Backward Wave Variable
w	Weight of Gaussian Kernel
x	Cartesian *x*-Position
x_d	Desired Cartesian *x*-Position
x_e	Environment Location (*x*-Position)
x_s^d	Desired Velocity for the Slave Manipulator
$\dot{x}$	Cartesian *x*-Velocity
X	Cartesian Position Vector
X_a	Adjusted Position Vector
Y	Regressor Matrix
$\tilde{y}$	Prediction of output *y*
Z_{to}	Impedance Presented to the Operator
α	Twist Angle
θ	Joint variable
ω	Cartesian Angular Velocity
π_i	Dynamic Coefficients Vector of Link *i*
η	Gaussian Noise
μ_k	Centre of Gaussian Kernel
β	Regressor Coefficients

Φ	Regressor Matrix (in Parameter Estimation)
λ	Forgetting Factor

List of Abbreviations

ANN	Artificial Neural Networks
API	Application Programming Interface
ARMAX	Autoregressive Moving Average eXogeneous
BCKDF	British Columbia Knowledge Development Fund
CFI	Canada Foundation for Innovation
CoM	Center of Mass
DOF	Degrees of Freedom
EoM	Equation of Motion
FoGS	The Faculty of Graduate Studies
GPR	Gaussian Process Regression
HEC	Higher Education Commission
IAL	Industrial Automation Laboratory
ICT	Information and Communications Technology
LWPR	Locally Weighted Projection Regression
MSN	Master-Slave-Network
nMSE	Normalized Mean Squared Error
NSERC	Natural Sciences and Engineering Research Council
PLSR	Partial Least Squares Regression
RF	Receptive Field
RLSR	Recursive Least Squares Regression
ROS	Robot Operating System
SVR	Support Vector Regression
TCP	Tool Centre Point
TDPC	Time-Domain Passivity Control
UBC	University of British Columbia
UDP	User Datagram Protocol
WAM	Whole Arm Manipulator

Acknowledgments

I wish to acknowledge and thank my supervisor, Prof. Clarence W. de Silva for supervising my thesis research. I feel honored to have remained under the supervision and in company of such a learned, knowledgeable, kind, and supportive personality. Throughout the course of my Ph.D. Research, I have learned extensively from him and by reading his well-written and comprehensive textbooks, research publications, talks, discussions at group meeting, and even his inspirational speeches that he has made on various occasions. His presence in my life has always encouraged me to work harder as I feel there is always someone who can steer me through difficult situations, if they arise.

I thank the members of my research committee, Dr. Mu Chiao, Dr. Mohammad Gadala, and Dr. Ryozo Nagamune, whose thoughtful feedback has led to improvement of my thesis.

The research reported in this thesis has been supported by research grants to Prof. Clarence W. de Silva from the Natural Sciences and Engineering Research Council (NSERC), the Canada Foundation for Innovation (CFI), British Columbia Knowledge Development Fund (BCKDF), the Tier 1 Canada Research Chair in Mechatronics and Industrial Automation, and the scholarship to me from Higher Education Commission (HEC) of the Government of Pakistan through my home university, University of Engineering & Technology, Peshawar. I wholeheartedly thank them for their support.

Special thanks to my colleagues at the Industrial Automation Laboratory (IAL), Dr. Mohammad Tahir Khan, Dr. Rolang Lang, Dr. Edward (Yanjun) Wang, Mr. Yunfei Zhang and all others. It was due to their continuous support and friendship that I have enjoyed my stay at UBC. Special thanks to Dr. Khan for introducing me to Dr. de Silva. This was the best gift a senior could ever give to his colleague.

A very special thanks to my parents for their love and encouragement, especially my mom who has always pushed me to work hard.

Finally, and most importantly, I would like to express my love and appreciation to my wife, Summiya, and son, Hayyan, who have been very supportive and patient throughout. I know at times it became unbearable for them to accept my busy schedule. Without your support this thesis would not have been possible. Thank you!

Chapter 1: Introduction

1.1 Motivation

The rapidly increasing elderly population in our society is a challenging trend which is expected to accelerate in the coming years. Between 2006 and 2011 the population of Canadian seniors aged 65 saw a record high increase of 14.1% and reached nearly 5 million [1]. As of July 1, 2015, seniors accounted for 16.1% of the national population of Canada [2]. For the first time in the history of Canadian census, there were more persons aged 65 years and older than children aged 0 to 14 years. This number is projected to reach 20.1% by 2024, which means one in five Canadians will be aged 65 or over at that time. For the Canadian workforce, it will have several negative consequences. For example, it will be hard to provide adequate healthcare services to meet the growing demand for the care of the elderly. According to the Canadian Federation of Nurses Unions, a shortfall of 60,000 nurses is expected by 2022 [3]. Due to the recent attitudes towards service robots and the increasing value of the global market for technology products related to the care of older persons (anticipated to grow to $4 billion by 2015 [4]) there is a great need for developing innovative technologies that can provide homecare including healthcare to the elderly and the disabled. Reference [5] highlights the successful projects as well as the shortcomings of Information and Communications Technology (ICT) services for digital homecare.

Our focus here is to develop haptic assisted, semi-autonomous robots in a home setting where they can carry out basic service and survival tasks such as serving food and medicine, cleaning, bathing, and providing assistance for mobility. The development of a teleoperation system with multimodal feedback (e.g., haptic, verbal and visual) of these service robots from a hospital control room will further improve the efficacy of the solution. Together, the two modes of operations of the robot: autonomous and semi-autonomous with teleoperation, can enhance an older person’s ability to function as independently as possible in a natural (home) environment and call for emergency help as and when needed.

One of the ways that can reduce the spending on healthcare of older people is to exploit the recent technological advancements in robotics and information and communication technologies (ICT) for providing high-quality supportive environments for older people in their homes. As a practical direction to the proposed research, a robotic homecare environment that will have autonomous robots can be augmented by the haptic teleoperation capability comprising a haptic-assisted remotely controlled robot to monitor and assist individuals within the home environment. The design challenges for such sophisticated application of robots include dealing with the instability due to communication constraints such as time delay and ensuring transparent manipulation of the remote environment through force feedback. A focus of this thesis is to formulate a bilateral control architecture that addresses these challenges.

1.2 Problem Formulation

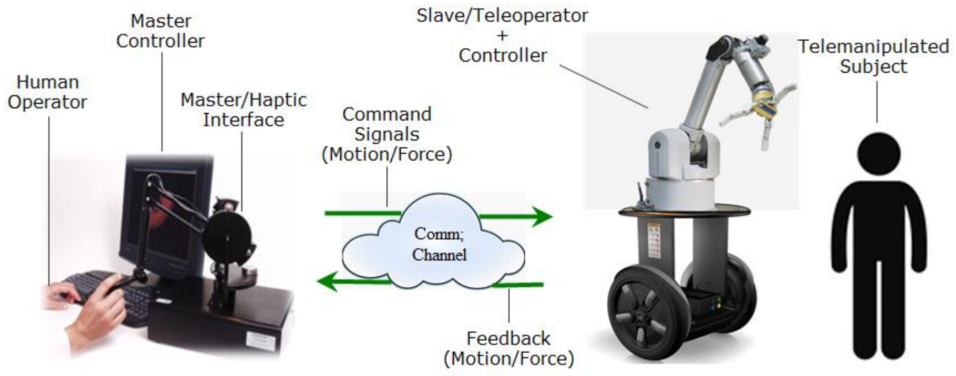

Figure 1.1 Bilateral teleoperation system.

In a bilateral teleoperation system, as shown in Figure 1.1, a human operator with the help of a master manipulator (a *haptic interface*) interacts remotely with an environment (a human in a homecare environment) through a slave manipulator that is usually located at distance. The human operator inputs motion (such as position and velocity) and force into the system, which is sent over a communication channel to the slave manipulator. The communication channel incurs time delay in any information that is sent through it. The slave manipulator tracks the motion received from the master and interacts with the environment accordingly. The slave manipulator also measures or estimates the interaction force between its end effector and the environment. It then sends the force estimate back to the master side where the haptic interface allows the human operator to feel the interaction force and adjusts his hand motion accordingly. This also closes the control loop at the master side. The feedback of force as well as any other information such as visual, tactile etc. is intended to improve the system performance and enhance transparency.

In haptic teleoperation the emphasis is more on the force feedback as the term *haptic* refers to the sense of touch. A teleoperation system is called *transparent* if the human operator receives an authentic feel of the remote interaction. In particular, under transparent conditions, the operator will not feel any unwanted dynamics added to the interaction force. The *teleoperator*, a subsystem that comprises master/slave manipulators and the communication channel, must not contribute to modifying the feel of the interaction force negatively and therefore deteriorating the transparency. This means the teleoperator must act like a massless and infinitely stiff connection between the human operator and the environment. In addition to the proper force feedback (transparency) the overall system must be stable. Stability and transparency together define key characteristics of system performance. The control design stage must ensure improved performance of the teleoperator. This is usually not easy to achieve as ensuring enhanced transparency comes at the cost of degraded stability and vice versa; the two are conflicting goals. A perfectly stable teleoperation system may still not be ideal due to compromised transparency. Some technical factors that may adversely affect the system performance include:

- Presence of the human operator in the closed-loop system
- Kinematics and dynamics of the master and slave manipulators
- Computational and communication time delays and other communication constraints such as noise and packet loss/drop-out
- Strict loop-rate requirement of the controllers
- Unavailability of adequate sensor types due to cost constraints
- Strict safety regulations such as those in homecare and medical robotics

Particularly because of these challenges haptic teleoperation has been an active area of research.

1.3 Goals of the Research

The main objectives of this thesis are as follows:

- To develop a bilateral haptic teleoperation scheme for a homecare environment that is safe and reliable.
- To design an advanced interaction controller such as impedance control that can achieve a desired level of compliant behavior in the manipulator.
- To analyze performance of the haptic system with the developed teleoperation scheme subject to uncertainties in the robot dynamic model and other undesirable effects.

- To evaluate and validate the performance of the developed system through computer simulation and laboratory experimentation with a prototype haptic teleoperation system.

1.4 Related Work

1.4.1 Real-time Online Model Learning in Robotics

Model-based control such as computed-torque control and impedance control requires an accurate dynamic model of the robot that is amenable to computation within a fast control loop (usually running at 500Hz to 1000Hz). As an alternative to deriving equations of motion for the robot by traditional approaches such as the Lagrangian and Newton-Euler formulations, learning of these equations using sensory data has recently drawn the attention of researchers. The Lagrangian formulation is not only complex to derive but calculating the kinetic energy of a robot is computationally expensive and therefore cannot be used in real-time control. The Newton-Euler algorithm, due to its recursive nature, has facilitated computation of forward and inverse dynamics in real-time. Both these analytical models are parametric in nature and rely on the inertial parameters (such as link mass, center of mass, moments of inertia, and friction parameters) provided by the manufacturer, which are not usually accurate and may change over time due to wear and tear. In some situations they may be completely unavailable for an added extra part (such as a tool, sensor, or gripper) or even for the entire robot. Techniques based on machine learning, on the other hand, are non-parametric and data-driven and also fast, adaptive and efficient (if properly trained). Moreover, machine learning techniques can be used not only for learning the inverse dynamic model, but also to learn the forward kinematic model [6], inverse kinematic model [7] and task-dependent impedance models [8][9] for the robot. Another reason which have made them popular in the last three decades is the advancement and cost-effectiveness of computer memory and processing hardware.

Machine learning techniques can be categorized as supervised or unsupervised. In supervised learning, the model is trained on both the input and the correct desired outputs (targets). The model then performs *generalization* in which it has to produce, based on an inferred approximated function, reasonable outputs for the input data that is unseen and not encountered during the training. In robotics, supervised learning is mostly used for learning the inverse kinematic/dynamics model, which is then used in computed-torque control, impedance control, model predictive control, or operational space control of the robot.

In learning kinematic or dynamic model of a robot, the function that is to be approximated is highly nonlinear and the input data is high-dimensional. The function approximation methods used and found in the literature can be classified as either global or local learning methods. In global methods, distributed weight representations are used by the learning algorithm, which attempts to fit the input-output data with a single global function. Every time a new input data are incorporated into model training and development, the weights are updated globally. In local methods, weighting functions (usually Gaussian) are local and therefore update of only a small subset of weights is required. Here, the given nonlinear function is fitted with simple local functions. Local methods are well-suited for real-time control in robotics. With the availability of modern software packages, which use incremental learning where local functions and weights are updated according to the incremental addition of new input data over time. Unlike traditional machine learning, the learning process does not assume the availability of sufficient training data. They also require less computer memory as all the raw data is not saved for training. This is the reason why machine learning has received increasingly more attention of researchers in modern robotic control these days.

Global learning frameworks include Gaussian Process Regression (GPR) [10] and Support Vector Machines (SVM) [11]. LWPR [12] and local GPR [13] are examples of local learning systems. ANN has been used as both global and local function approximation method. In [14], [15] and [16], the authors have presented a survey of model learning for robot control.

Salaun et al. in [17] used incremental model learning method to learn the forward model of a redundant robot. The intended use was to achieve motion tracking by the manipulator in task-space which had been overlooked by the researchers earlier. Unlike model learning in joint-space, learning in task-space is harder due to the fact that the mapping between two spaces (joint- and task-spaces) is ill-posed, especially for redundant robots in which infinite number of inverse mappings exist. The LWPR algorithm was used for model learning. The proposed method was tested on a simulated 3-DOF planar robot.

Nguyen-Tuong and Peters 2011 [14] have categorized the ways a learned model is used by a learning control framework into three categories: *direct modeling, indirect modeling,* and *distal teacher approach*. Each category uses a different approach to learn and update the model while they are in use. The distal learning approach is particularly useful for ill-posed mapping problem where the input-output relationship is ill-posed. In this approach, a direct dynamic model is used to guide learning of the inverse model. The direct dynamic model may be computed by using the known or partially known dynamic parameters provided by the manufacturer.

Sun de la Cruz et al. in [18], [19], have used LWPR to learn the manipulator inverse dynamics. The authors have proposed a learning algorithm that can incorporate prior knowledge about the model being learned. An approximation of the prior information is used to initialize the local linear models of the LWPR framework. The proposed approach was evaluated on a simulated 6-DOF PUMA 560 robot. The same authors, in [20], have incorporated a-priori knowledge of the system model in the learning algorithm. More specifically, the authors have used the first order approximation of the available dynamic model to initialize the learning algorithm LWPR (explained next). The proposed approach allows the system to generalize outside of the regions of training and also operate even if not enough measurements are available for initial training. The validity of the proposed approach has been shown experimentally on a 6-DOF arm where the tracking performance has significantly been approved.

Nguyen-Tuong and Peters in [21] proposed a model parameterization approach for the local model to achieve operational space control of redundant robots based on the online learning model. The proposed approach was evaluated on 3-DOF and 7-DOF versions of a simulated WAM arm. The same authors along with other colleagues in [22] have used GPR, SVR, and LWPR methods to learn inverse dynamics of both simulated and real versions of the SARCOS arm to compare performance of the three algorithms. It was found that the LWPR algorithm outperformed the other two in terms of online execution time, is relatively faster, and therefore best for time critical control of the robot. On the other hand GPR and SVR were found to be more accurate.

The latest work in learning inverse dynamics came from the authors in [9] where the learning process also incorporates contact with the environment. The focus is on the whole-body control where the contacts may occur anywhere along the robot body. The authors have proposed a data-driven mixture-of-experts learning approach using Gaussian processes. The learned model predicts joint torques generated through contact with the arm body from the raw data obtained with tactile and force/torque sensors. Evaluation of the proposed method is done on the *iCub* humanoid robot, a popular open source cognitive robotic platform.

1.4.2 Interaction Control

In many robotic applications in the industry and healthcare the robot end-effector comes in contact with the environment. There is an exchange of forces (action/reaction) between the robot and the environment. During interaction, control of both motion and force (directly or indirectly) becomes crucial. Interaction control algorithms can be classified depending on whether the control law uses the relation between position and force (stiffness control), between velocity and force (impedance/admittance control), or is directly based on explicit position and/or force feedback [23].

Volpe and Khosla in [24] experimentally evaluated both force and impedance control strategies. The paper focused on the importance of considering the implementation details that are required for the force and impedance control of robotic manipulators. Ueberle and Buss in [25] analyzed and evaluated using a set of hardware experiments the performance of haptic control architectures including the standard impedance and admittance control schemes. In [59] the authors have experimentally evaluated position-, force-, and impedance control for their applicability in minimally invasive surgical applications. Stanzyk et al. in [26] has developed an experimental teleoperation system with an admittance controller on the master side and a position-based impedance controller on the slave side. They showed the effectiveness of the implemented control schemes in a series of experiments such as tracking of free space in motion, haptic exploration of different materials, and driving a screw with an aluminum tool. The choice of appropriate control scheme (admittance control on master side and impedance control on the slave side) helps them successfully achieve the requirements such as accurate position/force tracking, singularity-robust kinematic transformations, and stable environment interaction.

Recently, Kikuuwe in 2014 [27] has enhanced the inner motion tracking loop in admittance control by proposing a new position controller which behaves as a proportional-integral-derivative (PID) controller with acceleration feedforward in normal situations and as a sliding mode controller when the actuator force is saturated. The proposed controller, which is validated through experiments on a 6-DOF industrial manipulator, achieves better stability and smoothness and also quickly responds to changes in the applied force.

1.4.3 Impedance Control for Haptic Teleoperation

Bilateral teleoperation allows a human operator to interact with a remote environment using the enhanced skills of actuation and sensing of a robot and the superior cognitive skills of a human. It has shown promising results in applications such as telemedicine, telesurgery, micro- and macro-telemanipulation, and remote access to hazardous and difficult environments. In most such applications, the human operator has to rely only on visual feedback for transparency in carrying out the task properly. It is believed that along with visual feedback, the introduction of force (haptic) feedback in the teleoperation system can result in significantly improved performance.

In a bilateral teleoperation system, a human operator with the help of a master manipulator (a haptic interface) interacts remotely with an environment (in the present application, a human in a homecare environment) through a slave manipulator (teleoperator). The interaction force (sensor information) between the slave manipulator and the environment is reflected back to the human operator giving him the feeling of interaction in a transparent manner, resulting in improved performance [28].

The choice of the control scheme for teleoperation depends on such considerations as the nature of the application and the task that is being carried out. Different control techniques have been considered by researchers and are found in the literature.

References [29] and [30] provide a comprehensive survey of the literature on bilateral teleoperation. In a position-position control scheme for teleoperation, a bilateral proportional-derivative (PD) controller connects the master and slave manipulators via a virtual spring and damper between the two end effectors. The work reported in [31] uses a local contact force controller to control the slave manipulator during contact. The scheme is a modification of the well-known virtual coupling scheme and uses a virtual spring to connect the end effectors of the master and slave robots. The desired contact force, which the slave controller is expected to track, is computed at the master side and is proportional to the error signal, which is the relative position difference between the master and the slave manipulators. A force representing this error is fed back to the master and felt at the haptic interface of the human operator. This approach is claimed to have enhanced stability because the contact force measurement is not used in the control loop to avoid the effect of associated noise. However, this enhancement comes at the expense of degraded transparency because the impedance felt by the human operator is now a function of the virtual (coupling) impedance and the environmental impedance.

An alternative scheme of position-force control enhances the haptic feedback by explicitly measuring the contact force between the slave and the environment and sending it to the master manipulator, to be presented to the human operator. If the contact instability is properly dealt with, it will provide enhanced transparency by hiding unwanted dynamics of the slave manipulator from the user and therefore resulting in more realistic haptic feedback.

In [32], it is shown that the direct display of measured force to the human operator is of limited benefit. Consequently a four-channel architecture may be implemented where information on both motion and interaction force is exchanged between the master and the slave manipulators [33]. The objective is to match the impedance felt by the human operator with that of the remote environment. Therefore, instead of directly feeding the environmental force to the human operator, impedance control of master and slave manipulators may be used to ensure reflection of the desired impedance characteristics to the operator at the master manipulator.

The concept of impedance control in robotics was first introduced in [34]. Instead of directly controlling the robot's motion, the mechanical impedance (velocity/force ratio) was controlled such that a desired impedance model is enforced. In [35], impedance control is used in bilateral teleoperation. They have used sliding mode control on the slave side to enforce the desired impedance and achieve perfect motion tracking in the presence of time delay in the communication channel. Recently, in [36] the authors have used impedance control in their proposed four-channel force-reflecting teleoperation algorithm. In [37], a teleoperation architecture has been proposed and is termed position-based admittance control with force–force exchange. There the focus is on the robust stability analysis and the effect of variable human movement on task performance and the feeling of Telepresence.

1.4.4 Time Delayed Teleoperation

Several methods exist in the literature that address the problem of instability due to the time delay in a teleoperation system. Often the goals are to: (1) stabilize the teleoperation

operation system under constant or variable time delay, (2) achieve or improve transparency of the teleoperation by faithfully displaying the interaction force and impedance to the human operator at the master side, and (3) achieve bilateral motion synchronization between the master and slave ends. The two popular methods that have been used to achieve these goals are: (1) scattering or wave variables, and (2) Time Domain Passivity Control (TDPC). Both are based on the passivity theory.

Pioneering work that deals with bilateral teleoperation that includes time delay came from Anderson and Spong [38] and Niemeyer and Slotine [39]. They used passivity theory, which deals with the exchange of energy between interconnected systems. In order to make use of passivity theory for bilateral teleoperation with constant time delay, the teleoperation system, as shown in Figure 1.2, is treated as an interconnection of 1- or 2-port networks where energy at each port is exchanged in the form of an effort-flow pair of variables. For a mechanical system the effort-flow pair corresponds to the force-velocity pair, which for a robot manipulator may also be interpreted as the torque-angular velocity pair at each revolute joint. These effort ($f(t)$) and flow ($\dot{x}(t)$) variables at each port are related by a scattering operator S as follows:

$$\begin{aligned} u_m(t) &= (f_m^d(t) + b\dot{x}_m(t))/\sqrt{2b}, v_m(t) = (-f_m^d(t) + b\dot{x}_m(t))/\sqrt{2b} \\ v_s(t) &= (-f_e(t) + b\dot{x}_s^d(t))/\sqrt{2b}, u_s(t) = (f_e(t) + b\dot{x}_s^d(t))/\sqrt{2b} \end{aligned} \tag{1.1}$$

$$\begin{bmatrix} v_m & u_s \end{bmatrix} = S\begin{bmatrix} v_s & u_m \end{bmatrix}^T \tag{1.2}$$

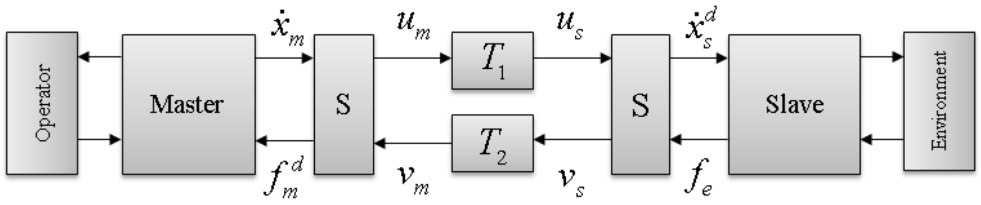

Figure 1.2 Scattering transformation in a bilateral teleoperation setup.

In order for an n-port network in the teleoperation system to be passive, the required bound condition on the norm of the scattering operator is given by

$$\|S(j\omega)\|_\infty = \bar{\sigma}(S^T(j\omega)S(j\omega)) \leq 1 \tag{1.3}$$

Therefore, passivity becomes a condition on the system gain, which is unaffected by the time delay. If a human operator and the environment are represented by a 1-port network and the master/slave manipulators and the communication channel are represented by a 2-port network, then the aforementioned scattering formulation can be used to guarantee stability of the closed-loop teleoperation system by developing appropriate control laws for the master and slave manipulators. Chopra and Spong [40] and Lee and Spong [41] has extended and improved over this previous work. In particular, Chopra and Spong discuss time-varying delays and position synchronization. Passivity-based approaches assume perfect transmission of data between the master and the slave sides with no loss of information and errors in the communication channel. The stability conditions are also overly conservative in some cases. Passive controllers cannot hide the dynamics of the slave robot from the human operator and therefore transparency will be compromised.

Scattering or wave-variables-based approach has been actively investigated and extended as can be found in the recent work in [42], [43], and [44]. In [42], the author has extended the wave-variables scheme to the case where the slave manipulator transitions between two different environments. The work addresses the problem of tuning the wave impedance which is an important design parameter in the wave-based approach. They suggest that instead of having a constant wave impedance, its value should be varied smoothly between a certain interval. This will help adjust and improve the level of transparency of the teleoperation system. The wave impedance is kept small when the slave robot is in free motion, and increased when the slave robot is in contact with a hard environment. It was observed that faster variations in the value of the wave impedance resulted in adding unwanted oscillations to the transient response.

In [43], the authors have proposed a new four-channel (4-CH) architecture to deal with the negative effect of wave-based reflections. They propose two modifications to the wave-based scheme to produce improved transparency and stability under large time delays. In their modifications, the outgoing wave variables on the master and slave ends are computed such that they no longer retain information from the incoming wave variable. Experimental results demonstrate the effectiveness of the proposed 4-CH system.

In [44], the authors have investigated the effects of communication delays on the backdrivability of a teleoperation system. The wave-variable-based control techniques have been used. They suggest that position compensation, force compensation, or decoupling of the slave velocity can achieve better backdrivability in wave-based controllers and result in reduced reflections for unknown environments.

The other popular approach for dealing with time delay in teleoperation is the TDPC which was originally presented in [45][46]. Currently, this is an active research topic with more recent articles appearing as in [47], [48], and [49]. Rebelo et al. in [48] has extended the TDPC approach to a 4-CH bilateral controller. This new architecture provides perfect transparency unlike the previous position-force and position-position architectures. In [49], the authors address the problem of position synchronization in the TPDC approach.

1.5 Contributions and Organization of the Thesis

Unlike traditional teleoperation, haptic teleoperation in the context of homecare involves some challenges that need to be carefully addressed. The master and slave manipulators must behave in as much compliant manner as possible. This should not be achieved at the expense of loosing motion accuracy. In this thesis the problem of interaction control is addressed. Following are the key contributions made in the thesis:

1. The inverse dynamic model of a four-degree-of-freedom (4-DOF) commercial robot manipulator is identified by a complete data-driven learning-based technique, which does not assume any a-priori knowledge of the dynamic parameters. For comparing the performance, the dynamic model is also analytically derived and simulated. The developed model can be used and updated online within the real-time control loop of the manipulator and therefore solves a major problem in the implementation of impedance control in haptic teleoperation.

2. Three interaction control techniques: stiffness control, standard impedance control, and admittance control, are designed and experimentally evaluated for their performance and applicability in homecare robotics. Shortcomings of stiffness control and impedance control are highlighted with the help of experimentation and are then improved by admittance control. Criteria for selecting the desired impedance/admittance model are proposed.

3. Position synchronization and transparency of the popular wave-variables based approach for time-delay compensation in haptic teleoperation is improved. The results of the

proposed approach are presented for an experimental 1-DOF test bed which demonstrate the effectiveness of the proposed improvement.

4. A complete experimental testbed for haptic teleoperation with special focus on safety and reliability is developed in the laboratory.

This thesis is divided into the following five chapters:

- **Chapter 1** presents the motivation behind the thesis and a review of the relevant literature on the subject. This provides a context where this work fits in and also identifies the gaps in the literature that need to be closed.
- In **Chapter 2** a detailed dynamic model of the robot is derived and experimentally identified. A new learning-based technique is used, which has advantages over traditional approaches.
- **Chapter 3** presents interaction control algorithms based on the inverse dynamic model. It is found experimentally that the admittance control outperforms the rest of the techniques.
- **Chapter 4** develops and implements a haptic teleoperation architecture with time delay. The time delay in the communication channel, which poses a significant threat to the stability and the transparency of the teleoperation system, is addressed. The popular wave-variable-based approach is used and ways are presented in which position tracking and transparency can be improved further.
- **Chapter 5** provides a summary of the overall thesis. Possible directions for future work are presented as well.

Chapter 2: Derivation and Identification of Kinematic and Dynamic Models

2.1 Introduction

This chapter first addresses the development of an experimental testbed for the present research. Next, kinematic and dynamic model are analytically derived. Some simulation studies are carried out. To come up with a robust and realistic inverse dynamic model of the robot, a data-driven learning-based technique is used. Procedures for identification of the model parameters are presented.

2.2 Development of Experimental Setup

This section provides an overview of the hardware and software systems that are used for experimental validation of the algorithms developed in this work.

Barrett WAM Arm

The Whole Arm Manipulator (WAM) developed by Barrett Technology, shown in Figure **2.1**, is a backdrivable (with backdrivability greater than 95%) serial-link manipulator with human-like kinematics. Each axis of the arm is driven by a brushless motor that has a digital motor controller, called *Puck*, directly mounted on it. Puck also has sensors for position, temperature, and current, and therefore plays a crucial role in monitoring the system health for safety purposes. The arm can be controlled by an internal or external computer connected through its CAN bus interface. The latter provides more memory and processing power which is required for advanced control algorithms as developed in this work. An external PC is therefore configured. Communication between the Pucks and control PC is achieved through a built-in CAN bus at the rate of 1Mbps. The control loop, which runs at 500Hz (but scalable up to

1000Hz) involves reading joint positions from Pucks, calculating new torque signal by the control algorithm based on the sensed information and presenting the result to the Puck which then applies it to the actuator. WAM is fundamentally a torque-controlled robot, but allows for direct position control (independent joint proportional-integral-derivative or PID control) too.

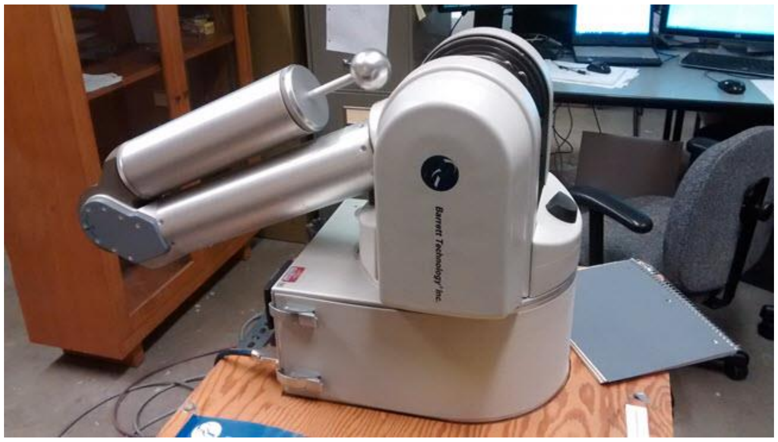

Figure 2.1 WAM arm by Barrett Technology.

Application development for the WAM arm is done in the open-source C++ library, called libbarrett [50], which is provided and supported by the manufacturer. The library communicates with the arm and lets the developer adjust the control loop sample time, log data in real-time, which can be used later for plotting and analysis purposes (in MATLAB/SIMULINK), and provides the control loop callbacks that can allow calculations based on WAM data in real-time [51]. In order to make the real-time control of the Barrett products (WAM arm) possible, the libbarrett library must be compiled on a customized PC with real-time Operating System. While Barrett provides an internal PC as part of the WAM arm, a customized PC with advanced hardware and software configuration has been built at the IAL, the details of which are presented in Table 2.1.

Table 2.1 Specifications of external WAM PC.

Motherboard	ASUSTek P5GD1
Processor	3.2 GHz Intel Pentium 4
Memory	2.0 GB
Linux distribution	Ubuntu 10.04 (lucid)
Linux kernel	2.6.32.59
Realtime development framework	Xenomai 2.5.5.2
Ethernet	Gigabit
CANbus	Peak-System PCAN-PCI Dual Channel
libbarrett API	1.2.1

PHANToM Haptic Interface

The PHANToM haptic interface (shown in Figure 2.2) is a popular commercially available haptic system. The interface was originally developed by Sensable Technologies and is currently supported by 3D Systems Geomagic Solutions, SC, USA. The device, which actually is a 6DOF serial manipulator, is manipulated by holding a haptic stylus (also called *gimbal*). The stylus is actuated by three small motors in case of force only feedback and six motors in case of both force and torque feedback. Apart from its use in haptic teleoperation for application in homecare, hazardous material handling, mining, and so on the device has also been used in medical and flight simulators, video-gaming, mobile phones, and robot-assisted surgery.

Haptic devices in general and PHANToM haptic device in particular, are designed as low-impedance hardware. Like other industrial robots, they are a kinematic chain of rigid links, but with insignificant dynamics which are often neglected in controller design. Friction is low due to the cable-driven mechanism. Gravity is compensated for by using the counterweights that are physically mounted on the mechanism. This is the reason why the device is known in haptic research community for open-loop impedance control.

Application development for the PHANToM device is done in the open-source C++ library, called CHAI 3D [52], which is a multiplatform haptic development framework for computer haptics, visualization, and interactive real-time simulation. The library supports many of the commercially available haptic devices. Therefore an application developed on one haptic device can be run on different hardware without any changes. The complexities of underlying device hardware are abstracted away from the application developer. CHAI 3D runs on PHANToM PC (see specifications in Table 2.2) that is used as a controller of the PHANToM device. Although application development is done in Microsoft Windows which is a non-real-time operating system, CHAI 3D strives to achieve a strict haptic rendering update rate of 1kHz.

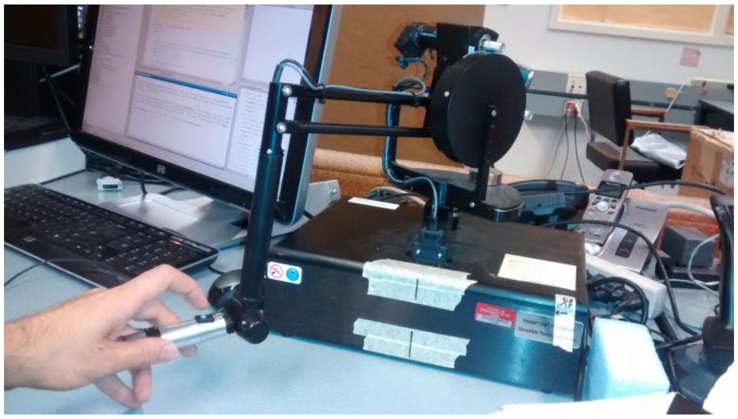

Figure 2.2 PHANToM haptic interface by Sensable Technologies.

Table 2.2 Specifications of PHANToM PC

Motherboard	ASUSTek P5GD1
Processor	3.2 GHz Intel Pentium 4
Memory	2.0 GB
Operating system	Windows
Ethernet	Gigabit
CHAI3D API	3.0.0

Complete Experimental Testbed

A complete experimental setup is shown in Figure 2.3, which highlights the hardware/software systems and control loop rates. The communication between the master and slave ends is carried out in the form of User Datagram Protocol (UDP) packets, which is an ideal protocol for real-time communication due to low traffic-overhead involved as compared to other protocols in the Internet protocol suite.

Figure 2.3 also highlights the control loop bandwidths. The high control loop rate on PHANToM PC (master side) is guaranteed by CHAI 3D only for the high priority thread which does the haptic force computation. Due to the non-real-time nature of the underlying system, variation in update rate is observed to be in the range 900Hz to 1kHz, which is acceptable. The haptic rendering rate puts a limit on the bandwidth of force displayed to the human operator. With the given 1kHz limit on the sampling rate, the CHAI 3D haptic loop can reconstruct and display forces having bandwidth less than or equal to 500Hz (following the Shannon sampling theorem). In the context of teleoperation, it means that an interaction force, simulated or received from the actual remote environment, can only be displayed with a reasonable amount of stability and smoothness if it lies below the aforementioned limit (500Hz). Extremely stiff objects, therefore, may cause instability of the haptic rendering algorithm. The slave side runs even slower (at a default rate of 500Hz). The reason is that the intensive computations such as computing the inverse dynamic model cannot be achieved at a rate higher than 500Hz. The effective bandwidth of the system is therefore 500Hz although some parts such as haptic system and force/torque sensor may have higher bandwidths. This works well in the context of haptic teleoperation in a home environment.

Kinematic and dynamic specifications of PHANToM and WAM as provided by the manufacturers are given in Appendix A and Appendix B, respectively.

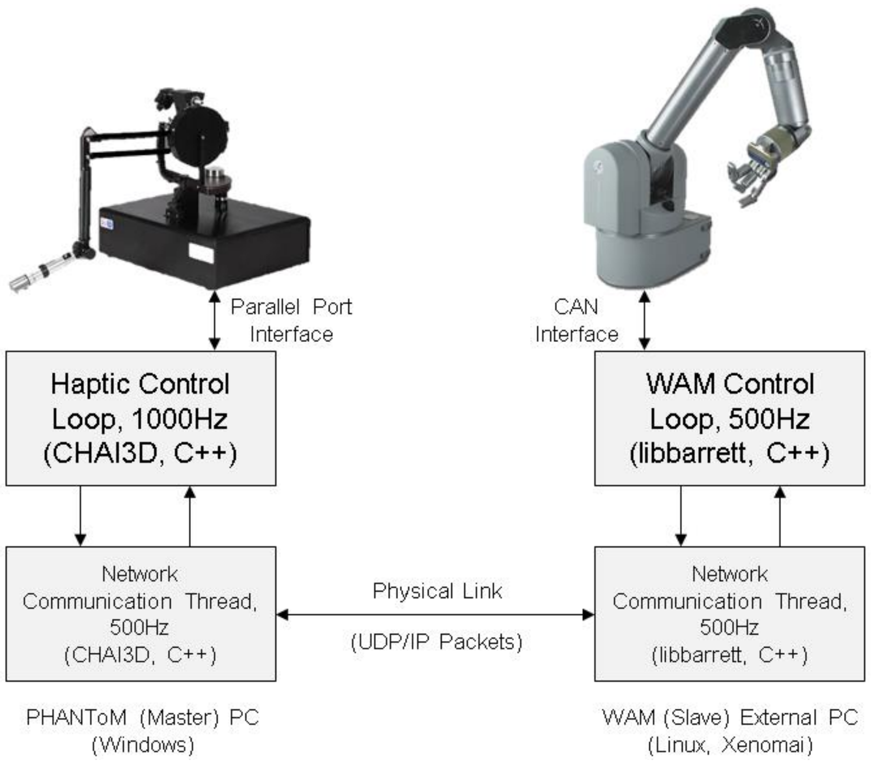
Parallel Port Interface
CAN Interface
Haptic Control Loop, 1000Hz (CHAI3D, C++)
WAM Control Loop, 500Hz (libbarrett, C++)
Network Communication Thread, 500Hz (CHAI3D, C++)
Physical Link
(UDP/IP Packets)
Network Communication Thread, 500Hz (libbarrett, C++)
PHANToM (Master) PC (Windows)
WAM (Slave) External PC (Linux, Xenomai)

Figure 2.3 Experimental teleoperation testbed.

2.3 Derivation of Kinematic Models

2.3.1 Kinematic Modeling

In forward kinematics a set of kinematic equations is obtained, which can be used to compute the position and the orientation of the end-effector for a given set of values of joint variables. The kinematic equations are obtained using homogeneous transformation matrices, which characterize the relative movements of the coordinate frames assigned to each link. As a first step towards determining the kinematic equations, coordinate frames are assigned to each link using the Denavit-Hartenberg (D-H) convention. According to this convention, the frames are assigned such that the axis x_i (i.e. the *x*-axis of the i^{th} frame) is perpendicular to, and also intersects, the z_{i-1} axis (i.e. *z*-axis of the previous frame). Once the coordinate frames are assigned following the D-H convention, a set of four D-H parameters is obtained for each link, which are defined as follows:

- Link length a_i: Distance between z_{i-1} and z_i. It is measured along x_i and is always constant.
- Link twist α_i: Angle between z_{i-1} and z_i. It is measured around x_i and is always constant.
- Link offset d_i: Distance between o_{i-1} and intersection of z_{i-1} and x_i. It is measured along z_{i-1} is variable if the joint is prismatic.
- Joint angle θ_i: Angle between x_{i-1} and x_i. It is measured around z_{i-1} and is variable if the joint is revolute.

These parameters are summarized in a table, called *D-H Table*, where row i consists of the D-H parameters of link i. The number of rows *n* corresponds to the number of DOF of the manipulator (i.e. $i = 1, 2, \ldots, n$). Corresponding to each row of the D-H table, is a unique homogeneous transformation matrix A_i computed as ([53]),

$$A_i = \mathrm{Rot}_{z,\theta_i}\mathrm{Trans}_{z,d_i}\mathrm{Trans}_{x,a_i}\mathrm{Rot}_{x,\alpha_i}$$
$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$
$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where

- $\mathrm{Rot}_{z,\theta_i}$: counterclockwise rotation by θ_i about the z_i -axis.
- Trans_{z,d_i} : translation by d_i along the z_i -axis.
- Trans_{x,a_i} : translation by a_i along the x_i -axis.
- $\mathrm{Rot}_{x,\alpha_i}$: counterclockwise rotation by α_i about the x_i -axis.

The transformation matrix A_i represents the position and orientation of frame $o_i x_i y_i z_i$ with respect to the previous frame (i.e. $o_{i-1}x_{i-1}y_{i-1}z_{i-1}$). The transformation matrix T_n^0 that relates the end-effector frame $o_n x_n y_n z_n$ to the base frame $o_0 x_0 y_0 z_0$ is given as,

$$T_n^0 = A_1(q_1)A_2(q_2)\cdots A_n(q_n) = \begin{bmatrix} R_n^0(q) & o_n^0(q) \\ 0 & 1 \end{bmatrix} \tag{2.2}$$

2.3.2 Differential Kinematics

The forward or direct kinematics model that was developed in the previous section is used to convert the specified values of joint variables into the end-effector position and orientation. The differential kinematic model relates incremental joint movements to incremental end-effector movements (both linear and angular). Hence it provides a relationship between the joint velocities and the end-effector linear and angular velocities. Essential to this modeling is finding the *Jacobian matrix*, $J \in \mathbb{R}^{6\times n}$, which gives the instantaneous mapping between a vector of joint velocities $\dot{q} \in \mathbb{R}^n$ and the end-effector linear and angular velocities $\dot{x}(t) = [v_x(t), v_y(t), v_z(t), \omega_x(t), \omega_y(t), \omega_z(t)]^T \in \mathbb{R}^6$.

$$\dot{x}(t) = J(q)\dot{q}(t) \tag{2.3}$$

An important use of the Jacobian is to find *singular configurations* of the robot. Singularities play an important role in the design and control of robot manipulators. At singular configuration or the singularity, the null space of the Jacobian, $N(J^T)$, has a non-zero dimension and the external forces have to be borne by the manipulator structure of manipulator. Moreover, bounded end-effector velocities (force and torques) may result in unbounded joint velocities (joint torques).

Dynamic modeling and control of manipulators also make extensive use of the Jacobian matrix which will be explained later (in Section 2.4).

In order to compute the manipulator Jacobian, it is partitioned into two sub-parts, $J_v(q)$ and $J_\omega(q)$, as follows,

$$J(q) = \begin{bmatrix} J_v(q) \\ J_\omega(q) \end{bmatrix} \tag{2.4}$$

Here $J_v(q)$ corresponds to the linear velocities of the end-effector and relates them to joint velocities $\dot{q}$ while $J_\omega(q)$ corresponds to angular velocities of end-effector about an arbitrary direction. The number of columns in the Jacobian matrix corresponds to the number of DOFs of the manipulator. Column i of $J_v(q)$ matrix is computed as,

$$J_{vi}(q) = \frac{\partial o_n^0(q)}{\partial q_i} \tag{2.5}$$

where $o_n^0(q)$ is the origin of the end-effector frame (and it moves as the configuration of the manipulator changes) and is extracted from Equation 2.2. Column i of $J_\omega(q)$ matrix is computed as ([53]),

$$J_{\omega i}(q) = \rho_i z_{i-1} = \rho_i R_{i-1}^0 \hat{k} \tag{2.6}$$

where $\rho_i = 1$ if joint i is revolute, and is 0 if it is prismatic. Matrix R_{i-1}^0 represents the rotation of frame $i-1$ relative to the base frame. Joint i rotates about a unit vector $\hat{k}$ about its z-axis (by the DH convention).

A special case of the manipulator Jacobian is the *link Jacobian* $J_{ci}(q)$ ($i = 1,\ldots,n$) which relates the linear and angular velocities of the center of mass c_i of the link to the angular velocities of all the joints that contribute to it.

$$\begin{bmatrix} v_{ci} \\ \omega_{ci} \end{bmatrix} = J_{ci}(q)\dot{q} \tag{2.7}$$

If c_i represent the center of mass of link i relative to base coordinates, then

$$J_i(q) = \begin{bmatrix} \frac{\partial c_i}{\partial q_1} & \frac{\partial c_i}{\partial q_2} & \cdots & \frac{\partial c_i}{\partial q_i} & 0 & \cdots & 0 \\ \rho_1 z_0 & \rho_2 z & \cdots & \rho_i z_{i-1} & 0 & \cdots & 0 \end{bmatrix} = \begin{bmatrix} J_{vi}(q) \\ \\ J_{\omega i}(q) \end{bmatrix} \tag{2.8}$$

Inverse Differential Kinematics

Control of robotic manipulators in Cartesian space involves inverse of the Jacobian matrix, J^{-1}. For this purpose, the inverse solution of Equation 2.3 is obtained which only exists if J is a square matrix ($n \times n$). In the case of under-actuated robots ($n < m$) (m being the number of Cartesian DOFs) or redundant robots ($m < n$), $J^{-1}_{m \times n}(q)$ does not exist. In these cases *pseudoinverse* or generalized inverse of the Jacobian, $J^{+}(q)$, is computed which always exists and is unique. For under-actuated robots having $n < m$ it is can be computed as [53]:

$$J^{+}_{m \times n} = (J^T J)^{-1} J^T \tag{2.9}$$

while, for kinematically redundant robots ($m < n,\ J \in R^{m \times n}$), it is computed as:

$$J^{+}_{n \times m} = J^T (JJ^T)^{-1} \tag{2.10}$$

where $(JJ^T)^{-1}$ exists if $rank(J) = m$; i.e. J is full (row) rank. If the full (row) rank requirement is not met, the pseudoinverse J^{+} is computed numerically using the *singular value decomposition* (SVD) as:

$$J^{+} = V \Sigma^{-1} U^T \tag{2.11}$$

where Σ is a diagonal matrix of the singular values σ_i of J i.e., $J = U \Sigma V^T = \sum_{i=1}^{n} u_i \sigma_i v_i^T$. The commanded joint-space trajectory $q(t)$ can now be calculated from the reference Cartesian trajectory $\dot{x}$ as:

$$\dot{q} = J^{+}\dot{x} + (I - J^{+}J)b \tag{2.12}$$

where $I \in R^{n \times n}$ is the identity matrix and $b \in R^{n}$ is an arbitrary vector. Equation 2.12 is a nonlinear differential equation whose solution can be obtained using least square techniques which aim to minimize $\| \dot{q}(t) \|^2$ subject to the constraint $\dot{x} = J(q)\dot{q}$ [54]. Equation 2.12 along with the initial condition $q(0) = f^{-1}x(0)$ (where f^{-1} is the inverse kinematics function) is commonly referred to as *resolved-motion rate control* [54] and is depicted in Figure 2.4.

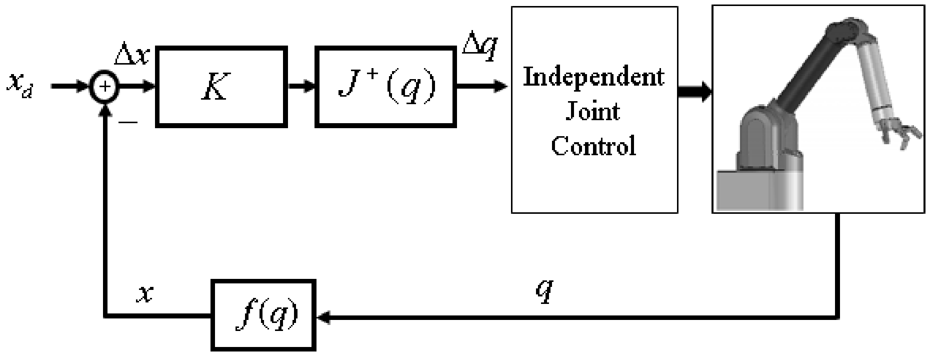

Figure 2.4 Resolved-motion rate control (based on the Jacobian-inverse).

The disadvantage of Cartesian control using the Jacobian-inverse method (Figure 2.4) is its non-robustness to joint-space singularities. Although in case of redundant robotic arms, the extra joints can be used to avoid singularities, the Jacobian-inverse-based rate control is still not suited for real-time robotic applications such as homecare. It is due to the fact that their numerical implementation is computationally intensive and therefore online and real-time computation (within control loops of speed of the order 500Hz) is impractical. Conventional industrial robots, when controlled in the Cartesian space to carry out tasks such as spray-painting, drilling, cutting, and welding have to compute the Jacobian-inverse off-line on powerful processors [55].

In order to overcome the above shortcoming of Jacobian-inverse-based motion control, modern robotic arms and haptic devices use the *Jacobian-transpose*-based control which is shown in Figure 2.5. Robotic arms such as WAM arm and PHANToM haptic interface that are highly *backdriveable*, having inherent backdrivability $> 95\%$, make use of the Jacobian-transpose for Cartesian control of forces (force control), motion trajectories (position control) and impedance control where the force-motion relationship of the arm is controlled.

The Jacobian-transpose, $J^T(q)$, also plays an important role in torque-control of robotic arms (presented in Chapter 3) and also in the transformation of external forces to joint torques.

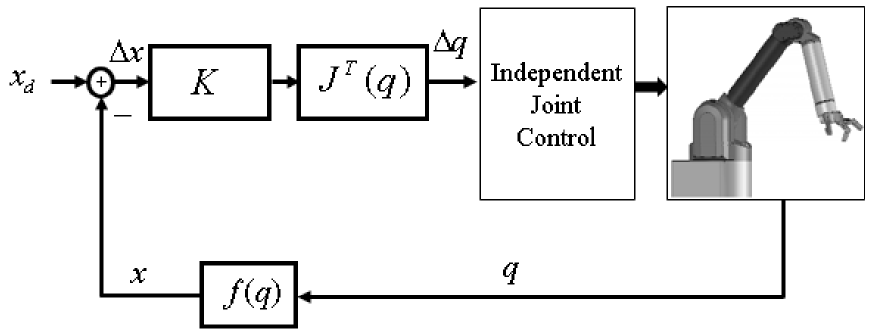

Figure 2.5 Resolved-motion rate control (based on the Jacobian-transpose).

2.3.3 Force/Torque Relationship

As noted before, the manipulator's Jacobian matrix can also be used to relate small changes in angular displacement, Δq, at the joints to small changes in position and orientation of the end-effector, ΔX, as,

$$\Delta X = J(q)\Delta q \tag{2.13}$$

where $X = [x, y, z, \alpha, \beta, \gamma]^{\mathrm{T}} \in \mathbb{R}^6$ is a vector comprising of ΔP, which are small changes in end-effector's $x, y,$ and z positions, and $\Delta\phi$, which are small changes in end-effector's orientation about $x, y,$ and z axes. If an external force $F_{ext} = [f_x, f_y, f_z, n_x, n_y, n_z]^T$ (a vector of static or dynamic forces and moments) is applied at the end of the arm, which causes an infinitesimal displacement ΔX then the net work done, ΔW, at the end point will be,

$$\Delta W = F_{ext} \cdot \Delta X = F_{ext}^{T} \Delta X \tag{2.14}$$

According to the *principle of virtual work* [54], there will be a corresponding net work done by the joints,

$$\Delta W = \tau \cdot \Delta q = \tau^{T} \Delta q \tag{2.15}$$

From Equations 2.13, 2.14, and 2.15, the following relationship between external forces (and moments) and joint torques can be derived,

$$\tau = J^{T} F_{ext} \tag{2.16}$$

If $K_{\theta} = \text{diag}\{k_{\theta 1}, k_{\theta 2}, \ldots, k_{\theta n}\}$ is a matrix of joints stiffness, then under static conditions the application of external force will be corrected by a restoring torque at the joints as,

$$\tau = K_{\theta} \Delta q \tag{2.17}$$

Using equations 2.13, 2.16, and 2.17, the following expression can be derived,

$$\Delta X = G_{x}(q) F_{ext} \tag{2.18}$$

where $G_{x}(q) = J(q) K_{\theta}^{-1} J^{T}(q)$ is the end-of-arm *compliance* and it measures the sensitivity of the arm to deflections. For a constant external force, increased compliance would mean more deflection of the end effector of the manipulator. The reciprocal of compliance is the end-of-arm *stiffness*, $K_{x}(q)$, and is given as,

$$K_{x}(q) = J^{-T}\left(q\right) K_{\theta} J^{-1}\left(q\right) \tag{2.19}$$

The Cartesian stiffness, $K_{x}(q)$, is a positive-definite and symmetric matrix of size 6×6 and represents the apparent stiffness of a manipulator at its end-effector. The properties of positive-definiteness and symmetry are preserved as long as the Jacobian matrix, $J(q)$, is non-singular (i.e., $J^{-1}(q)$ exists). At static singularities (i.e., when $\det(J^T)=0$), the apparent stiffness of the manipulator gets extremely large resulting in an unstable and unsafe behavior, which may ultimately damage the environment and/or the robot itself. One of the practical solution in the context of the current work may be to not let the robot operate in configurations where singularities may occur. Proper analysis of the robot workspace and kinematic behavior needs to be performed before deploying the robot in a homecare environment, which usually involves human robot interaction (such as care-giving to elderly). One such measure of kinematic optimization of a manipulator is manipulability, which is discussed next.

2.3.4 Manipulability Measure

Manipulability of a robotic system is derived from a manipulator's kinematic properties and was first defined by Yoshikawa [56]. It is the product of the singular values of the Jacobian matrix. Later, the concept was extended to include dynamic characteristics of the robot as well.

Manipulability can help find out regions of the robot workspace that are safe and where the arm's configurations are away from singularities. These singularities are particularly detrimental to haptic interfaces because the end-point of a haptic interface becomes extremely stiff and it poses resistance to motion by the human operator. Similarly, for the slave robot, a singularity would result in extremely large velocities of the end effector for finite joint velocities. Safety is therefore compromised if proper analysis of both the master and slave robot's workspace is not carried out. The worst directions of the end effector motion in the workspace as pointed to by the singular values of the Jacobian must be avoided, and the master and slave robots must be put in safe home positions. Manipulability therefore can be treated as a dexterity measure for teleoperation.

Manipulability is best described and represented by a *manipulability ellipsoid* corresponding to a given pose (position and orientation). This ellipsoid describes the motion in Cartesian space as generated by unit joint velocities. Chiu in [57] used manipulability to describe the arm's force transmission capabilities.

A unit sphere in joints velocity $\dot{q} \in R^n$ is defined as

$$\| \dot{q} \|^2 = \dot{q}_1^2 + \dot{q}_2^2, \ldots, \dot{q}_n^2 \leq 1 \tag{2.20}$$

which using Equation 2.12 can be projected into Cartesian space as

$$\dot{x}^T (JJ^T)^{-1} \dot{x} \leq 1 \tag{2.21}$$

This refers to an ellipse in the Cartesian space, called *velocity ellipsoid*, and it characterizes the achievable velocities in Cartesian space, $\dot{x} \in R^m$, subject to the constraint $\| \dot{q} \|^2 \leq 1$. As Jacobian is a function of joint angles q, it means there is one ellipsoid for each joint configuration q and its corresponding Cartesian position $x = f(q)$. The lengths and the directions of the principal axes of this ellipsoid represent feasible velocities in the Cartesian space. Mathematically, the magnitudes are the square roots of the eigenvalues $\sigma_{\min}, \sigma_{\max}$ of the term JJ^T in the ellipsoid equation (2.21). The axes of the ellipsoid are given by the eigenvectors $v_{\min}, v_{\max}$ corresponding to the eigenvalues. Yoshikawa [56] considered the volume of this ellipsoid, given by $\mu(q) = \sqrt{\det(JJ^T)}$, as a measure of manipulability and singularity avoidance and proposed that this measure can be maximized by exploiting the kinematic redundancy of a manipulator.

Similarly a *force manipulability ellipsoid* is an ellipsoid that is defined as the projection of a unit sphere in joint torques $\|\tau\|^2 \leq 1$.

$$F^T(JJ^T)F \leq 1 \tag{2.22}$$

The force ellipsoid represents the directions of feasible forces in the Cartesian space. The principal axes are now obtained from eigen-decomposition of $(JJ^T)^{-1}$. This is in contrast to the eigen-decomposition of JJ^T as in the case of the velocity ellipsoid. It is because of this reciprocity that the eigenvectors (directions of the ellipsoids) of both force and velocity ellipsoids are the same but their eigenvalues are the reciprocal of each other. The principal axes of one ellipsoid coincide with those of the other, but their lengths are in inverse proportions to each other. This is known as force/velocity duality [57]. This is in line with the conservation of energy principle, if the manipulator is viewed as a mechanical power transformer.

2.3.5 Kinematic Models of WAM Arm and Haptic Interface

Figure **2.6** shows the kinematic chain of the Barrett WAM arm. The D-H parameters are given in Table **2.3**.

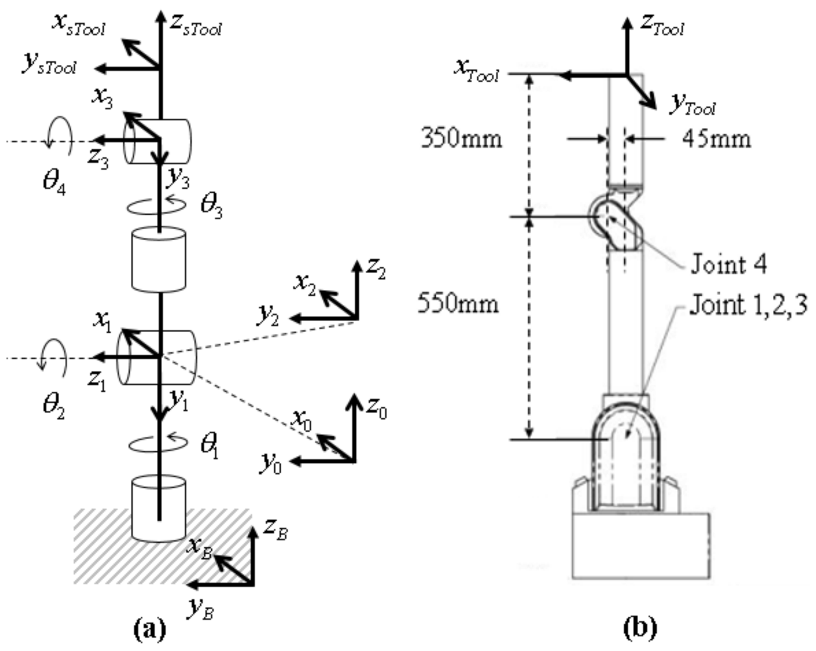

Figure 2.6 Kinematic chain representation of the WAM arm. (a) rear view, (b) side view.

Table 2.3 The D-H parameters of the WAM arm

Axis	θ	d	a	α
1	θ_1	0	0	$-\frac{\pi}{2}$
2	θ_2	0	0	$\frac{\pi}{2}$
3	θ_3	d_3	a_3	$-\frac{\pi}{2}$
4	θ_4	0	$-a_3$	$\frac{\pi}{2}$

Figure 2.7 shows the kinematic chain representation of the PHANToM haptic interface. The D-H parameters are given in Table 2.4.

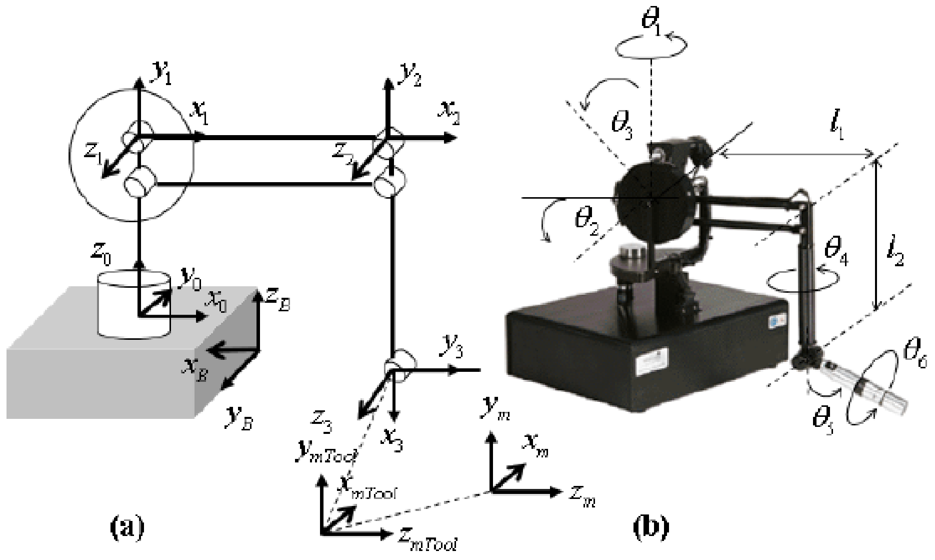

Figure 2.7 Kinematic chain representation of the PHANToM haptic interface. (a) Frames assigned, (b) Joint angles.

Table 2.4 The D-H parameters of the PHANToM haptic interface.

Axis	θ	d	a	α
1	θ_1	l_2	0	$\frac{\pi}{2}$
2	θ_2	0	l_1	0
3	$1.5\pi - \theta_2 + \theta_3$	0	l_2	0

Detailed derivations of the homogeneous transformations and the Jacobian matrices for both manipulators are given in Appendices A and B.

Workspace Mapping for Teleoperation

The tool (end effector) Cartesian coordinate frame $x_{mTool}y_{yTool}z_{zTool}$ of the PHANToM interface moves relative to its fixed frame $x_m y_m z_m$. Similarly the WAM tool moves with respect to its own fixed frame $x_0y_0z_0$. In order to relate the motions of both tools, their motion must be relative to a common frame of reference. Choosing the WAM's frame $x_0y_0z_0$ as a reference (see Figure 2.8), the following homogeneous transformation relates the two motions:

$$T_m^0 = T_{sTool}^0 T_{mTool}^{sTool} T_m^{mTool} \tag{2.23}$$

where T_{mTool}^{sTool} is a constant transformation, which is chosen according to the task. It maps the coordinates of the PHANToM tool to those of the WAM tool. One choice of T_{mTool}^{sTool} is:

$$T_{sTool}^{mTool} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.24}$$

which states that the $x, y,$ and z-axes of the WAM arm are mapped onto the $y, x,$ and z-axes of the PHANToM interface, respectively.

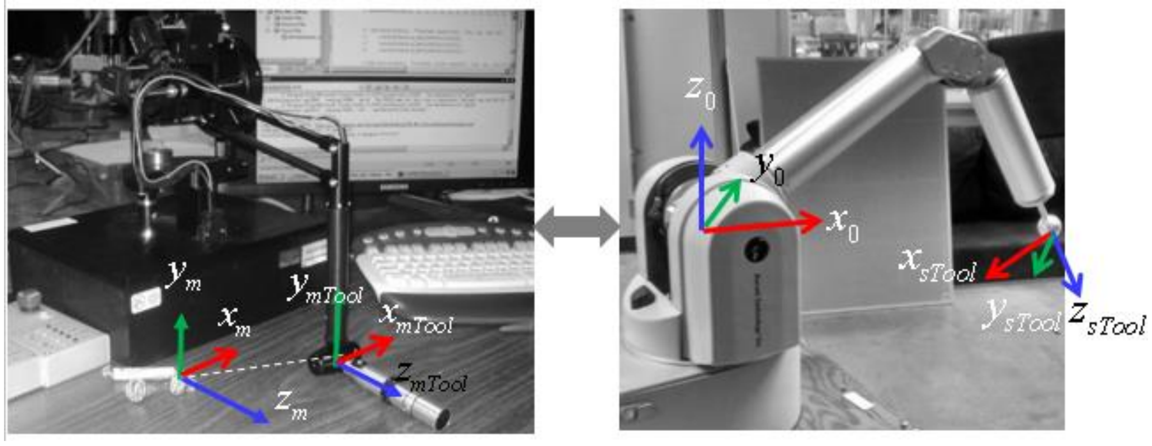

Figure 2.8 Workspace mapping for haptic teleoperation.

2.3.6 Experimental Results

Experimental results of the developed physical setup at IAL are shown in Figure 2.9 to Figure 2.12. These show perfect tracking by the slave of the trajectory as commanded by the master in all Cartesian directions i.e., *x*, *y*, and *z*. Keeping track of WAM's manipulability during teleoperation helped avoid the singular configurations and resulted in safe and reliable teleoperation. This feature is even more helpful in later chapters when WAM will be impedance controlled unlike here, where it is position controlled.

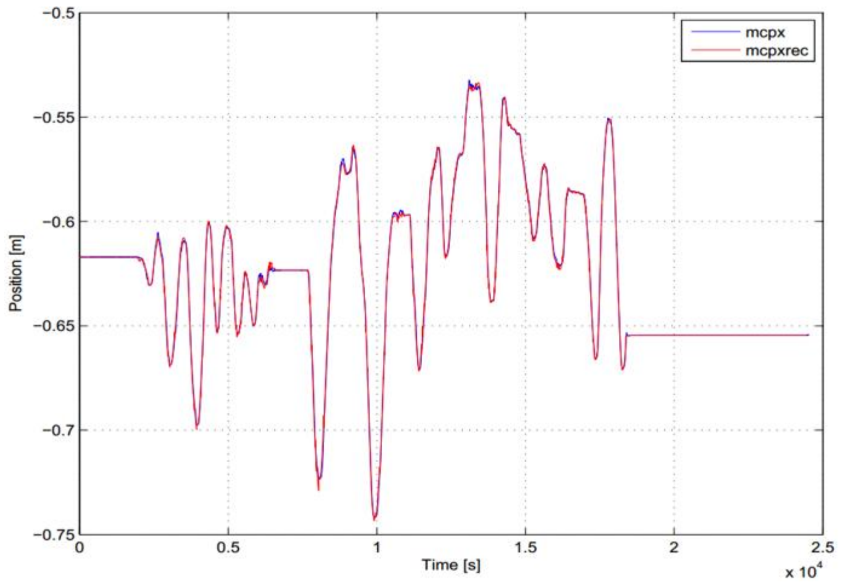

Figure 2.9 Master-slave teleoperation: *x*-Position tracking.

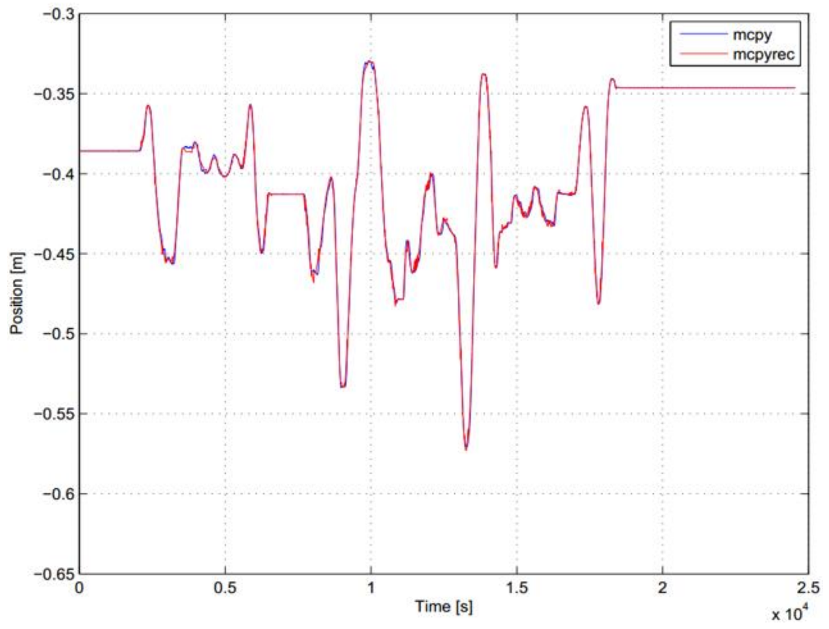

Figure 2.10 Master-slave teleoperation: *y*-Position tracking.

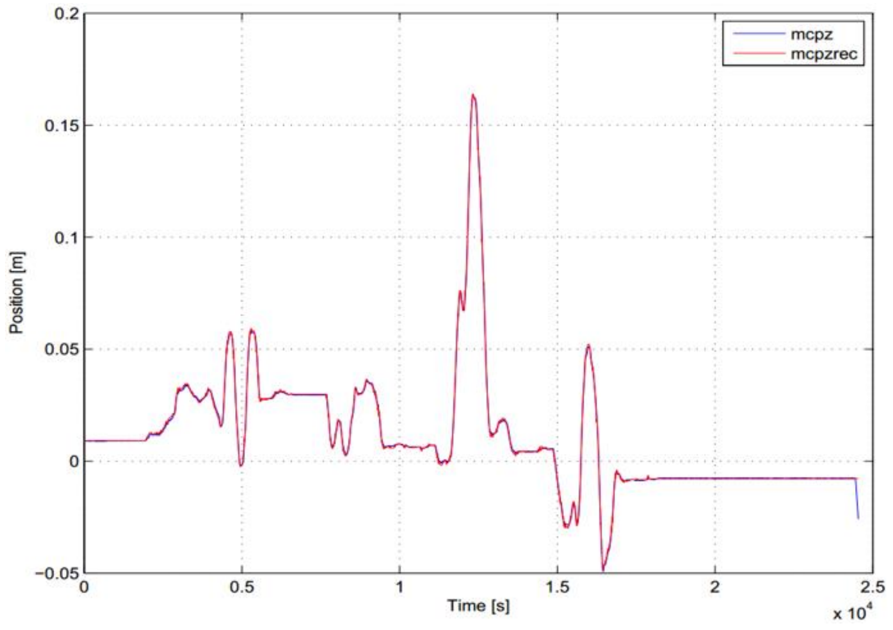

Figure 2.11 Master-slave teleoperation: *z*-Position tracking.

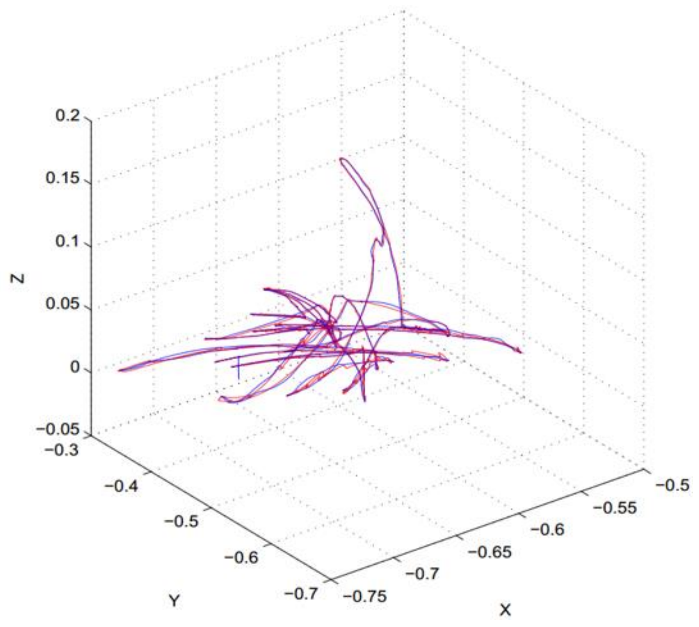

Figure 2.12 Master-slave teleoperation: tracking in 3D space.

2.4 Derivation of Dynamic Models

The Manipulator dynamic model provides a mathematical description of the manipulator dynamics in the form of differential equations. The model accounts for time-dependent changes in the state of the system. It comprises *n* coupled second-order nonlinear differential equations (one for each joint). These equations, also called *Euler-Lagrange equations*, are found from the total energy of the system and they describe the relationship between the internal or external forces that act on the arm (input torque, gravity, centrifugal, Coriolis, and friction) and the resulting motion (displacement, velocity, and acceleration). A dynamic model is useful in design, simulation, and control of the manipulator. The following sub-sections provide formulation and identification of the dynamic model which will be used for control design purposes later, in Chapter 3.

2.4.1 Lagrangian Dynamic Formulation

In this section, the formulation of manipulator dynamics is developed using the energy-based (Euler-Lagrangian) approach. Two versions of the robot dynamic problem will be discussed: direct dynamic problem and inverse dynamic problem. They are shown in Figure 2.13. The direct dynamic model, using a given trajectory, $q, \dot{q}, \ddot{q}$, finds the required vector of joint torques, τ. This model is used to simulate the manipulator dynamics in Matlab/Simulink for analysis purposes. The second version, inverse dynamic model, calculates, for a given torque vector τ, the resulting motion of the manipulator, $q, \dot{q}, \ddot{q}$. This model is used in model-based control design in subsequent chapters.

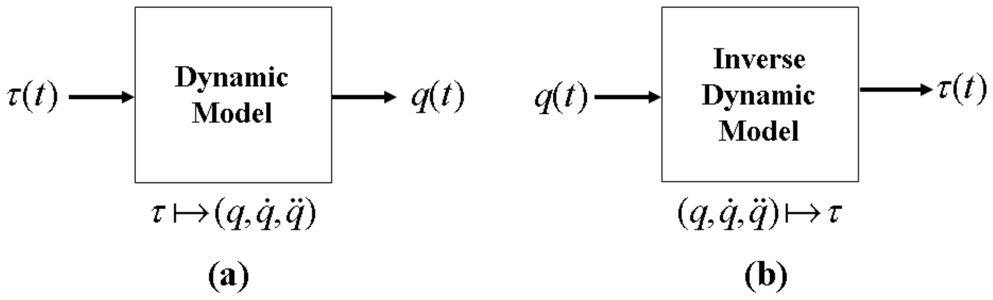
$\tau(t)$
Dynamic Model
$q(t)$
$\tau \mapsto (q, \dot{q}, \ddot{q})$
(a)
$q(t)$
Inverse Dynamic Model
$\tau(t)$
$(q, \dot{q}, \ddot{q}) \mapsto \tau$
(b)

Figure 2.13 Robot dynamic models. (a) Direct, (b) Inverse.

Let $q(t)$ be the set of n generalized coordinates (joint variables), then define the *Lagrangian* L as,

$$L(q, \dot{q}) = K(q, \dot{q}) - P(q) \tag{2.25}$$

where K and P are the total kinetic and potential energies of the system. Using the calculus of variations it can be shown that [54],

$$\frac{d}{dt}\frac{\partial L(q, \dot{q})}{\partial \dot{q}_k} - \frac{\partial L(q, \dot{q})}{\partial q_k} = \tau_k, \quad 1 \le k \le n \tag{2.26}$$

where τ_k is the input torque acting on joint k. Equation 2.26 is also known as the *Euler-Lagrange* equation.

The kinetic energy of an n-link manipulator can be computed as,

$$K(q,\dot{q}) = \sum_{i=1}^{n} \frac{(v_i)^T m_i v_i}{2} + \frac{(\omega_i)^T I_i^0 \omega_i}{2} \tag{2.27}$$

where m_i is the mass of link i at its Centre of Mass (CoM). v_i is the linear velocity of the CoM, ω_i is angular velocity about the CoM, and I_i^0 is the 3×3 inertia tensor, all expressed in base frame $O_0X_0Y_0Z_0$. Using the kinematic model of the manipulator, Equation 2.27 can be expanded into,

$$\begin{aligned} K(q,\dot{q}) \quad &= \frac{1}{2}\dot{q}^T\left[\sum_{i=1}^{n}\left\{m_i J_{v_i}(q)^T J_{v_i}(q) + J_{\omega_i}(q)^T R_i(q) I_i R_i(q)^T J_{\omega_i}(q)\right\}\right]\dot{q} \\ &= \frac{1}{2}\dot{q}^T M(q)\dot{q} \end{aligned} \tag{2.28}$$

where I_i represents inertia tensor in coordinate frame attached to the CoM of link i and is transformed into the base frame as $I_i^0 = R_i(q) I_i R_i(q)^T$. J_{vi} and $J_{\omega i}$ represent the link Jacobian (of link i) as given by Equation 2.8.

Matrix $M(q)$ in Equation 2.28 is the manipulator *inertia matrix* of size $n\times n$ and is always symmetric and positive-definite. As can be seen, the inertia matrix is configuration-dependent.

For a rigid-link manipulator, the potential energy P can be computed as,

$$P(q) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} m_i g^T r_{ci} \tag{2.29}$$

where r_{ci} is the position of the CoM of link i and g is the gravity vector, both expressed in the inertial frame of reference.

The substitution of Equations 2.28 and 2.29 into Equation 2.25 results in the following expression for the Lagrangian,

$$\begin{aligned} L(q,\dot{q}) &= K(q,\dot{q}) - P(q) \\ &= \frac{1}{2}\dot{q}^T M(q)\dot{q} - P(q) \\ &= \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n} m_{ij}(q)\dot{q}_i\dot{q}_j - P(q) \end{aligned} \tag{2.30}$$

where $m_{ij}(q)$ refers to ij^{th} element of the inertia matrix $M(q)$.

The Euler-Lagrangian equation (Equation 2.27) now becomes,

$$\sum_{j=1}^{n} m_{kj}\ddot{q}_j + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\left\{\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j} - \frac{\partial m_{ij}}{\partial q_k}\right\}\dot{q}_i\dot{q}_j + \frac{\partial P}{\partial q_k} = \tau_k \tag{2.31}$$

By letting

$$c_{ijk} \equiv \frac{1}{2}\left(\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j} - \frac{\partial m_{ij}}{\partial q_k}\right) \tag{2.32}$$

and

$$g_k = \frac{\partial P}{\partial q_k} \tag{2.33}$$

Equation 2.31 can be re-written as,

$$\sum_{j=1}^{n} m_{kj}(q)\ddot{q}_j + \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i\dot{q}_j + g_k(q) = \tau_k, \quad 1 \le k \le n \tag{2.34}$$

which is the dynamic Equation of Motion (EoM) of an n-link manipulator and is commonly written in a concise (vector) form as,

$$M(q)\ddot{q}+C(q,\dot{q})+G(q)=\tau \tag{2.35}$$

If the effects of friction are modeled as $B(\dot{q})$ and external forces/moments F_{ext} (Section 2.3.3) are also considered, then equation 2.35 can be re-written as,

$$M(q)\ddot{q}+C(q,\dot{q})+B(\dot{q})+G(q)=\tau-J^T(q)F_{ext} \tag{2.36}$$

here

$M(q)$: the inertia matrix,found using the link Jacobian $J_c(q)$
$C(q)$: the velocity cross – coupling matrix, found using the inertia matrix
$G(q)$: the gravity loading vector, found using the link Jacobian $J_c(q)$
$B(\dot{q})$: the friction vector that comprises both viscous and Colulomb frictions
τ : the input torque vector from actuators

For simplicity, Equation 2.36 can be re-written as

$$M(q)\ddot{q}+n(q,\dot{q})=\tau-J^T(q)F_{ext} \tag{2.37}$$

where the Coriolis, centripetal, friction, and gravity torques are grouped as $n(q,\dot{q})=C(q,\dot{q})+B(\dot{q})+G(q)$.

Example Simulation: Cartesian PD Control of WAM Arm with Dynamic Compensation

In this example simulation, a Cartesian PD controller of the 4-DOF WAM arm is designed with compensation for nonlinear dynamics in the joint-space. The purpose of this simulation is to evaluate the accuracy of the kinematic and dynamic models that have been derived. The dynamic parameters of the WAM arm as provided by the manufacturer are used (see Appendix B) to compute the model. The Robotic Toolbox in [58] is used to simulate the arm in MATLAB/SIMULINK (see Appendix B.3 for the code). Block diagram of the implemented controller is shown in Figure 2.14. The desired Cartesian trajectory is a circle of radius r in the

XY plane centered at point (c_x, c_y). If the circle needs to be traced out in time T, then the trajectory can be expressed as,

$$x_d(t) = [c_x + r\cos\frac{2\pi t}{T}, c_y + r\sin\frac{2\pi t}{T}, 0]^T, 0 \le t \le T \tag{2.39}$$

The arm is initially at configuration $q_{init} = [\text{-}0.0048 \quad \text{-}1.9912 \quad \text{-}0.0112 \quad 3.1227]^T$ (rad). It is not shown in the block diagram (Figure 2.14), but the arm interacts with a simulated environment that is modeled as a linear spring i.e., $F_e = k_e(x_e - x)$ where $k_e = 10000$ N/m. The environment is located in the same XY-plane where the desired circular trajectory is located. Nominal (uncompressed) position of environment is $x_e = -0.6$ m. Contact is made as soon as the x Cartesian position of end effector gets smaller than the nominal position i.e., when $x < x_e$.

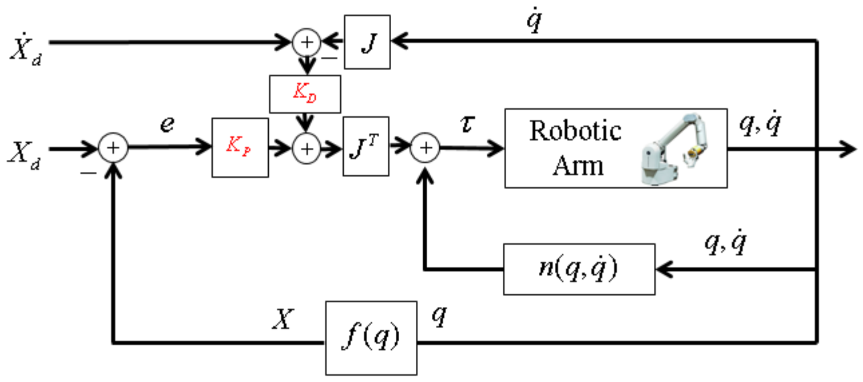

Figure 2.14 Cartesian PD control of WAM arm with inverse dynamic compensation.

The control law is computed as,

$$\tau = J^T(q)(K_p e + K_D \dot{e}) + n(q, \dot{q}) \tag{2.40}$$

with the controller gains selected as $K_p = 1000$ N/m and $K_D = 100$ Ns/m. Closed-loop dynamics (by substituting Equation (2.40) into (2.38)) are obtained as,

$$M(q)\ddot{q}+n(q,\dot{q}) = J^T(q)(K_p e+K_D\dot{e})+n(q,\dot{q})-J^T F_{ext} \tag{2.41}$$

$$M(q)\ddot{q} = J^T(q)(K_p e+K_D\dot{e})-J^T F_{ext} \tag{2.42}$$

$$\ddot{q} = M^{-1}(q)\left[J^T(q)(K_p e+K_D\dot{e}-F_{ext})\right] \tag{2.43}$$

As can be seen from Equation 2.43, the impedance controlled arm operates with its natural inertia and the desired impedance. This will be explained in detail later in Chapter 3.

Figures 2.15-2.20 show the resulting plots. As shown in Figure 2.15, 2.17, and 2.18, the given Cartesian trajectory (a circle in Cartesian space) is perfectly tracked. As the arm is not stiffer than the environment, it does not penetrate into the environment. Joints path is shown in Figure 2.19 with the corresponding manipulability index in Figure 2.20. From the manipulability index, it can be noticed that the distance to singularity is very small (less than 0.02) when the arm is making contact with the environment. It means the configuration of the arm was not optimized for interaction and therefore the manipulability is limited.

The dynamic model simulated in this example does not include the dynamics due to friction. In reality, such a perfect tracking and ideal cancellation of the manipulator nonlinear dynamics is almost impossible to achieve due to unmodeled dynamics, errors in dynamic parameters, and variations in the load. A more sophisticated approach is therefore needed that can experimentally identify the dynamic model to overcome the aforementioned issues. This is a first step toward any model-based control, including impedance control and is covered in the next section (Section 2.5).

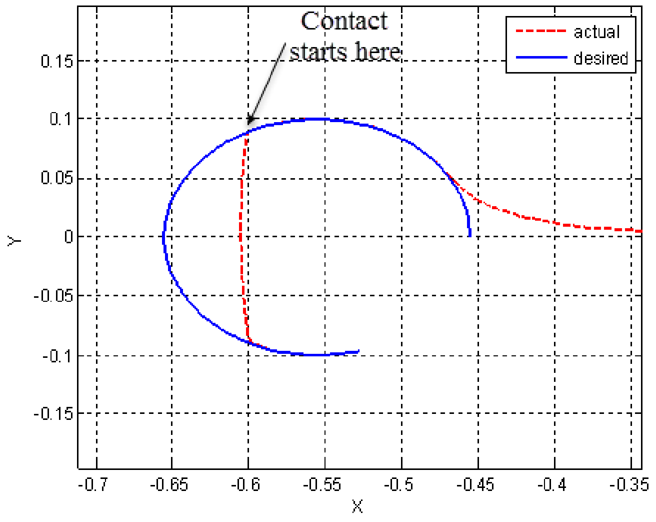

Figure 2.15 Tracking of the desired Cartesian trajectory and interaction with the environment.

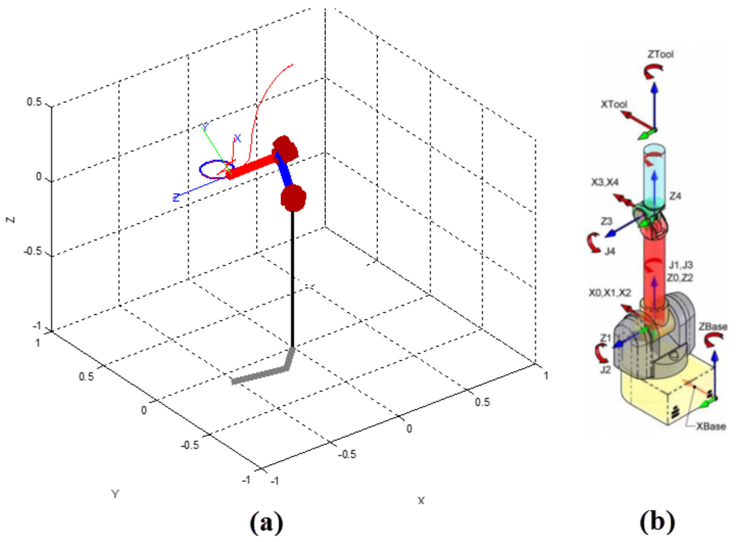

Figure 2.16 Animated WAM arm while tracking the given Cartesian trajectory. (a) WAM in initial configuration, (b) WAM frames of references, (*from* [51]).

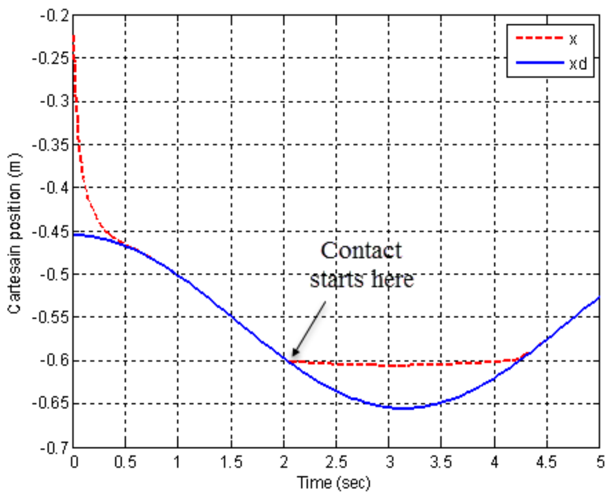

Figure 2.17 Tracking of the given Cartesian trajectory, *x*-direction.

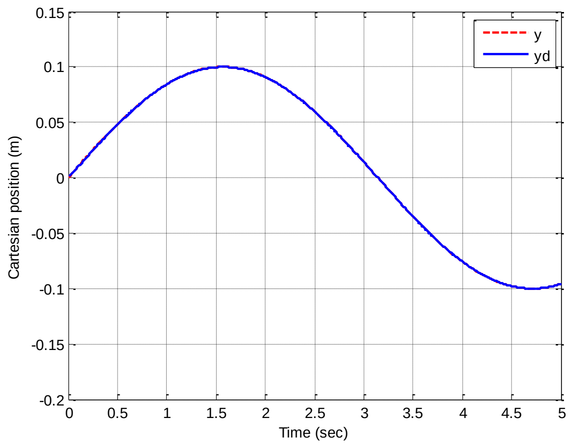

Figure 2.18 Tracking of the given Cartesian trajectory, *y*-direction.

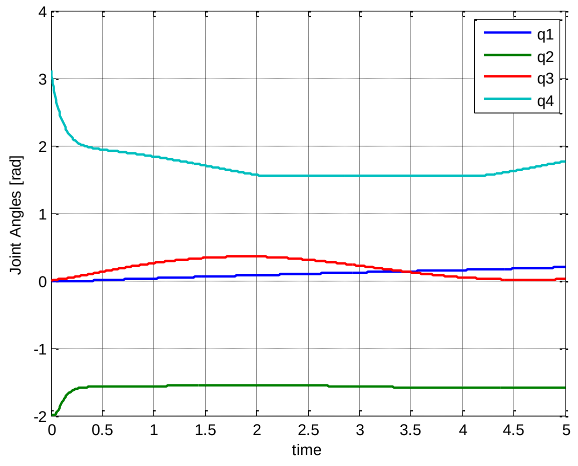

Figure 2.19 Joint angles during tracking.

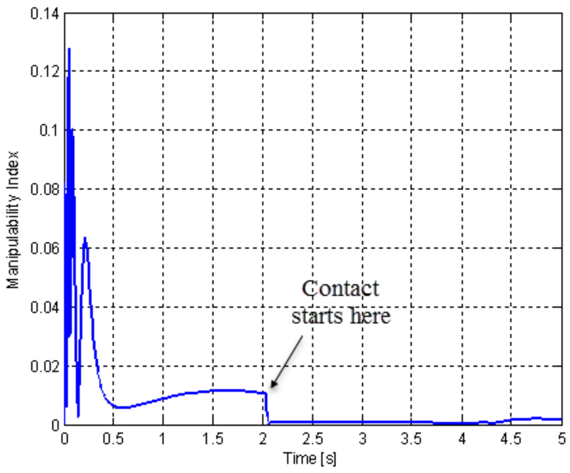

Figure 2.20 Manipulability index during tracking.

2.4.2 Operational Space Formulation

Cartesian control techniques require the dynamic model (Equation 2.36) to be described in Cartesian space instead of the joint space. The operational or Cartesian space formulation of the dynamic model relates the acceleration of the end-effector to the forces and moments acting on the end-effector, all represented in the Cartesian or tool space. This also forms a basis for impedance control of the manipulator, which is the subject of Chapter 3. The general form of the dynamic model in Cartesian space is given as [59],

$$M_x(q)\ddot{X} + C_x(q,\dot{q}) + G_x(q) = F \tag{2.44}$$

where

$$F = J^{-T}(q)\tau$$

$$M_x(q) = J^{-T}(q)M(q)J^{-1}(q) \tag{2.45}$$

$$C_x(q,\dot{q}) = J^{-T}(q)C(q,\dot{q}) - M_x(q)\dot{J}(q,\dot{q})\dot{q} \tag{2.46}$$

$$G_x(q) = J_i^{-T}G(q) \tag{2.47}$$

Given that, $X = [x, y, z, \alpha, \beta, \gamma]^{\mathrm{T}} \in \mathbb{R}^6$ is a vector of the position and orientation of the end-effector that is related to the manipulator's joint angles $q = [q_1, q_2, ..., q_n]^{\mathrm{T}}$ through the forward kinematic model as $X = f(q)$, the Cartesian acceleration, $\ddot{X}$, can be derived through differentiation as,

$$X = f(q) \tag{2.48}$$

$$\dot{X} = J(q)\dot{q} \tag{2.49}$$

$$\ddot{X} = J(q)\ddot{q} + \dot{J}(q,\dot{q})\dot{q} \tag{2.50}$$

It is this transformation of motion from joint coordinates to tool coordinates that has resulted in the operational-space formulation (Equation 2.44) of the dynamic model.

2.4.3 Linear Parameterization

The dynamic model of a robot can be factored into a vector of dynamic parameters of the robot, π, and a matrix that consists of joint position, velocities, and accelerations, $Y(q,\dot{q},\ddot{q})$ as

$\tau = Y(q,\dot{q},\ddot{q})\pi$. Written in this form, the dynamic model is linear in π. This form of formulation is usually required for experimental identification of the dynamic model.

Defining the dynamic coefficients vector for the WAM arm as (i being the joint number and $i = 1,\dots,n$),

$$\pi_i = \left[I_{ixx}, I_{ixy}, I_{ixz}, I_{iyy}, I_{iyz}, I_{izz}, m_i, m_i r_{ix}, m_i r_{iy}, m_i r_{iz} \right]^T \tag{2.51}$$

The Lagrangian in Equation 2.30 is factored into π_i (a vector comprising of all constants) and $y(\theta,\dot{\theta})$ as:

$$L = \sum_{i=1}^{n} y_i(q,\dot{q})^T \pi_i \tag{2.52}$$

The new dynamic equation for joint k now becomes:

$$\sum_{i=1}^{n} \pi_i^T \frac{d}{dt}\frac{\partial y_i(q,\dot{q})}{\partial \dot{q}_k} - \pi_i^T \frac{\partial y_i(q,\dot{q})}{\partial \dot{q}_k} = \tau_k$$

$$\sum_{i=1}^{n} y_{ki}(q,\dot{q},\ddot{q})^T \pi_i = \tau_k$$

$$Y(q,\dot{q},\ddot{q})\pi = \tau \tag{2.53}$$

Equation 2.53 is commonly used for model identification based on least-square techniques, which estimate the unknown parameters in π given the regressor, which consists of all measurements , $Y(q,\dot{q},\ddot{q})$.

2.5 Identification of Dynamic Model

System identification aims at building a mathematical model of a dynamic system based on given measurements of the system inputs and outputs. It can be categorized as 1). Identification based on assumed model structure (as given by Equation 2.53) and experimental data, 2). Identification completely based on experimental data assuming no prior knowledge of the system. In the first category, which can be referred to as *classical identification methods*, a rigid body model such as in Equation 2.53 is derived. The unknown parameters in π are then estimated a statistical technique using given measurements of the joint position, velocities, and accelerations corresponding to known input torques τ. While this method has been traditionally used in robotics, it has some drawbacks. First, as the method does not have any *learning* capability associated with it, the quality of approximation remains poor if the parameters are poorly estimated or the unmodeled dynamics in the system are significant. The models obtained through this method cannot adapt to unknown situations e.g., if a robot picks up an unknown payload or interacts with an unknown environments. Second, this method becomes tedious and difficult to use for large and complex systems (e.g., a 30-DOF robot). The parameterization is achieved once all the dynamic terms have been derived analytically. Last, the rigid body models in robotics usually require intensive numerical computations and are therefore not well suited for real-time applications like in haptic teleoperation.

The second category usually referred to as data-driven or intelligent system identification techniques are lately becoming popular due to the advances in the field of machine learning. These techniques can be run both online or offline, and are well-suited for real-time implementation due to the availability of sophisticated software libraries. It can be categorized into several types such as least-squares identification, Locally Weighted Projection Regression (LWPR), Gaussian-processes, artificial neural-networks, and so on. Sigaud et al. 2011 [16] and Nguyen-Tuong and Peters 2011 [14] present a comprehensive survey of model learning for robot control.

In robotic teleoperation applications, the problem of system identification is encountered at various stages and in various depths. Mostly these estimated models are highly nonlinear. In this thesis, two types of models are estimated: (1) model identification of the master and/or slave robots, (2) model estimation of the environment, which the master or the slave robot is interacting with. The challenging part here is building an accurate model and the ability to update it online.

2.5.1 Learning-based Identification

In this sub-section a complete data-driven, machine learning approach for model identification is discussed, which is used later in this thesis for learning the inverse dynamic model of the robot. Statistical multiple regression methods have recently become popular in the field of robotics. Furthermore, the availability of open-source, highly efficient and fast software packages such as LWPR [6] which can be run online and in real-time, have drawn attention of the researchers to develop practical robotic applications where model-based control is needed. These methods and algorithms have proved to be highly effective in learning the forward and/or inverse kinematic and dynamic models of a robot that has a complex mechanical design and has to interact with a complex physical environment. Even for common industrial robots, statistical learning offers a clear advantage over analytical methods of solving the inverse dynamic problem, which is a challenging task for the control designer and are also computationally heavy even for advanced hardware. Modern haptic applications are strict about the control loop timing and do not allow rates lower than $500\,\text{Hz}$. Real-time impedance control can particularly benefit from such techniques.

LWPR belongs to a more general category of locally weighted learning techniques that learn a variable (and optimized) set of weighted local linear models to estimate nonlinear models. They have the capability to handle high-dimensional input data spaces in which some of the dimensions may be irrelevant or even redundant. They also do not assume any model structure like in the case of parameter identification method described in the previous sub-section. In [60], the authors have compared the LWPR with ANN (more specifically the Radial Basis Function Network). They have found that while the structure of LWPR is similar to that of a basis function network, the flexibility that the weights, widths, and the number of receptive fields (nodes) can be incrementally updated during adaptation, gives them an added advantage in applications where rapid changes in observed data exist. Their work focuses on the use of LWPR for the human motor adaptation to novel tasks and environments.

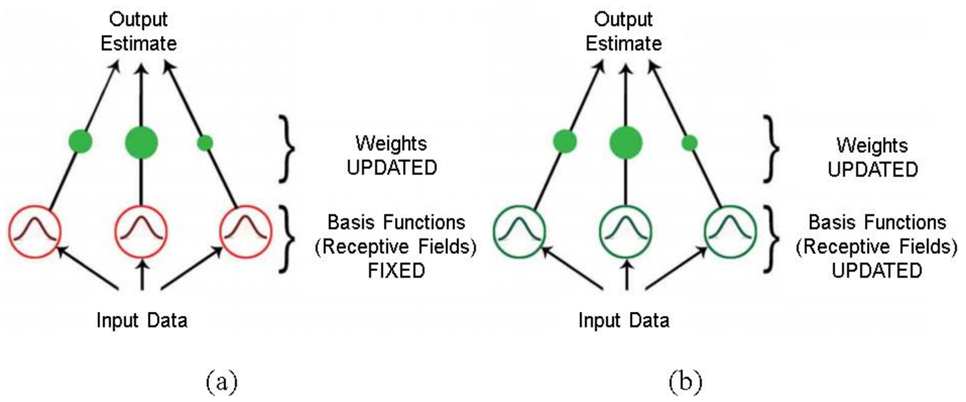

Figure 2.21 Comparison between LWPR and ANN. (a) ANN, (b) LWPR, *adapted from* [60]

In this thesis, the choice of using LWPR for online model learning can be justified due to the following reasons:

1. The inverse dynamic model to be learned is highly nonlinear. The algorithm is scalable to any extra degrees of freedom (an extra tool or gimbal) that are added to the arm.
2. The model must be able to adapt to changes over time (due to wear and tear of the robot), independently and without any interference.
3. Large amounts of experimental data is available for offline training. Once the application has been deployed in a homecare environment, model training and estimation can still be performed during the online operation of the arm in real time.
4. LWPR is fast and reasonably accurate. Due to the incremental learning approach, the algorithm can be implemented and updated online within the strict timing requirement of the control loop (which executes at frequencies between 500Hz to 1kHz).

In addition to the above-listed features, LWPR addresses the problem of nonlinear function approximation in high dimensional spaces with redundant input and irrelevant dimensions [61]. This is particularly useful in situations where the measurements (training data) collected from different sensors carry redundant information about the actual model.

2.5.2 Locally Weighted Regression

Suppose that the following nonlinear function needs to be approximated:

$$y = f(\mathbf{x}) + \eta \tag{2.54}$$

where $y \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^m$, and η is the white Gaussian noise with variance σ^2 i.e., $\eta \sim G(0,\sigma^2)$. LWPR then uses a set of spatially localized linear models that span the input space to predict estimate of the given function. These localized linear models are weighted using Gaussian distributions. As illustrated in Figure 2.22, for a given new input $\mathbf{x}$, the weight w_k corresponding to the k^{th} local linear model, also called a *receptive field*, is computed using the Gaussian kernel as

$$w_k(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mu_k)^T \mathbf{D}_k (\mathbf{x}-\mu_k)} \tag{2.55}$$

where D_k represents the distance metric and μ_k is the center of the k^{th} receptive field. The weight $w_k(\mathbf{x})$ defines the region of validity of the k^{th} local model and therefore is also called the activation function of input $\mathbf{x}$. During training, a new receptive field is created only if all the existing receptive fields do not yield activation above a certain threshold w_{gen} [6].

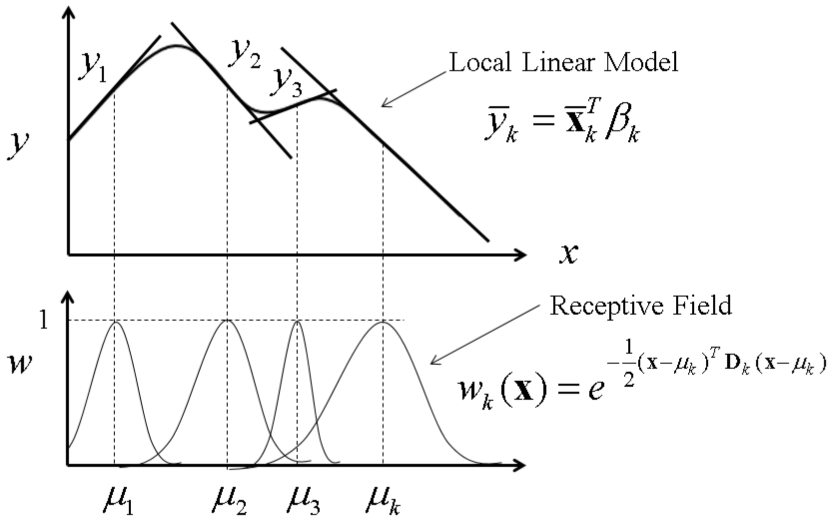
y_1
y_2
y_3
Local Linear Model
$\bar{y}_k = \bar{\mathbf{x}}_k^T \beta_k$
y
x
1
Receptive Field
w
$w_k(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mu_k)^T \mathbf{D}_k (\mathbf{x}-\mu_k)}$
μ_1
μ_2
μ_3
μ_k

Figure 2.22 Estimating nonlinear function with local linear models.

The output $\bar{y}_k$ of receptive field k is then calculated using univariate regression as

$$\bar{\mathbf{x}}_k^T = [(\mathbf{x}-\mu_k)^T, 1]^T \tag{2.56}$$

$$\bar{y}_k = \bar{\mathbf{x}}_k^T \beta_k \tag{2.57}$$

where β_k is the vector of estimated parameters for the k^{th} model.

The final predicted output value, $\tilde{y}_i = f(\mathbf{x})$, which is an approximation of the i^{th} element of the output $\tilde{y}$ in Equation 2.54, is then a weighted combination of K locally linear models and is given by:

$$\tilde{y}_i = \frac{1}{W(\mathbf{x})} \sum_{k=1}^{K} w_k(\mathbf{x}) \bar{y}_k(\mathbf{x}) \tag{2.58}$$

where $W(\mathbf{x}) = \sum_{k=1}^{K} w_k(\mathbf{x})$.

The number of receptive fields, K, the coefficients of regression, β, and weights w are learned and adjusted incrementally using a stochastic gradient descent algorithm where the cost function to be minimized is,

$$J_{\text{cost}} = \frac{1}{\sum_{i=1}^{M} w_i} \sum_{i=1}^{M} w_i \left\| y_i - \tilde{y}_{i,-i} \right\|^2 + \frac{\gamma}{N} \sum_{i,j=1}^{N} D_{ij}^2 \tag{2.59}$$

Here P is the inverted weighted covariance matrix of the input data and M is the number of data points seen during training.

In order to adjust a receptive field, the corresponding distance metric D is updated as

$$M^{n+1} = M^n - \alpha \frac{\partial J_{cost}}{\partial M} \tag{2.60}$$

$$D = M^T M \tag{2.61}$$

where M is an upper triangular matrix obtained from the Cholesky decomposition of D and α is the learning rate. Equation 2.61 ensures positive definiteness of D.

To reduce the dimensionality of the problem, the projected inputs z are obtained by removing the redundant and irrelevant dimensions from the input data. This subspace of lower dimensionality is used by the LWPR for learning instead of the original high-dimensional space.

Complete details of the algorithm are given in References [62] and [6].

Tuning the LWPR Parameters

There are several important parameters of the LWPR algorithm that need to be tuned. Some of the important parameters are listed in Table 2.5.

Table 2.5 LWPR parameters for tuning.

Parameter	**Description**
α	Learning rate of the gradient descent algorithm
γ	Multiplication factor in the cost function
D	Initial distance metric for newly created receptive fields
NormIn	Input normalization factor

The most important parameter to tune is *D* as it decides the shape and the size of the receptive field. The convergence properties and the speed of the algorithm strongly depend on how this parameter is being initialized. Too large a value of *D* means that the size of a receptive field is small and therefore the number of total receptive fields is very large. This can result in over-fitting of the data, thereby resulting in poor predictive performance. On the contrary, a small value means larger receptive fields and therefore smoother function, but it may cause slow convergence speed and can also have the local minima problem. The other two elements, α and γ , are also related to *D*. Too large a value of α can result in instability in convergence. Its value therefore may only be increased in situations where the MSE is decreasing, but the convergence is slow. Large values of γ will result in wider receptive fields (smaller distance metrics).

The last parameter to tune is *NormIn*. Normalization of the input data makes it dimensionless by dividing each dimension by its variance. Once the input data is normalized, the distances between the receptive fields and the input data is calculated.

Performance Measure

To evaluate the performance of the LWPR algorithm during training and validation, the approximation error can be used as a performance measure. It is the normalized mean squared error calculated as $nMSE = \frac{MSE}{\sigma^2}$ where MSE is the mean-squared error (between the actual and approximation output) and is calculated as $MSE = \frac{1}{N}\sum_{i=1}^{N}(y - \tilde{y})$. Also, $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(y - \tilde{y})$ is the variance of the outputs of the test set. During evaluation of the regression model by leave-one-cut cross validation, $nMSE = 0$ means perfect prediction. The $nMSE$ error is especially useful for query points (points that are not seen by the algorithm during training) as it shows confidence in the prediction. The algorithm can output the trace of the $nMSE$ error, the number of receptive fields added and pruned, and the amount of data processed during each cycle of execution. This information can later be used for purposes of plotting and analysis.

The LWPR algorithm can be executed in BATCH or ONLINE modes. In the BATCH learning mode, fixed training and testing data sets (recorded as ASCII files) are used for training and validation of the model that is being developed. During execution, the algorithm in the

BATCH mode still works incrementally and outputs the trace of the learning progress continuously. The ONLINE mode is particularly useful when a model, originally learnt in the BATCH mode, is to be updated during the real-time operation of the robot.

Example Simulation: Nonlinear Function Approximation with LWPR

In this example simulation, the two-dimensional nonlinear function $z = \max\left(\exp(-10x^2), \exp(-50y^2), 1.25\exp(-5(x^2+y^2))\right)$ is learned from a set of noisy data samples using the LWPR algorithm. The resulting plots are shown in Figure 2.23 for two different initial values of the distance metric *D*. These two values are selected such that the problem of underfitting and overfitting can be demonstrated. As can be seen, in the case of a low value of *D* (i.e., when *D* = 10), the resulting model (labeled as fitted function) does not generalize correctly from the training data. In other words, underfitting is observed due to the wider receptive fields. On the other hand, a relatively large value of *D* (i.e., when *D* = 70) results in a model that overfits the training data. Initialization of *D*, therefore, plays an important role in the performance of the LWPR algorithm.

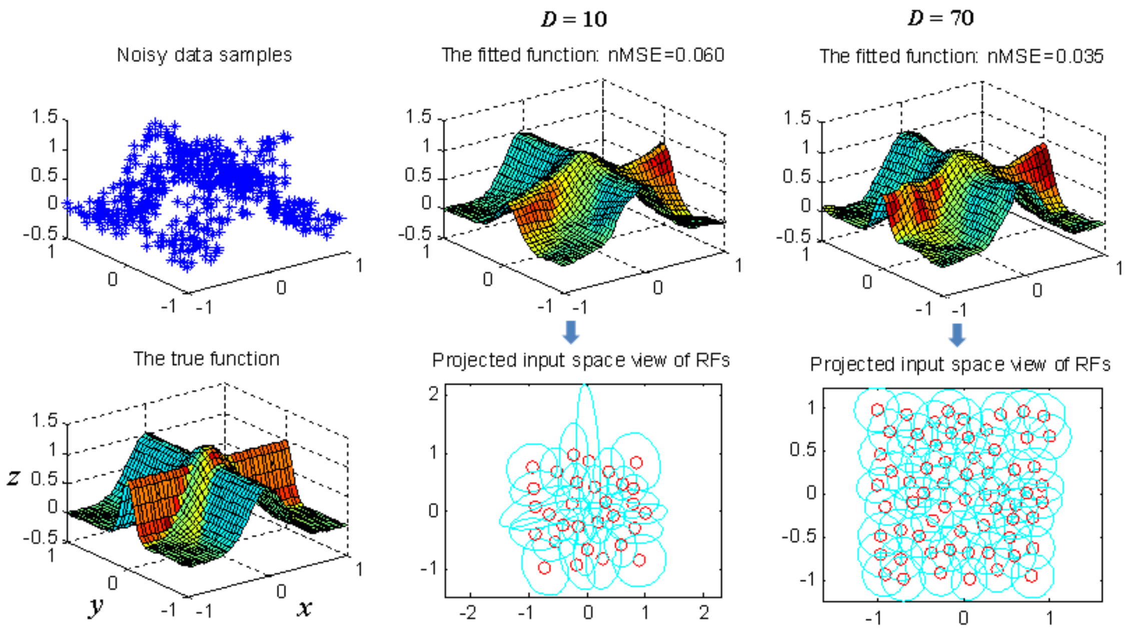

Figure 2.23 Example of nonlinear function approximation using LWPR

2.6 Results and Discussion

2.6.1 Experiments

Two sets of experiments are performed. In both these experiments, the “Teach and Play" feature of the WAM software library (the libbarrett library [50]) is utilized. Necessary modifications are made to the teach and play example that comes as part of the libbarrett library. During the TEACH mode, the WAM arm is taught a path by physically moving it in a random fashion with variable speed in all possible directions in the robot’s workspace as shown in Figure 2.24. The corresponding motion of all joints is shown in Figure 2.25. This is possible due to the high backdrivability feature offered by the WAM arm. Once the taught path is recorded, a spline is built between the recorded points in order to connect them smoothly. Then it is played back by the WAM arm. During the PLAY mode, the trajectory (Cartesian positions x and velocities $\dot{x}$, joint positions q, velocities $\dot{q}$, and accelerations $\ddot{q}$) along with the joint torques τ applied by the actuators are recorded from within the real-time control loop which runs at the rate of $500\,\text{Hz}$.

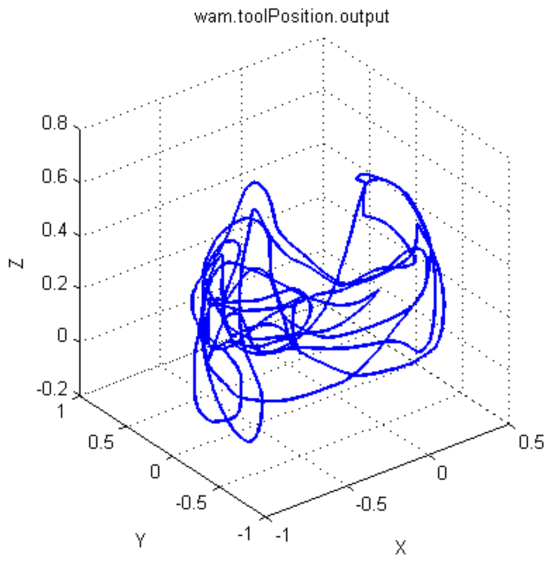

Figure 2.24 Taught Cartesian trajectory to the WAM arm (Experiments I & II).

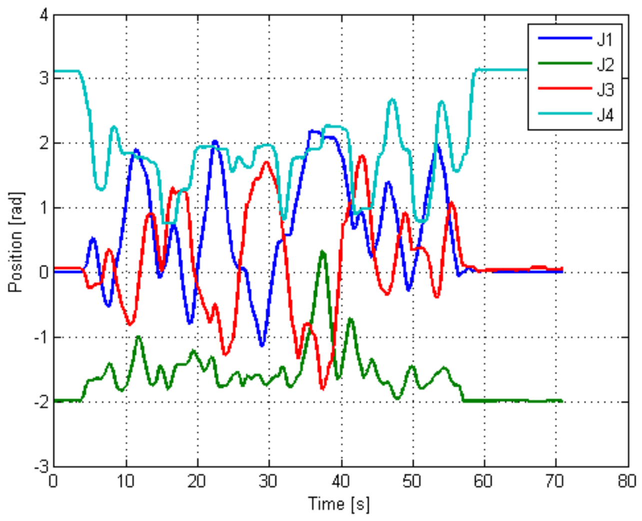

Figure 2.25 Joint positions for learning the inverse dynamic model (Experiments I & II).

Experiment I: Model Learning using Fast Trajectory

Figures 2.25 to 2.28 show the recorded joint positions, velocities, accelerations, and torques respectively. As can be seen from Figure 2.26, some of the joint velocities (e.g., joint 3) are as high as about $2\,\text{rads}^{-1}$. The WAM arm issues a fault signal (by turning on the red LED on its pendant) when the velocity of any joint exceeds the threshold value of $2\ \text{rads}^{-1}$. The corresponding joint accelerations and therefore joint torques are also significant (see Figures 2.27 and 2.28). While practically, any robotic arm would rarely execute such a large-amplitude and high frequency motion, the purpose of using a fast trajectory for learning the inverse dynamic model is to capture the high-frequency dynamics of the arm and also to analyze the performance of the learning algorithm in a worst case scenario. Moreover, assuming constant angular speed is not feasible in the context of haptic teleoperation where the velocities in the system actually originate from the hand of the human operator who is manipulating the haptic interface. The slave manipulator must have to track velocities received from the master side, even if they are high (but still under the safety threshold).

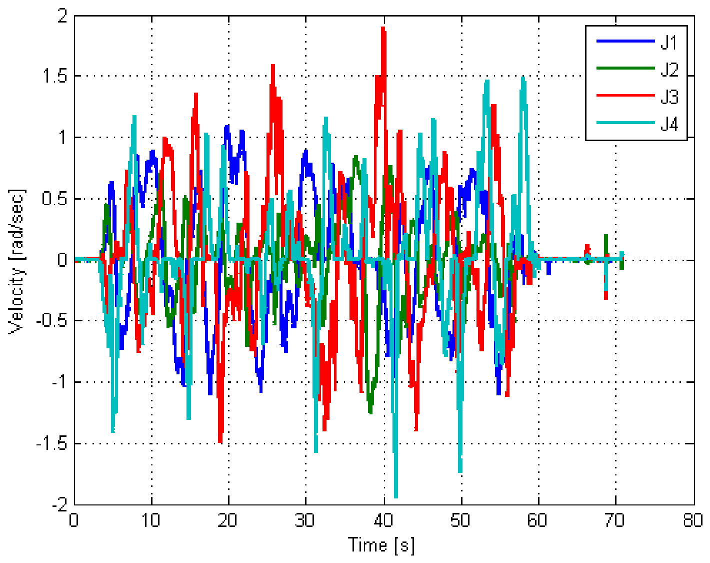

Figure 2.26 Joint velocities for learning the inverse dynamic model (Experiment I).

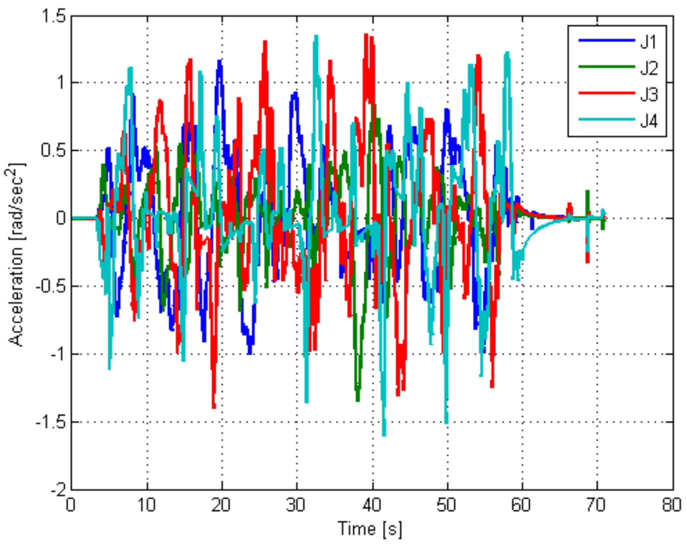

Figure 2.27 Joint accelerations for learning the inverse dynamic model (Experiment I).

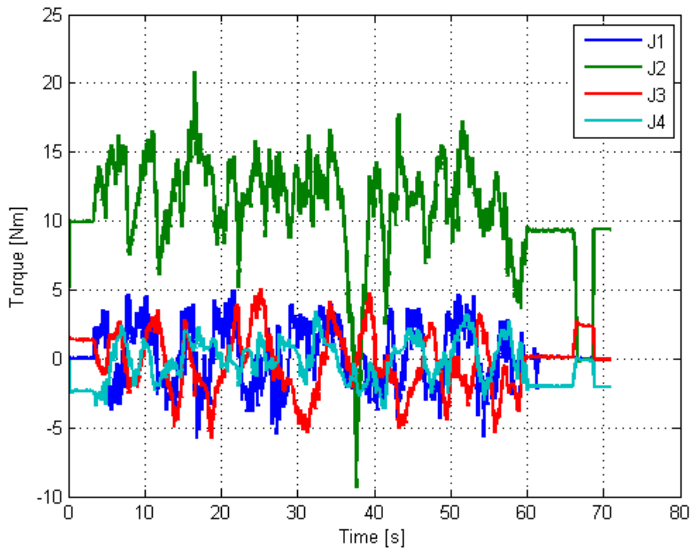

Figure 2.28 Joint torques for learning the inverse dynamic model (Experiment I).

Experiment II: Model Learning using Slow Trajectory

In this experiment, the same joint displacements as in Experiment I are used, but the desired trajectory is smoothen using a trapezoidal velocity profile between consecutive positions. The trapezoidal velocity profile ensures smooth and continuous motion with slower joints velocities and accelerations for the WAM to track. The resulting slow trajectory (velocities and accelerations) and the corresponding actuator torques are re-recorded and are shown in Figures 2.29, 2.30 and 2.31 respectively.

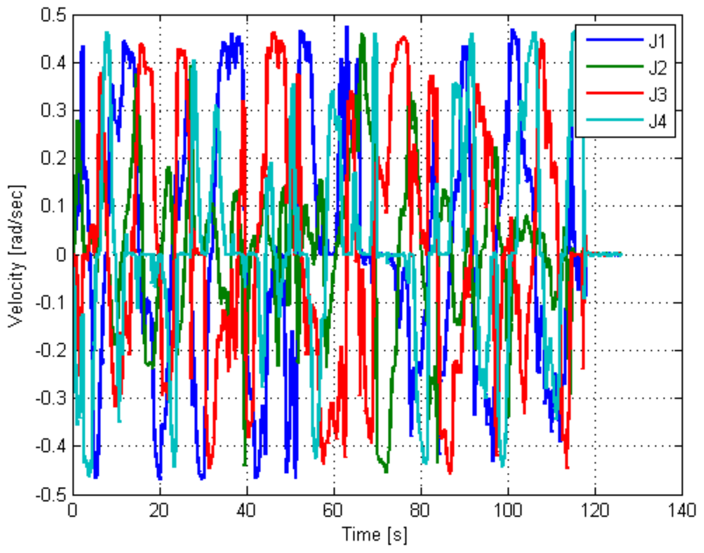

Figure 2.29 Joint velocities for learning the inverse dynamic model (Experiment II).

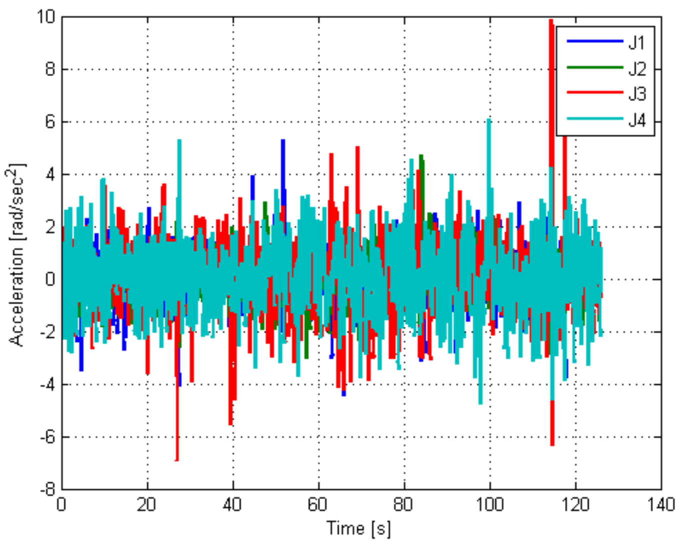

Figure 2.30 Joint accelerations for learning the inverse dynamic model (Experiment II).

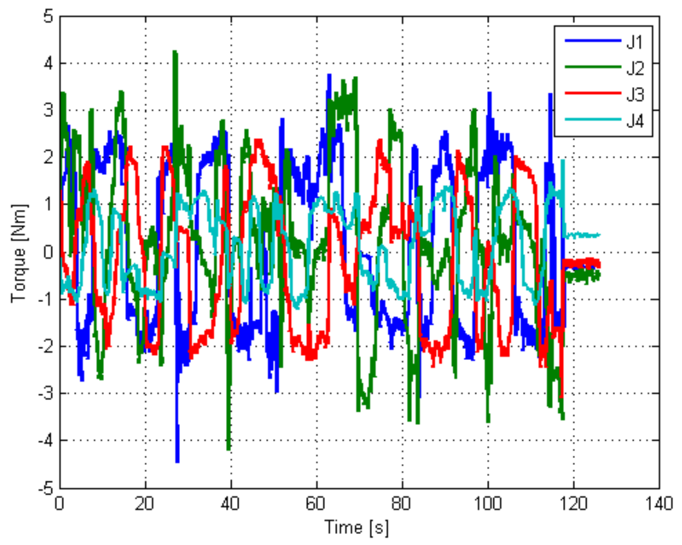

Figure 2.31 Joint torques for learning the inverse dynamic model (Experiment II).

2.6.2 Results and Discussion

In the fast trajectory case, model learning was difficult due to the corresponding large dynamic effects of inertia, Coriolis/centripetal and friction. The error convergence is shown in Figure 2.32 where the learned model still converges to an acceptable error level for all the joints. The corresponding number of receptive fields (RFs) during training is shown in Figure 2.33. As can be noticed, joint 2 had a complex model to approximate. The number of RFs therefore grows as the training continues. For the case of slow trajectory, the error convergence, as shown in Figure 2.34, and the number of RFs convergence was relatively smooth. For practical reasons and in the given context of homecare robotics, a fast trajectory as used in this experiment is never used due to safety reasons, but it has served the purpose of exposing the strengths and weaknesses of the learning algorithm. The successful convergence for both fast and slow trajectories show that the model learned with LWPR has a significant advantage over the analytical model and models identified with traditional techniques.

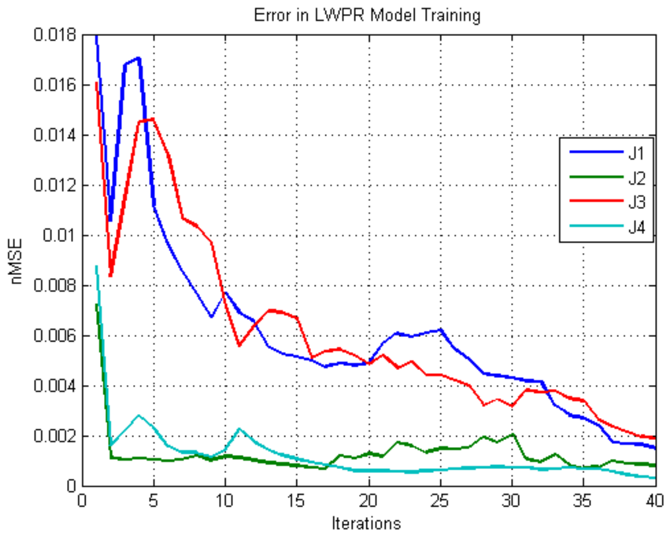

Figure 2.32 Error convergence in learning the inverse dynamic model (Experiment I).

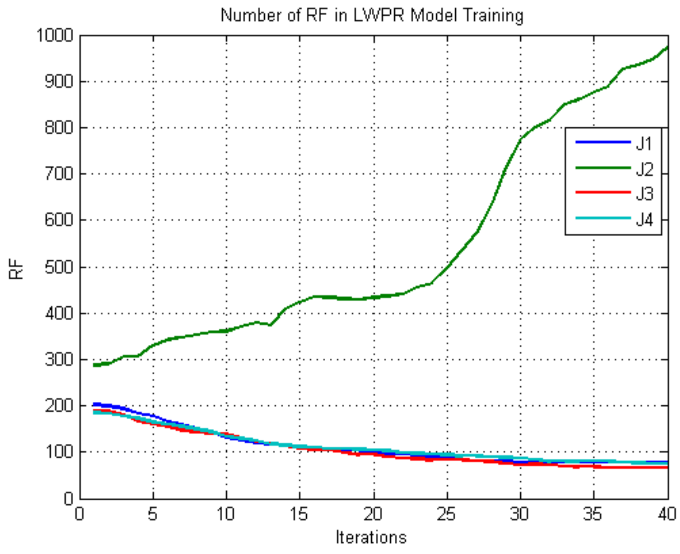

Figure 2.33 Evolution of number receptive fields in learning the inverse dynamic model (Experiment I).

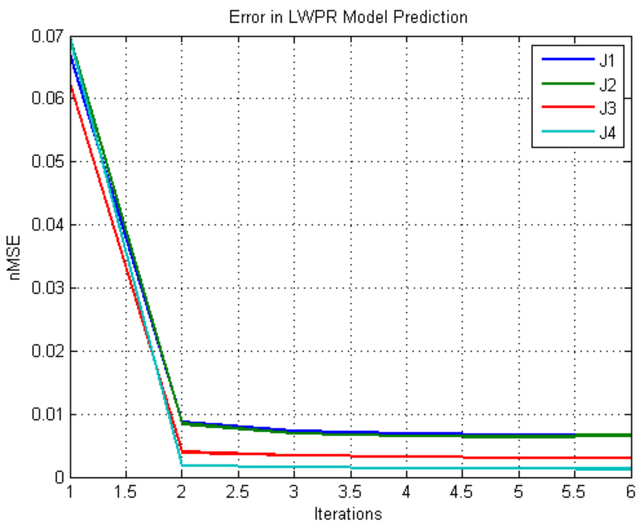

Figure 2.34 Error convergence in learning the inverse dynamic model (Experiment II).

2.7 Summary

This chapter first presented development of an experimental testbed for haptic teleoperation. This was followed by kinematic and dynamic modeling of the involved manipulators. A learning-based dynamic model identification technique was presented and its effectiveness compared to traditional techniques was studied and validated experimentally. The identified model will be used in impedance control in the next chapter.

Chapter 3: Interaction Control in Homecare Robotics

3.1 Introduction

Instead of the traditional control schemes such as motion and force control, for robot manipulators and haptic devices, active interaction control is used in cases where interaction with a dynamic environment is involved. Active interaction control can be categorized into indirect and direct force control. The difference between these two categories depends on whether there exists a direct force feedback loop or whether the force control is achieved via motion control. The first category (indirect force control) includes impedance control. There, using a desired impedance model, the contact force due to interaction with an environment is related to the deviation of the end-effector trajectory from the desired trajectory. Further categorization of impedance control is possible based on the order of the desired impedance model, which is a virtual mass-spring-damper system with adjustable parameters. In standard impedance control the desired model is second order and includes all components of impedance, desired inertia, damping, and stiffness. In case of a first-order model, only stiffness and damping are controlled. This will be subsequently called first-order impedance control. If the target behavior is provided in the form of an admittance model, the corresponding control method is called admittance control. This chapter focuses on the design and evaluation of these control schemes in the context of haptic teleoperation. An impedance controller based on learned inverse dynamics is also proposed.

3.2 Standard Impedance Control

Impedance control of robotic manipulators was originally introduced by Hogan in [34]. As defined in that work, an impedance controller regulates the dynamic behavior between the robot manipulator motion and the interaction force, which is the force exerted on the environment. An impedance-controlled manipulator can mimic the behavior of a human arm during interaction with an unknown environment. The block diagram of standard impedance control is given in Figure 3.1. As shown, the desired impedance block produces a dynamic force F_c corresponding to the desired relationship between the desired trajectory $X_d, \dot{X}_d$ and the contact force F_e.

For impedance control, the Cartesian-space formulation of the manipulator dynamic model (Equation 2.46) is used. An additional term, F_e, is added to the right side to represent the external force exerted by the environment on the end effector of the manipulator. Then the dynamic model of the manipulator becomes,

$$M_x(q)\ddot{X} + C_x(q,\dot{q}) + G_x(q) = F + F_e \tag{3.1}$$

The model given by Equation 3.1 would make the formulation of the impedance control laws (as derived next) consistent for manipulators with different DOFs and dissimilar kinematics. This is helpful for robot teleoperation where the master and the slave manipulators are usually different in workspace size, number of DOFs, and dynamic manipulability.

Impedance control requires a desired target dynamic behavior specified in the form of a given impedance function, usually called the *desired impedance model*, to be imposed on the manipulator during its interaction with the environment. The desired impedance model is given as

$$M_d(\ddot{X}-\ddot{X}_d)+B_d(\dot{X}-\dot{X}_d)+K_d(X-X_d)=F_e \tag{3.2}$$

$$\ddot{X}=\ddot{X}_d+M_d^{-1}\left[F_e+B_d(\dot{X}_d-\dot{X})+K_d(X_d-X)\right] \tag{3.3}$$

where M_d, B_d, and K_d are positive-definite matrices (usually diagonal) of size $N\times N$ representing the desired (virtual) inertia, damping, and stiffness, respectively; and X_d is the desired trajectory which the manipulator has to track while maintaining the contact with the environment in a desired fashion.

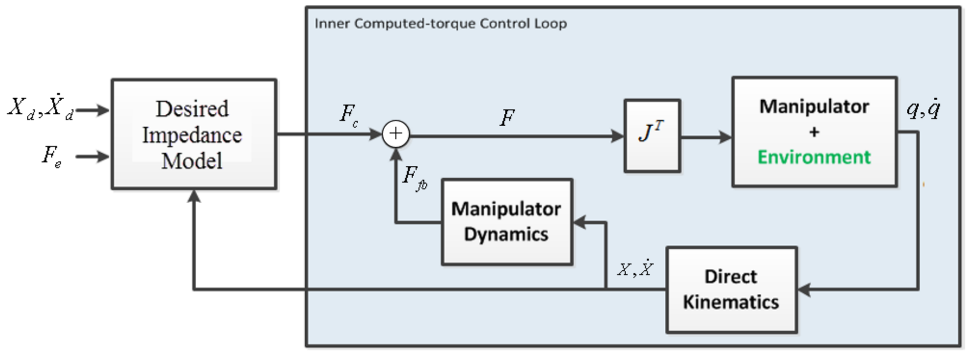

Figure 3.1 Standard impedance control.

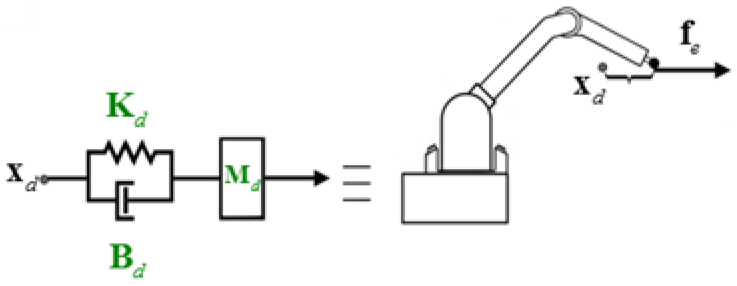

Figure 3.2 Impedance controlled manipulator – a spring-mass-damper system.

The control law in the standard impedance control scheme is based on *computed-torque control* and is given as

$$\tau = J^T(q)\left[M_x(q)\ddot{X} + C_x(q,\dot{q}) + G_x(q) - F_e\right] \tag{3.4}$$

Computed-torque control is a special application of *feedback linearization* of nonlinear systems.

Substituting Equation 3.3 into 3.4 results in the following control law:

$$\tau = J^T(q)M_x(q)\left[\ddot{X}_d + M_d^{-1}\left[F_e + B_d(\dot{X}_d - \dot{X}) + K_d(X_d - X)\right]\right]$$

$$+ J^T(q)C_x(q,\dot{q}) + J^T(q)G_x(q) - J^T(q)F_e \tag{3.5}$$

which can be simplified further using Equations 2.48 and 2.49 into

$$\tau = J^T(q)M_x(q)\left[(\ddot{X}_d - \dot{J}\dot{q}) + M_d^{-1}\left[F_e + B_d(\dot{X}_d - \dot{X}) + K_d(X_d - X)\right]\right]$$

$$+ C(q,\dot{q}) + G(q) - J^T(q)F_e \tag{3.6}$$

This is shown in the block diagram form in Figure 3.3. If the estimate of the nonlinear dynamic model is perfect, then the impedance-controlled manipulator behaves like a linear mass-spring-damper system as shown in Figure 3.2. Note that the only term of the robot's dynamic model in the Cartesian space is the inertia term $M_x(q)$. The rest of the dynamic terms ($C(q,\dot{q})\dot{q}$ and $G(q)$) are in the joint-space. This would ease the computation burden and therefore make real-time implementation of the control law possible. This will be discussed further in the section on experimental results.

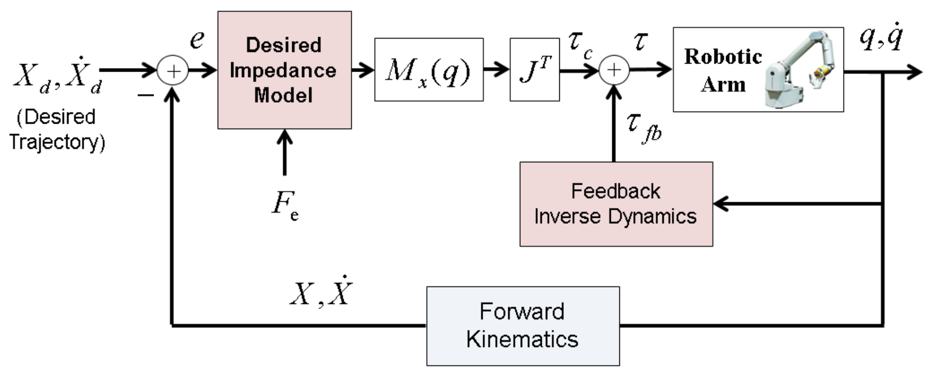

Figure 3.3 Standard impedance control.

3.3 First-order Impedance Control

If the focus in impedance control is only on the stiffness and damping behavior of the manipulator, then the standard impedance control simplifies to first-order impedance control. Here, the desired inertia is the same as the natural inertia of the manipulator.

The standard impedance control law in Equation 3.5 can be re-arranged as

$$\tau = J^T(q)M_x(q)\left[\ddot{X}_d - \dot{J}\dot{q}\right] + J^T(q)M_x(q)M_d^{-1}\left[B_d(\dot{X}_d - \dot{X}) + K_d(X_d - X)\right]$$

$$+C(q,\dot{q}) + G(q) - J^T(q)\left[M_x(q)M_d^{-1} - I\right]F_e \tag{3.7}$$

If the desired inertia is chosen equal to the natural Cartesian inertia of the manipulator, (i.e., $M_d = M_x(q)$), the impedance control law in (3.7) can be simplified as,

$$\begin{aligned}\tau = J^T(q)M_x(q)\left[\ddot{X}_d - \dot{J}\dot{q}\right] + J^T(q)\left[B_d(\dot{X}_d - \dot{X}) + K_d(X_d - X)\right] \\ +C(q,\dot{q}) + G(q)\end{aligned} \tag{3.8}$$

Assuming that $\ddot{X}_d = 0$ and also $\dot{J} \approx 0$, the control law becomes

$$\tau = J^T(q)\left[B_d(\dot{X}_d - \dot{X}) + K_d(X_d - X)\right] + C(q,\dot{q}) + G(q) \tag{3.9}$$

This is known as first-order impedance control or inverse dynamics based Cartesian PD (Proportional-Derivative) control and is shown in Figure 3.4. The resulting closed-loop system becomes,

$$\ddot{q} = M^{-1}(q)\left[\tau + J^T(q)F_e - C(q,\dot{q}) - G(q)\right] \tag{3.10}$$

$$\ddot{X} = M_x^{-1}(q)\left[-K_d e_x - B_d \dot{e}_x + F_e\right] \tag{3.11}$$

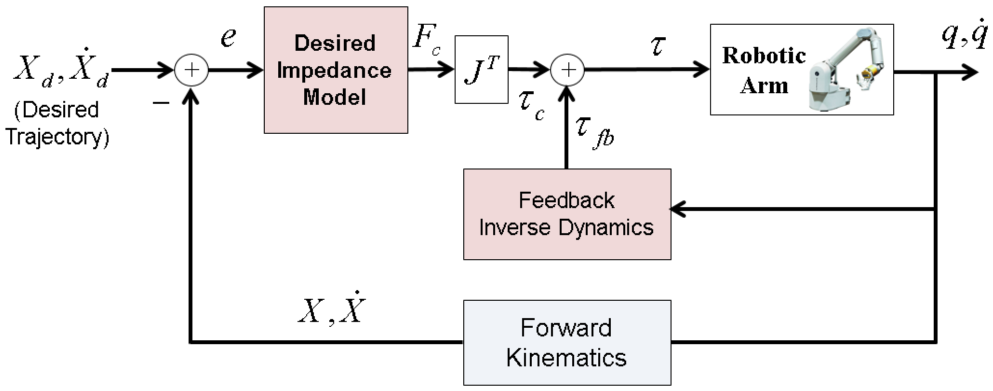

Figure 3.4 First-order impedance control.

An impedance-controlled manipulator effectively behaves as a linear mass-spring-damper system with the given (desired) parameters. This is shown in Figure 3.5 where the environment is modeled as a linear system of impedance $Z_e(s)$ having the equation

$$F_e(s) = Z_e(s)\dot{X}(s) \tag{3.12}$$

where $Z_e(s) = b_e + k_e / s$ is the environment impedance model, $F_e(s)$ is the force exerted on the environment, and $\dot{X}(s)$ is the velocity of manipulator at the contact point, all expressed in the Laplace (or frequency) domain. The environment undergoes a small deformation during contact.

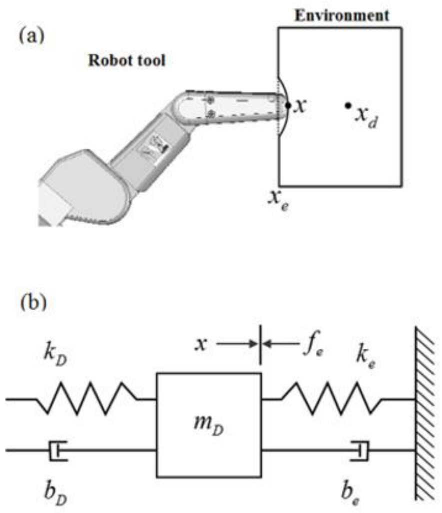

Figure 3.5 Impedance-controlled manipulator (a spring-mass-damper system) interacting with an environment (a spring-damper system).

At the equilibrium point (considering only 1-DOF for analysis purposes) we get

$$0 = -k_d(x - x_d) + f_e \tag{3.13}$$

For an environment located at x_e and modeled with lumped stiffness k_e, we can express

$$f_e = k_e(x_e - x), \quad x \geq x_e \tag{3.14}$$

This leads to the following expression for describing the coupled dynamic behavior between the robot and the environment:

$$x = (k_d + k_e)^{-1}(k_d x_d + k_e x_e) \tag{3.15}$$

$$x = k_{eq}\left(\frac{x_d}{k_e} + \frac{x_e}{k_d}\right) \tag{3.16}$$

The equivalent stiffness is $k_{eq} = k_d k_e / (k_d + k_e)$ (two springs in series). From (3.15) it is clear that in cases where $k_d > k_e$ (i.e., the manipulator's tool is stiffer than the environment), the tool will penetrate into the environment and will be able to reach its desired position $x \approx x_d$ (see Figure 3.5(a)). Similarly, we have $x \approx x_e$ in the case of a stiffer environment (i.e., $k_e > k_d$).

3.4 First-order Impedance Control based on Learned Inverse Dynamics

A major challenge faced by impedance control is that it is based on the computed-torque method. Computed-torque control makes use of the rigid-body dynamic model of the manipulator to cancel out the effects of the nonlinear inertial, Coriolis, centripetal and gravitational forces resulting from the motion of the manipulator. In practice, this computed torque (which is eventually a control action) includes two terms: one from the control law (such as the first- or second-order impedance controller), and the other from computation of the inverse dynamic model. The latter is aimed at canceling out the nonlinear dynamics of manipulator so that the desired level of compliance and impedance behavior is achieved. It is the computation of this second torque term that presents a bottleneck in the implementation of any impedance control scheme. First, it requires accurate knowledge of the robot dynamic model, which is usually partly or completely unknown. Secondly, its online computation is an extremely computationally intensive task, which cannot be accommodated online within the control loop of the manipulator that runs in hard real-time.

This problem can be solved by combining the strengths of feed forward control with incremental online learning of the inverse dynamic model. No a-priori knowledge about the system dynamics is assumed. For this purpose, a fast algorithm, LWPR is used. The torque that is computed based on the learned inverse-dynamic model is fed forward to the inner torque loop of the impedance controller. The proposed impedance control system that is based on learned inverse dynamics is shown in Figure 3.6.

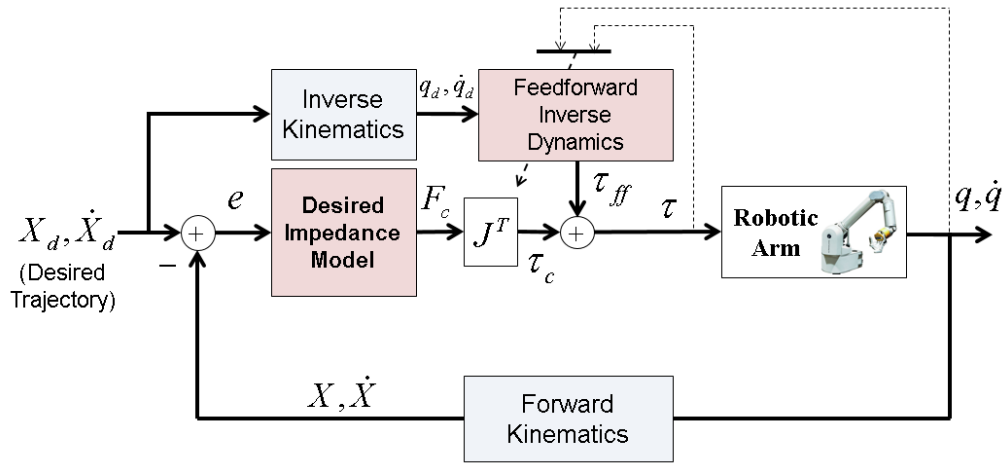

Figure 3.6 First-order impedance control based on learned inverse dynamics.

3.5 Admittance Control

Admittance control in the literature is also called position-based impedance control. Here the desired model is of admittance type and outputs motion to the inner computed-torque controller based on the input force and the desired motion. In this section, the admittance control law will be derived.

The desired impedance model (Equation 3.2) is now re-written as

$$M_d \ddot{X}_a + B_d \dot{X}_a + K_d X_a = F_e \tag{3.17}$$

$$\ddot{X}_a = M_d^{-1} \left[F_e - B_d \dot{X}_a - K_d (X_a \right] \tag{3.18}$$

where X_a is the *adjustment* in position that must be added to the desired (input) position X_d to obtain a new position X_r. This adjusted position is used as a *reference* position for the manipulator to track.

In case an inverse-dynamic-based PD control is used for the motion control, then the modified form of Equation 3.8 becomes

$$\tau = J^T(q)M_x(q)\left[\ddot{X}_r - \dot{J}\dot{q}\right] + J^T(q)\left[B(\dot{X}_r - \dot{X}) + K(X_r - X)\right] + n(q,\dot{q}) \quad (3.19)$$

This is shown in the block diagram form in Figure 3.7.

A simplified version of the admittance model (Equation 3.17) is obtained when only the desired damping B_d is specified as:

$$B_d\dot{X}_a = F_e \quad (3.20)$$

$$\dot{X}_a = B_d^{-1}F_e \quad (3.21)$$

Assuming that a desired velocity $\dot{x}_d$ is directly provided, then $\dot{X}_a$ can be used as an adjustment to obtain a reference velocity for the manipulator to track; i.e.,

$$\dot{X}_r = \dot{X}_d + \dot{X}_a \quad (3.22)$$

where the subscripts $_r,_d$ and $_a$ stand for reference, desired, and actual (end effector velocity). The reference velocity $\dot{X}_r$ can then be integrated to get the reference position X_r as

$$X_r = X_d + \int \dot{X}_r(\tau)d\tau \quad (3.23)$$

The derived reference position and velocity can then be used in the control law in Equation 3.19 which after the assumptions $\ddot{X}_r = 0$ and $\dot{J}\dot{q} \approx 0$ reduces to

$$\tau = J^T(q)\left[B(\dot{X}_r - \dot{X}) + K(X_r - X)\right] + n(q,\dot{q}) \quad (3.24)$$

The $K_a = B_d^{-1}$ in Equation 3.21 is actually a mechanical admittance (velocity/force) which is why this scheme is called *admittance* control. The desired admittance K_a is chosen according to the dynamics of the task that is being carried out. The motion control (inner loop) gains B and K in Equation 3.24 can now be chosen independent of the outer admittance control loop to overcome the system's unwanted nonlinear dynamics. This was not possible with the other two approaches (given in Sections 3.2 and 3.3). As can be noticed, the control law (Equation 3.24) is similar to that in Equation 3.9 except that in the former case the reference Cartesian trajectory is obtained from the admittance model in Equation 3.18.

In the absence of interaction, when $F_e = 0$, admittance control reduces to pure motion control and the control law (3.19) is used with the desired motion as a reference without making any adjustment to it.

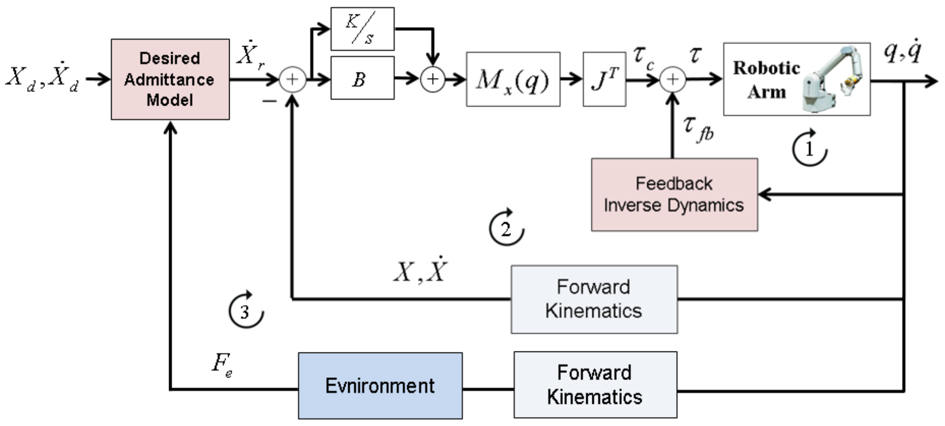

Figure 3.7 Admittance control.

Admittance control is also referred to as *position-based* impedance control in the literature because of the cascaded inner feedback loop (loop 2 in Figure 3.7) for position control of the manipulator. The main advantage of admittance control as compared to standard impedance control is that the inner and outer sub-systems can be designed independently. This makes the control design process easy. The inner loop is designed such that it can compensate for nonlinear dynamics of the robot (based on its inverse dynamic model) and also attenuate the effect of disturbances (unmodeled dynamics) in the output. The outer loop is designed such that the desired dynamic behavior is met without having to deal with the nonlinearities. Standard impedance control will have to achieve all these goals with a single set of control gains, which is not always possible. When the arm is controlled such that it achieves the desired level of compliance, motion tracking is compromised and vice versa. Admittance control solves this problem.

A requirement of implementing admittance control is that the inner loop (loop 2) must run faster than the outer loop (loop 3). This poses a constraint on the dynamics of the system. As the interaction force is fed back into the outer loop, a slower loop rate would mean that the force bandwidth is limited. Interaction with hard and stiff environments, which may contain frequencies up to 1000Hz or higher, therefore, would not be possible and can result in instability. In the context of haptic teleoperation this would mean that the stiff spring-like environment on the slave side may destabilize the control loop. On the master side, the human operator must not tense his arm while manipulating the haptic interface. Assumptions are therefore put on the environment and the human operator to be passive in order for the complete system to be stable. This is a common assumption in the haptic teleoperation literature.

3.6 Simulation Study: Interaction with Soft Environment

Simulation setup for demonstrating the effectiveness of all the impedance control schemes is shown in Figure 3.8. To make the simulation more realistic, actual haptic device (PHANToM) is used as the master manipulator. The slave manipulator and the environment are simulated as 1-DOF systems. The manipulator is simulated as a mass-damper system with parameters $ms = 0.3\text{kg}, \; bs = 1.5\text{Ns/m}$. The environment is simulated as a virtual spring for which the interaction force is calculated as $F_e = -k_e(x - x_e)$ where $k_e = 100$ N/m and $x_e = 0.075$ m (nominal location of the environment, which lies along the *x*-direction of the master's end effector). Cartesian positions of the master and slave systems are graphically displayed on computer screens.

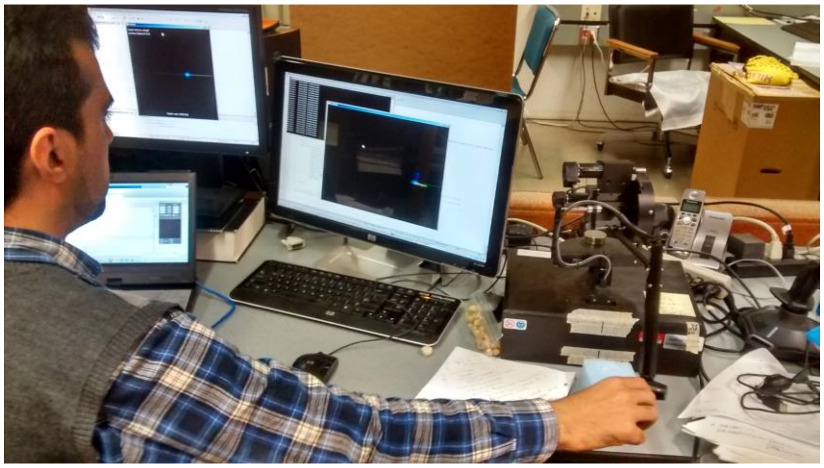

Figure 3.8 Experimental setup (haptic teleoperation with time delay).

Standard Impedance Control

The desired impedance model has the parameter values $M_d = 0.003$kg, $B_d = 0.5$Ns/m and $K_d = 100$ N/m. The results are shown in Figures 3.9 to 3.11. Figure 3.9 shows the tracking of the desired position (as dotted red) as received from the master side. Figure 3.10 shows velocity tracking and Figure 3.11 shows the interaction force.

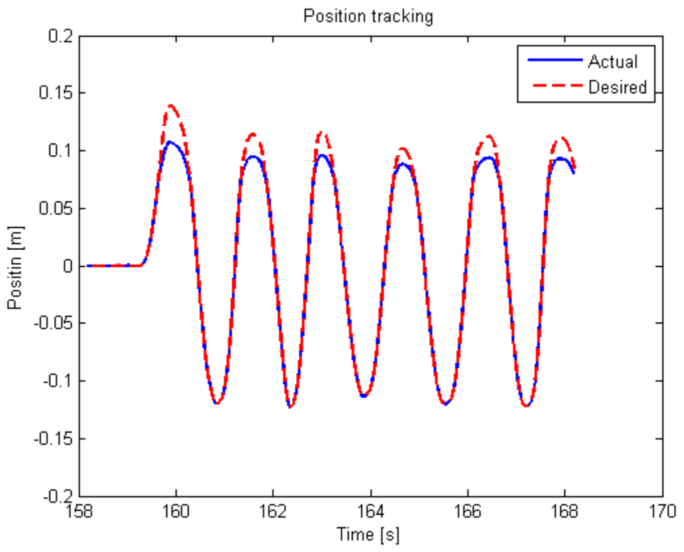

Figure 3.9 Position tracking (Standard impedance control).

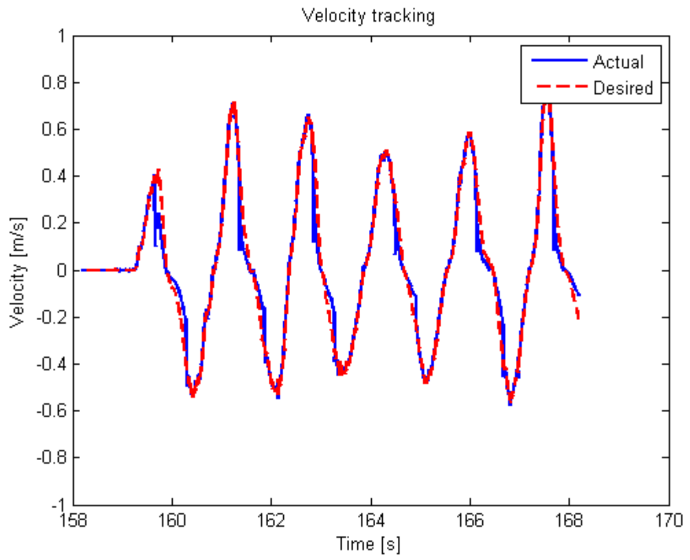

Figure 3.10 Velocity tracking (Standard impedance control).

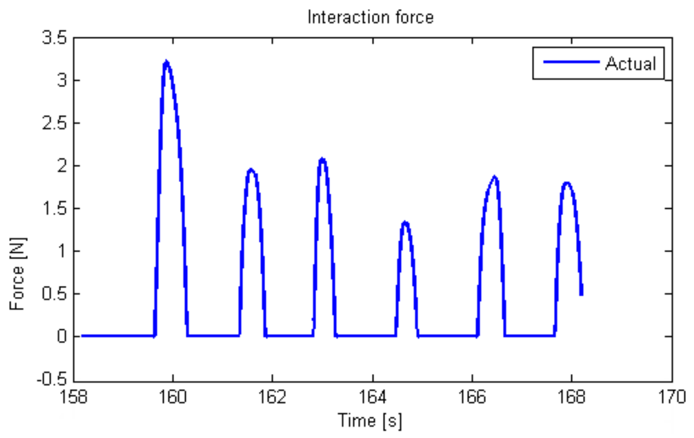

Figure 3.11 Interaction force (Standard impedance control).

First-order Impedance Control

The desired impedance model has the parameters values $K_d = 100\text{N/m}$ and $B_d = 10\text{Ns/m}$. Results are shown in Figures 3.12 to 3.14.

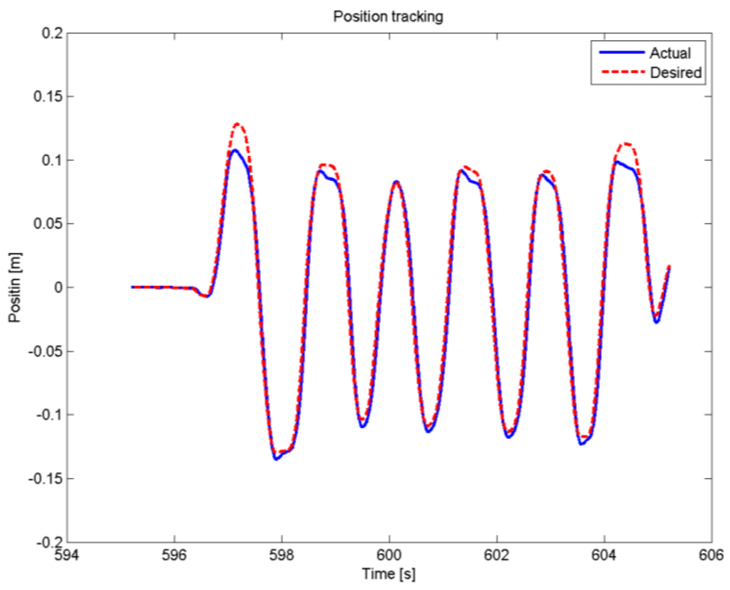

Figure 3.12 Position tracking (First-order impedance control).

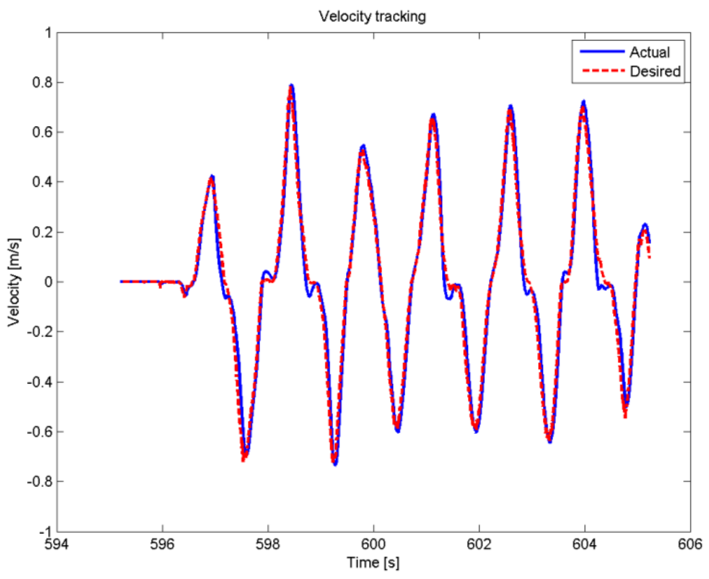

Figure 3.13 Velocity tracking (First-order impedance control).

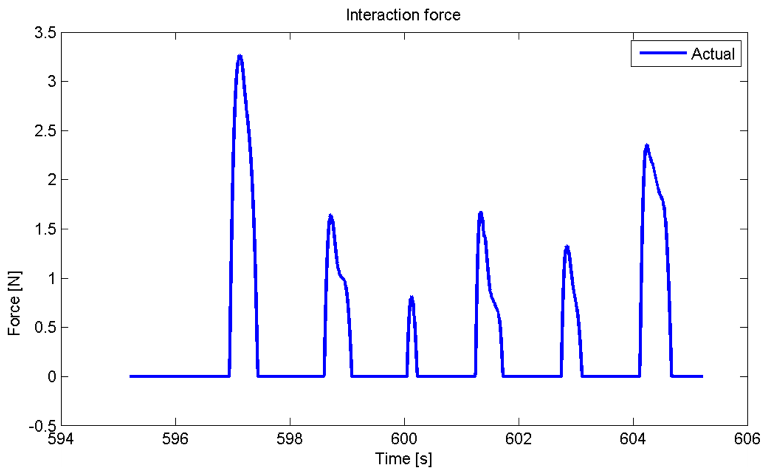

Figure 3.14 Interaction force (First-order impedance control).

Admittance Control

The desired admittance has been set to $K_a = 0.1\text{ms/N}$. The motion control parameter values are $K = 100\text{N/m}$ and $B = 10\text{Ns/m}$. Results are shown in Figures 3.15 to 3.17 for a case where position-error correction is not made (see Equation 3.23). As can be seen in Figure 3.15, position tracking is poor corresponding to the velocity tracking shown in Figure 3.16, which is rather accurate. As a result, interaction with the environment is limited (small magnitude forces in Figure 3.17). The drift in position is due to the fact that it has been obtained by integrating velocity and the initial condition is not set properly. Increasing the admittance gain (setting it equal to unity) further aggravates the problem by increasing the position drift. This was solved by also adjusting the reference position X_r by adding the desired position X_d to it (as in Equation 3.23). This adjustment of position actually matches the initial conditions of the Integrator to the actual desired position. This means that in the case of teleoperation, the desired positions must also be exchanged along with velocities between the master and slave manipulators. This will be discussed in detail in Chapter 4. Improved results are shown in Figures 3.18 to 3.20.

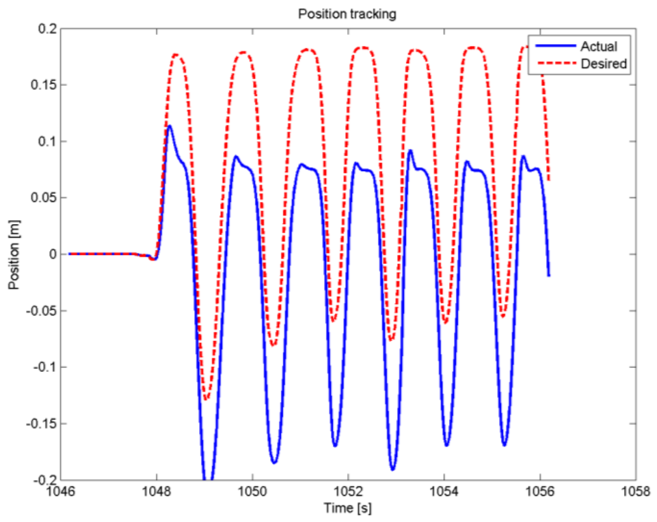

Figure 3.15 Poor position tracking (Admittance control).

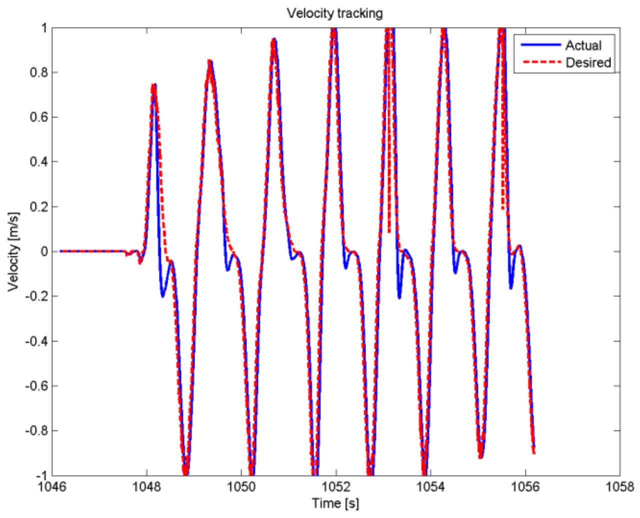

Figure 3.16 Velocity tracking in case of poor position tracking (Admittance control).

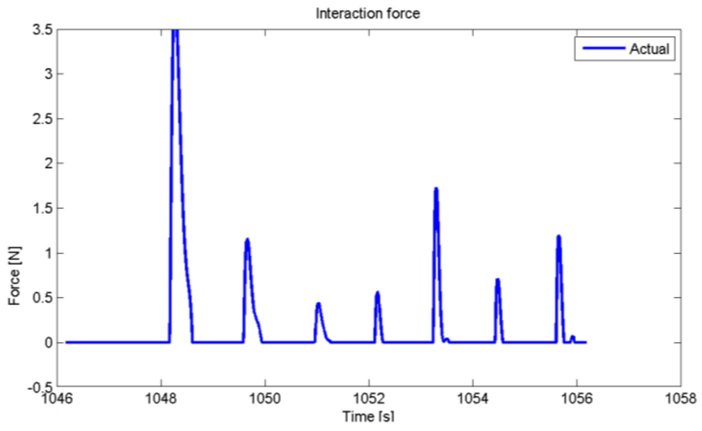

Figure 3.17 Interaction force in case of poor position tracking (Admittance control).

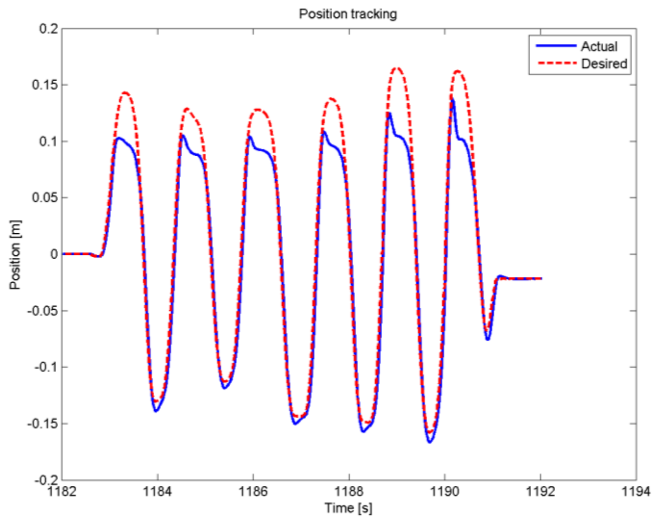

Figure 3.18 Position tracking (Admittance control).

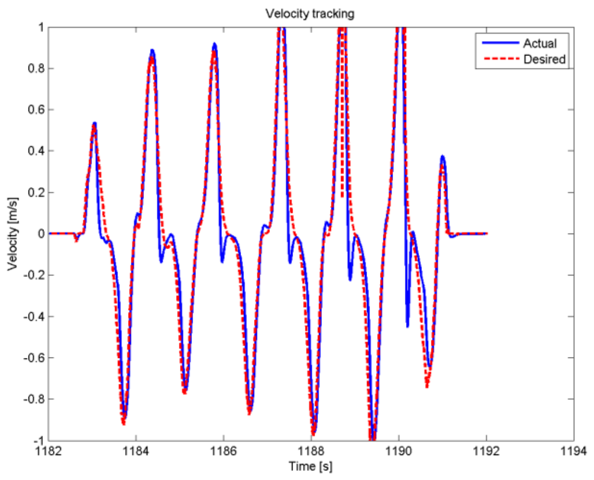

Figure 3.19 Velocity tracking (Admittance control).

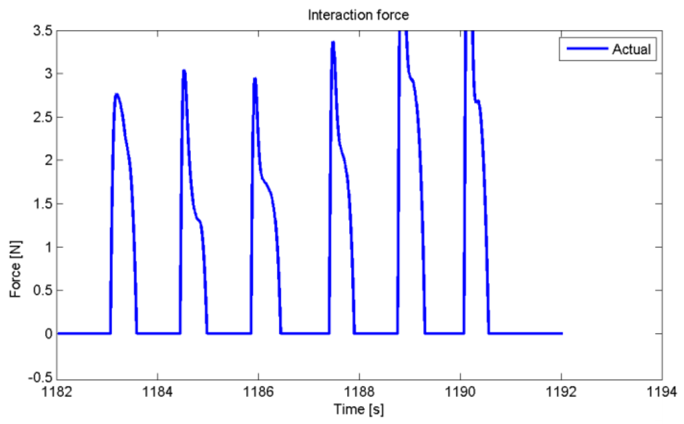

Figure 3.20 Interaction force (Admittance control).

3.7 Experiment 1: Control of PHANToM Interacting with a Hard Environment

This experiment is designed to demonstrate the strengths of each control scheme presented in this chapter while interacting with a stiff environment. This is important in order to assess the safety and stability of each control scheme for practical application in a homecare environment. Experimental setup is shown in Figure 3.21 where the actual master and slave manipulators are used (both PHANToM devices). The environment is simulated as in the previous simulation, but with larger stiffness value $K_e = 800$N/m located at $x_e = 0.065$ m. The haptic controllers are Windows-based Intel x86 PCs with real-time scheduling ensured by the CHAI 3D library with a servo rate of 1,000 Hz. Communication between the master and slave systems occurs on a Local Area Network (LAN) network. For the current results, the round-trip time delay is less than 1ms and is therefore neglected. The desired apparent inertia of the slave PHANToM is set equal to the natural inertia. The nonlinear dynamics were low and therefore neglected. The results of the first-order impedance control are shown in Figure 3.23. The desired stiffness and damping were selected as $k_P = 200$ N/m, and $b_D = 1$Ns/m. The desired stiffness is selected such that the manipulator should feel compliant (soft) while interacting with a hard environment. But, as indicated from the results, this extra level of compliance comes at the cost of loosing the position accuracy. This sluggish behavior is inherent in first-order impedance control and can be considered as its weakness. This problem is solved by admittance control as can be seen in Figure 3.24 which shows a stable interaction. As seen in the velocity plot, the end effector slows down while making contact with the hard environment. As position tracking is achieved by the inner loop, the impedance parameters in the outer loop can be independently selected. For robotic surgery and homecare applications, the desired damping term must be selected high in order to dampen out fast oscillatory motions that may arise during making contact with a stiff environment. This work is published in [63].

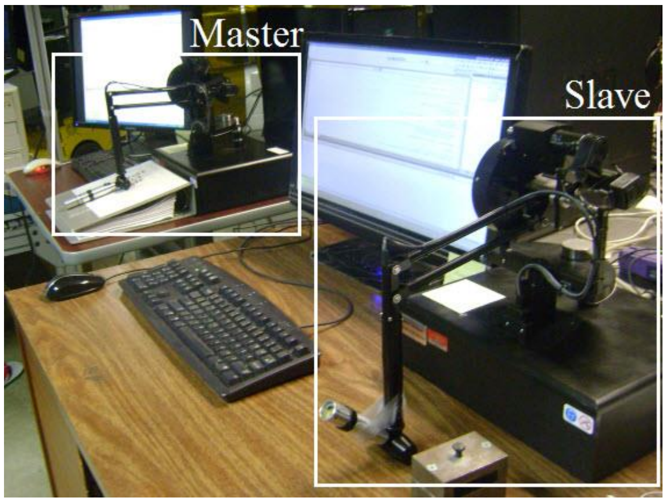

Figure 3.21 PHANToM devices as the master and slave manipulators with their corresponding controllers (PCs).

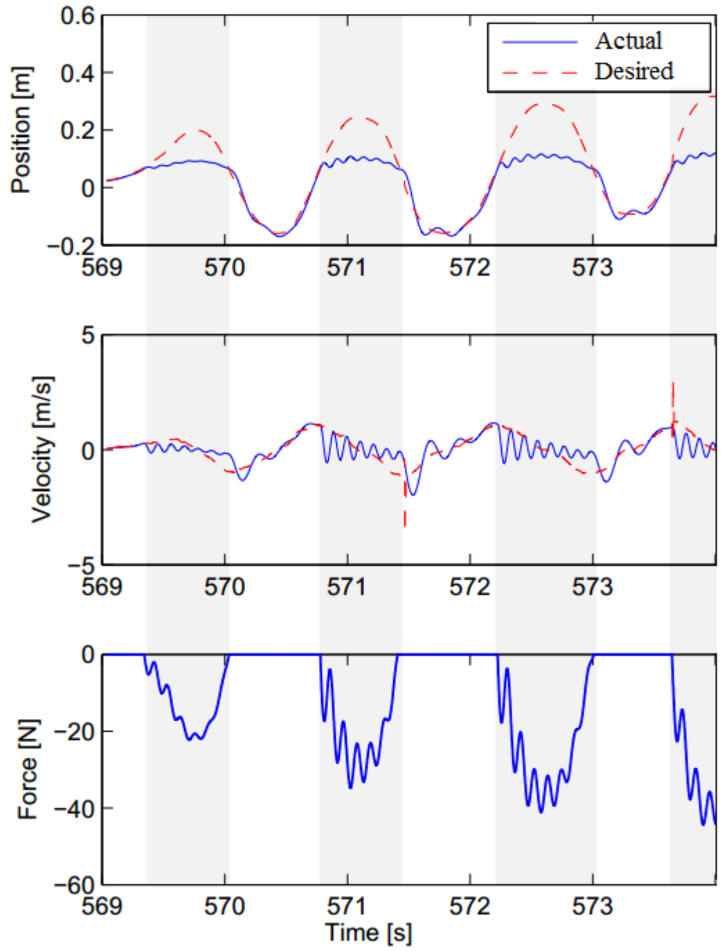

Figure 3.22 First-order impedance control (Interaction with hard environment).

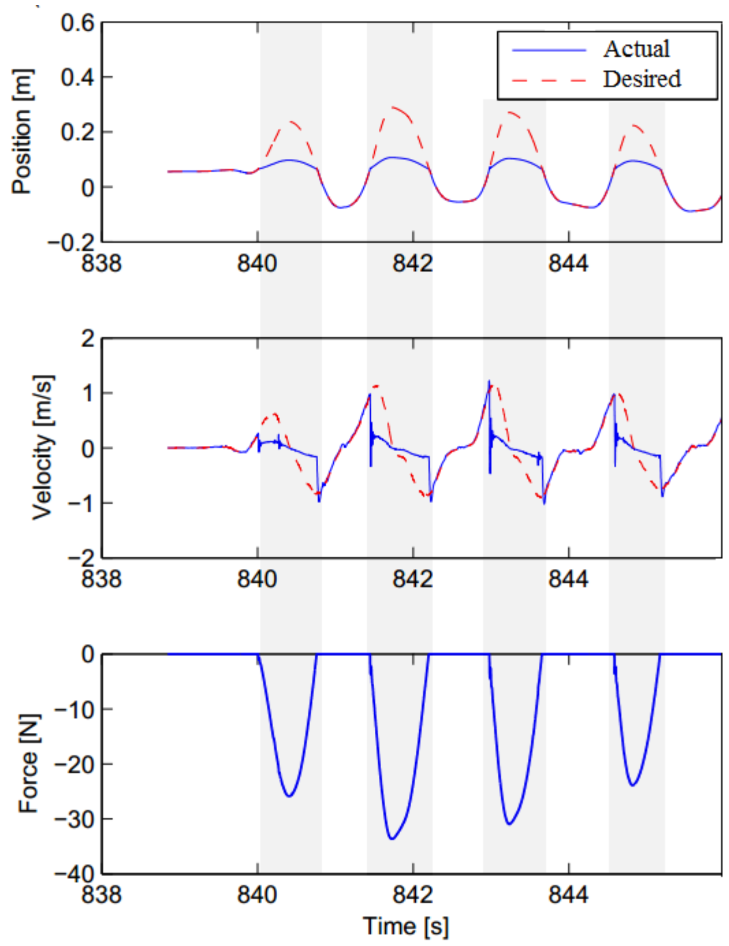

Figure 3.23 Admittance control (Interaction with hard environment).

3.8 Experiment 2: First-order Impedance Control of WAM Arm

In this final experiment, the first-order impedance control was implemented on the WAM arm whose nonlinear dynamics such as friction and Coriolis cannot be ignored. The WAM arm is instructed to follow the same circular trajectory as in the example simulation in Chapter 2 of this thesis. Interaction with any environment could not be performed due to the unavailability for force/torque sensor. So the first-order impedance controller is practically equivalent to Cartesian PD controller with inverse dynamic compensation. The experiment still served to assess the performance of the learned inverse dynamic model based on the LWPR algorithm. For comparison the same trajectory was tracked using the libbarrett's inverse dynamic system (`systems::InverseDynamics`) which uses the Newton-Euler formulation to compute the inverse dynamic model using the dynamic parameters provided by the manufacturer. Results are shown in Figures 3.24 to 3.29. For the case of LWPR, Figure 3.24 shows the end effector position in *x*-*y* plane during tracking. Figure 3.25 shows the feed forward torques as computed by the LWPR model based on the desired trajectory (in the joint-space which was computed from the desired Cartesian trajectory using inverse kinematics; see Figure 3.6). Figure 3.26 shows the total torque sent to the actuator. This is the sum of the feed forward torque and the torque from the controller. Similar plots for the libbarrett inverse dynamic model are shown in Figures 3.27 to 3.29. Results show that the tracking performance of the LWPR model is comparable to that of the analytical model.

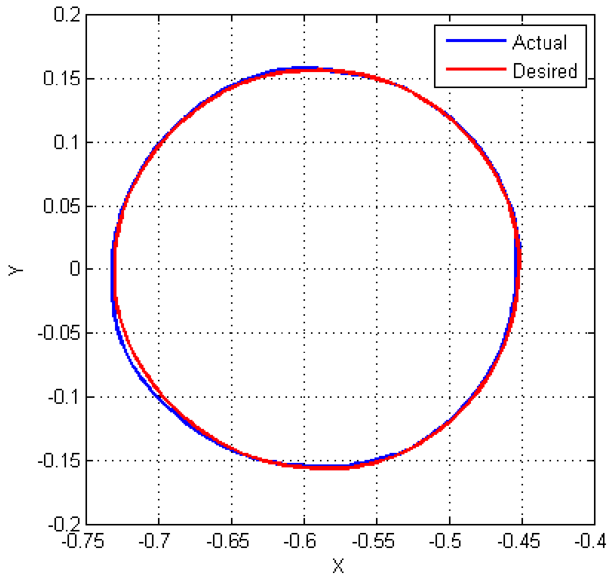

Figure 3.24 Tracking of a desired Cartesian trajectory based on LWPR-generated model.

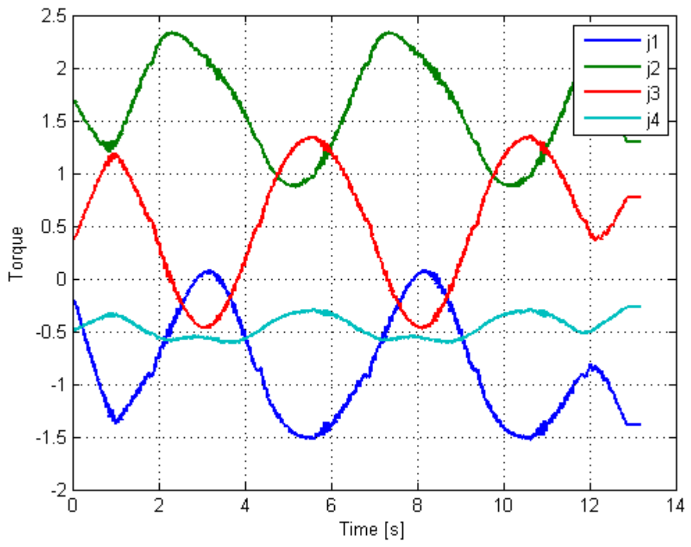

Figure 3.24 Output of the inverse dynamic model (tracking based on LWPR model).

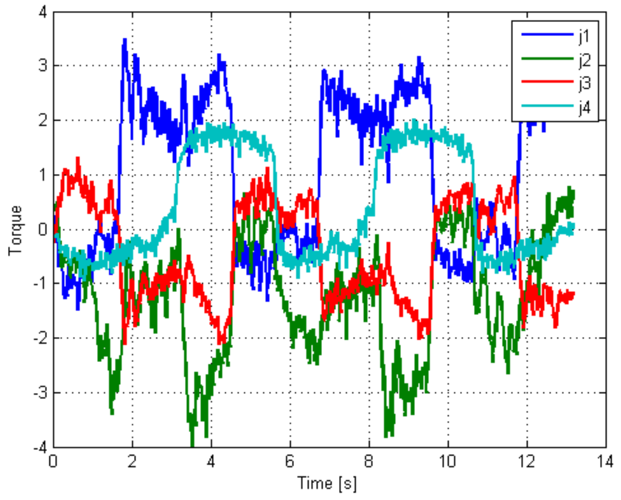

Figure 3.25 Controller output (tracking based on LWPR model). Gains $k_p = 2000, b_d = 20$.

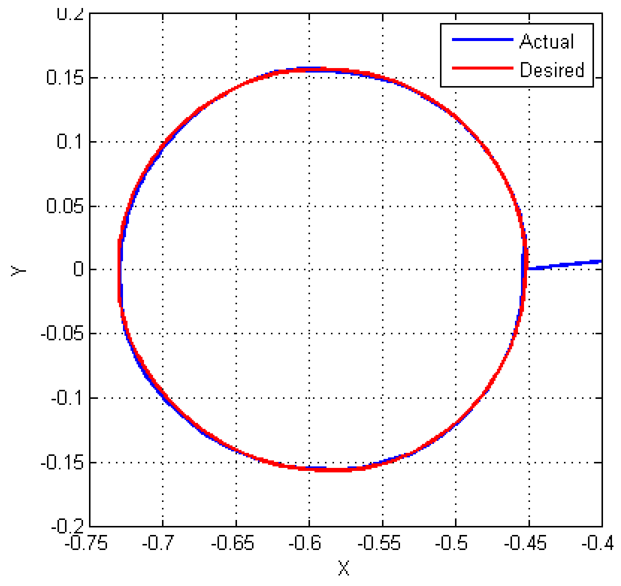

Figure 3.26 Tracking of a desired Cartesian trajectory based on libbarrett's inverse dynamics model.

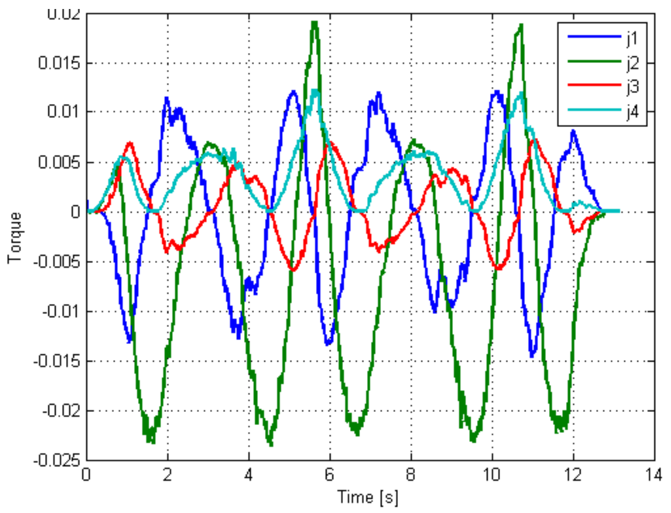

Figure 3.27 Output of the inverse dynamic model (tracking based on libbarrett's inverse dynamics model).

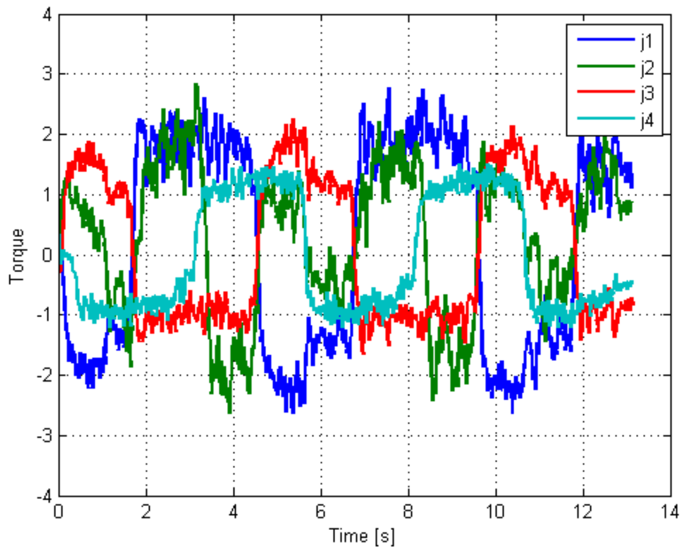

Figure 3.28 Controller output (tracking based on libbarrett's inverse dynamics model). Gains $k_p = 2000, b_d = 20$.

3.9 Estimation of the Environment Dynamic Model

During interaction control, dynamics of the environment (the contact object) couple with the dynamics of the manipulator. The environment's behavior affects the control of the manipulator. In order for the impedance control to be accurate, knowledge about the dynamic models of the manipulator and the environment both are required. In Section 2.4 enhanced method for computing the inverse dynamic model of the manipulator using a data-driven approach has been presented. The dynamics of the environment, however, are usually completely or partly unknown. The estimation technique of the environmental dynamic model depends on the type of environment that the manipulator is interacting with. The focus of the current work is on the type of environment that is stationary and has a rigid or compliant surface.

Several methods have been investigated to determine the unknown impedance model of the environment (e.g., see in [64], [65], and [66]) and selecting the corresponding controller gains according to the estimated model parameters. A practical and recommended method uses the online identification of the environmental model and adaptive tuning of the controller impedance gains. In most cases the environmental dynamic model is assumed as a first- or second-order linear system. Once the model structure is assumed, the model parameters can be estimated using least-square recursive regression techniques. These techniques have the capability to deal with time-varying systems and therefore form the basis of many of the adaptive control strategies. They also can be initially trained offline (in an ideal laboratory environment) and then iteratively updated online. Lastly, they are computationally efficient, less memory intensive, and therefore can be implemented in real time.

Generally, the dynamic behavior of the environment can be approximated by a lumped stiffness k_e and a lumped damping coefficient b_e. Using measurement data such as information from the force, position and velocity sensors, the unknown environmental parameters: the coefficients b_e and k_e can be estimated online. Once the experimental data are available, a recursive least-squares estimator is used to estimate the impedance parameters. This is illustrated in Figure 3.30. The estimation process is executed at discrete samples in real-time where the current state at time instant i is predicted from the previous state (at $i-1$). The predicted state is then combined with the current measurements to estimate the current state. Details of the RLSR Algorithm are presented next.

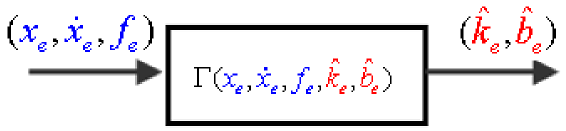

Figure 3.29 RLSR algorithm for environment model identification.

The environment can be modeled as a mass-spring-damper system with lumped damping coefficient b_e, and lumped stiffness constant k_e:

$$f_e = b_e \delta \dot{x} + k_e \delta x \tag{3.25}$$

Here $\delta x = x - x_e$, and x_e is the location of the environment. In the discrete (z-transform) domain, the system (Equation 3.25) can be represented as

$$f_e = \left[b_e \left(\frac{2}{T_s} \right) \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right) + k_e \right] \delta x \tag{3.26}$$

where T_s is the sampling period and z^{-1} is the time-shift operator (or, the z-transform variable). The resulting difference equation is:

$$f_e(i) + f_e(i-1) = a_1 \delta x(i) + a_2 \delta x(i-1) \tag{3.27}$$

where i represents the time-step index. The coefficients a_1 and a_2 are the model parameters that need to be estimated. The system constants b_e and k_e are then computed from these parameters.

The position $x(t)$ is treated as the input to the system and the force $f_e(t)$ as the output. This means the environment is treated as an impedance-type, although the algorithm presented here is applicable to admittance-type models as well.

The system in Equation 3.27 can be represented as a linear regression

$$y(i) = \phi^T(i)\theta \tag{3.28}$$

where $\theta = [a_1, a_2]^T$ is parameter estimate vector and $\phi(t) = \left[\delta x(i), \delta x(i-1) \right]^T$ is the regression vector. Variable $y(i)$ is the system output, which in the current case equals $f_e(i) + f_e(i-1)$. The model in (3.28) belongs to a family of more general time-domain input-output models referred to as ARMAX which are usually used for system identification purposes. Their application is not limited to robotics only.

Considering that the regression matrix $\Phi(n)$ is formed from the regression vector $\phi(i)$ as $\Phi(n) = \left[\phi^T(1), \phi^T(2), \ldots, \phi^T(n)\right]^T$ and the output vector as $y(n) = \left[y(1), y(2), \ldots, y(n)\right]^T$, the normal least-squares estimate of Equation 3.28 can be found as:

$$\hat{\theta}(n) = \left[\Phi(n)^T \Phi(n)\right]^{-1} \Phi(n)^T y(n) \tag{3.29}$$

where $\hat{\theta}(n)$ represents an estimate of the output parameters at time $t = n$. As the ARMAX system has its inputs and outputs that rely on their current and previous values, it lends itself to recursion and therefore identification in real time.

A complete RLSR algorithm can be obtained in two steps as outlined below [67]:

Prediction step:

$$\hat{\theta}(i) = \hat{\theta}(i-1) + K(i)\left[y(i) - \varphi(i)\hat{\theta}^T(i-1)\right] \tag{3.30}$$

Update step:

$$P(i) = \frac{1}{\lambda}\left[P(i-1) - \frac{P(i-1)\varphi(i)\varphi^T(i)P(i-1)}{\lambda + \varphi^T(i)P(i-1)\varphi(i)}\right] \tag{3.31}$$

$$K(i) = \frac{P(i-1)\varphi(i)}{\lambda + \varphi(i)P(i-1)\varphi(i)} \tag{3.32}$$

where P is the error covariance matrix (P being a large positive number) and $\lambda < 1$ is the forgetting factor, which determines that the measurements that are τ samples old will be forgotten and not used in the current computation. $\tau = \frac{1}{1-\lambda}$ is called the *memory horizon* of the algorithm. The gain, $K(i)$, determines the effect of the current prediction error ($y(i)-\varphi(i)\hat{\theta}^T(i-1)$) on the parameter estimate.

The RLSR algorithm with a forgetting factor can estimate the time-varying environmental parameters in real time provided that the system dynamics change slowly with time.

3.9.1 Simulation Study

An example simulation is presented here to assess the strengths and shortcomings of the system identification algorithm that has been presented. The assumed model structure whose parameters are to be identified is a second-order linear system (a mass-spring-damper) having natural frequency $\omega_n = 37.4$ rads^{-1} and damping ratio $\zeta = 0.53$. This corresponds to actual stiffness $k_e = 1400$ Nm^{-1} and damping coefficient $b_e = 40$ Nsm^{-1}, which are the target values of the parameters for the identification process. It is assumed here that the available measurements (sensor identification data) include the exogenous input to the system position x, and the output interaction force f_e. An input signal consisting of two sinusoidal functions of different frequencies, as shown in Figure 3.31, is used:

$$x(t) = \sin(2\pi f_1 t) + 0.5\sin(2\pi f_2 t) \tag{3.33}$$

The input signal has sufficiently rich frequency content (in the given context of teleoperation) to serve as a persistent excitation. The frequencies considered here are 10 Hz and 20 Hz. From a practical viewpoint, these frequencies are of the order in magnitude as the human arm movement (the source of input motion) while he/she is manipulating the master robot in a real-time teleoperation application. The sampling time is taken as $T_s = 0.001$ s. The white noise of standard deviation 0.01 is also added to the input. The results of the simulation are presented in Figures 3.32 to 3.34 which show the convergence of the algorithm to the target values.

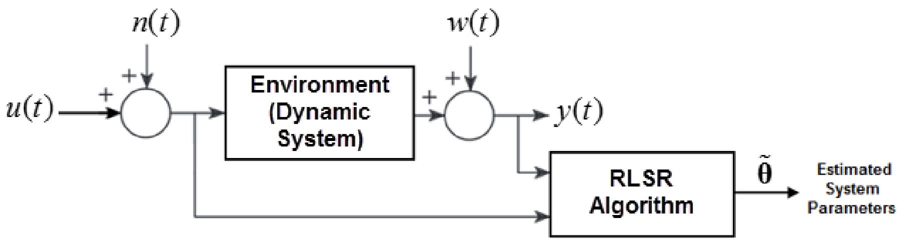

Figure 3.30 Simulation setup for example simulation of online identification of the environment model.

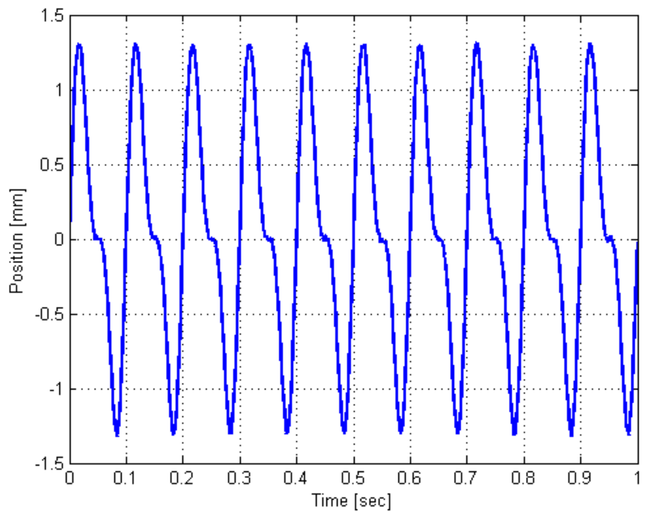

Figure 3.31 Input signal (example simulation of online identification of the environment model)

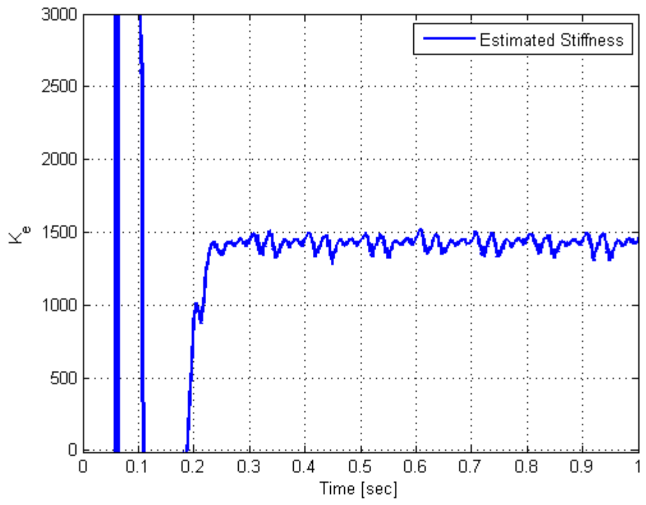

Figure 3.32 Stiffness estimate (example simulation of online identification of the environment model)

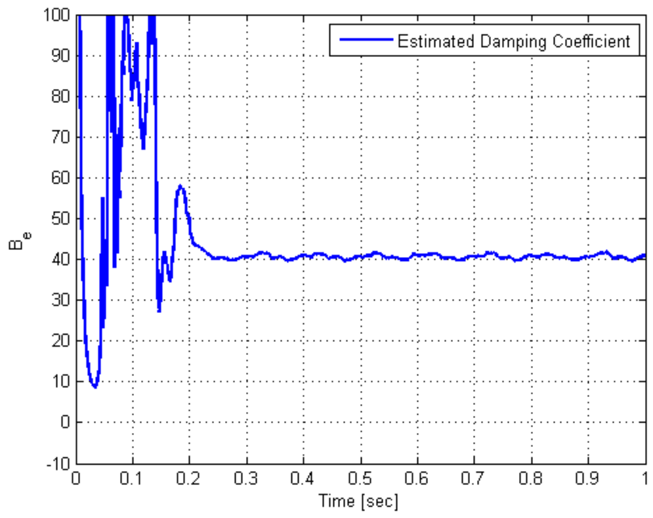

Figure 3.33 Damping estimate (example simulation of online identification of the environment model).

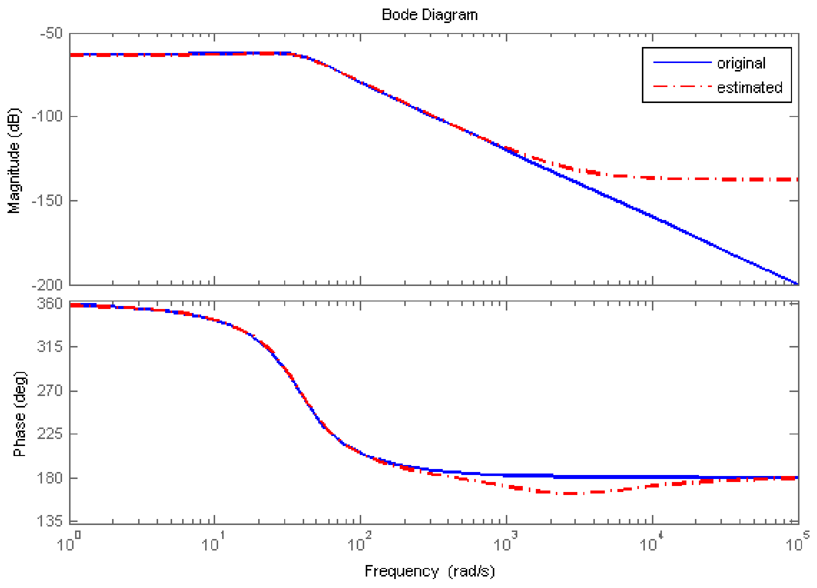

Figure 3.34 Bode plot of actual and estimated systems (example simulation of online identification of the environment model).

3.10 Selection of Desired Impedance Model

Once the environmental impedance is estimated online, the desired impedance parameters for impedance control of the slave manipulator can be selected according to the identified parameters. This section provides some guidelines that can help tune the desired parameters.

For an impedance-controlled robot (where $\{m_d, b_d, k_d\}$ are the desired impedance parameters) that is in contact with an environment (modeled as a linear spring with stiffness k_e and damping b_e), the desired natural frequency and the damping ratio along each Cartesian axis are given by

$$\omega_d = \sqrt{\frac{k_d + k_e}{m_d}} \tag{3.34}$$

$$\zeta_d = \frac{b_d}{2\sqrt{m_d(k_d + k_e)}} \tag{3.35}$$

To overcome the contact instability problem, an over-damped behavior ($\zeta_{ds} > 1$) is stipulated. The manipulator end effector may exhibit chattering behavior, especially in the case of contact with a very stiff environment. The contact instability problem can be resolved by selecting the damping term to be larger than the corresponding mass and stiffness terms.

A large damping ratio also means avoiding large impact forces, which may arise during contact with an environment having uncertain geometry. This necessitates a small value for k_d (high compliance) in directions where contact with a stiffer environment ($k_e >> k_d$) is foreseen. Thus the desired stiffness of the slave manipulator matches the environmental stiffness in a complementary way. Once the desired stiffness term is selected based on the estimated environmental stiffness, the desired damping term b_d is used to shape the closed-loop transient behavior. The desired stiffness of the master manipulator should be set equal or proportional to the estimated environmental stiffness to achieve enhanced transparency.

One way to select the desired impedance characteristics is based on qualitative analysis. Criterion formulated as a result of detailed stability analysis does not usually suffice due to the fact that stable impedance control gains may still be practically unsafe. Following are some general guidelines for the selection of desired impedance parameters [68]:

- Desired inertia should be selected relatively high in directions where a contact is expected. This will limit the dynamics of the robot and result in limited contact forces.

- Desired damping is selected to be relatively high in directions where a (critically- or over-) damped response is required. This will lead to dissipating more energy along those directions.
- Desired stiffness is selected to be relatively high in a direction where position control is required and relatively low where contact is expected. High stiffness value in position-controlled directions will also lead to enhanced accuracy in motion tracking by the end effector.

3.11 Summary

This chapter presented three types of impedance controllers. It provided an experimental evaluation and showed the superiority of admittance control over the other two. Impedance control based on learned inverse dynamic model was also proposed. Algorithm and guidelines for selecting parameters of the desired impedance model were given.

Chapter 4: Haptic Teleoperation with Time Delay

4.1 Haptics

The word *haptic* is from the Greek “haptikos" and it means “pertaining to the sense of touch." It is related to the perception of objects through touch. A haptic system (which comprises software and hardware) allows humans to interact with real or virtual environments. Haptic feedback is the haptic information that a haptic interface provides to a human operator. Haptic feedback can be categorized into two different classes: *tactile* and *kinesthetic*. The kinesthetic haptic feedback is felt by the sensory receptors (known more formally as *proprioceptors*) in our muscles, joints and tendons. They excite the human locomotive system, which includes joint positions, limb alignment, body orientation, and muscle tension [69]. We can feel the weight, mass, stiffness, orientation etc. of an object from the sensation of movement or stress in our muscles, tendons, and joints. On the other hand, tactile information is felt by receptors embedded in our skin. It enables us to feel vibration, pressure, pain, touch, texture, temperature, and so on.

Haptic teleoperation using a haptic interface (normally a 6DOF robotic manipulator) helps reproduce to the operator, contact forces that are actually generated at a remote location or in a virtual world (such as within a computer simulation). In the context of haptic teleoperation, the term *force feedback* usually describes the kinesthetic feedback, which is normally provided using physically grounded haptic interfaces. Availability of force feedback to the human operator has resulted in improved task performance. As an example, in [70], the authors have experimentally demonstrated that kinesthetic force feedback would improve the task performance in carrying out a ground-space bilateral teleoperation of a robot arm. Similarly, in [71], the authors have assessed the benefits of force feedback in robotic surgery. They found force feedback helpful in blunt dissection as it helped constrain the surgeon’s hand from commanding inappropriate motions that could generate large forces and damage the surrounding tissues. Reference [69] also shows that the force feedback helps lower the error rate during telemanipulation.

The PHANToM Premium 1.5 model provides force feedback around roll, pitch, and yaw axes in addition to three translations. In this thesis, the haptic feedback is limited to kinesthetic feedback in the three spatial directions (3-DOF). Structurally, it is an *impedance-type* haptic interface because the human operator can mechanically input motion by holding and moving the position of the Tool Centre Point (TCP) into the system. The mechanical output is in the form of a force or torque that the device (with the help of actuators) displays at the stylus location (a 3D point in the workspace) for the operator to feel (see Figure 4.1). As the position is measured and the force is generated, and hence the name "impedance-type." The impedance - type would also mean that the device can be naturally impedance controlled both in open- and closed-loop fashions.

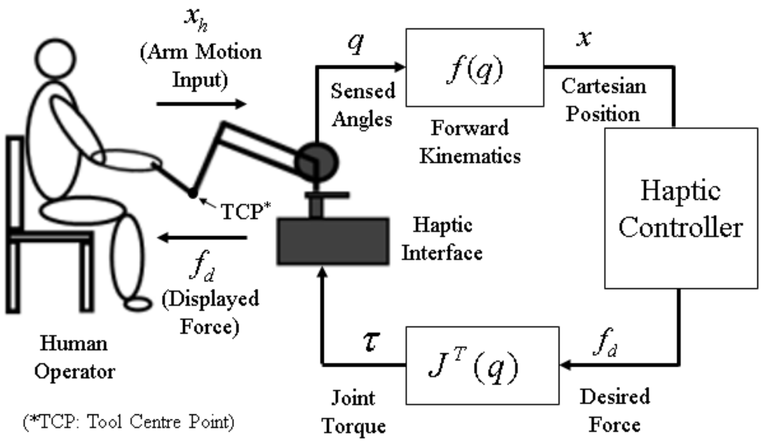

Figure 4.1 PHANToM Premium haptic interface and controller.

4.2 Haptic Teleoperation with Time Delay

Figure 4.2 shows a network representation of a teleoperation system. This way of modeling and representing a robotic teleoperation system was originally inspired by network theory from the field of electrical power systems. The human operator, the Master-Slave-Network (MSN) subsystem, and the environment are modeled as a 1-port, 2-port, and 1-port network, respectively. The operator's hand velocity, V_h, is provided to the MSN system, which returns to him an output force feedback F_h. On the right side of MSN is an environment, which applies velocity V_e as a result of its interaction with the end effector of the slave arm. The direction of V_e is the reverse of V_h. Also, F_e represents the environmental force, which is to be fed back to the human operator. The MSN subsystems comprise the master and slave manipulators along with their controllers. Later, the communication channel between the master and slave ends will also be included.

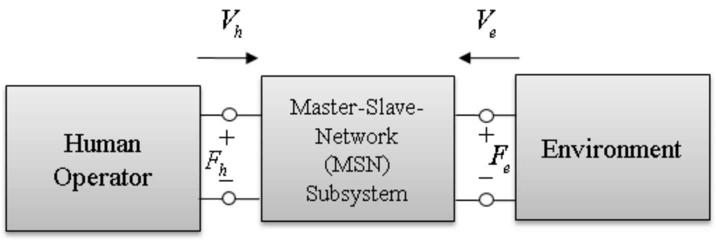

Figure 4.2 Bilateral teleoperation system.

In order to study and analyze the stability and performance of the system, the MSN subsystem is represented by a set of four transfer functions between various inputs and outputs at the interface of MSN with the external world. These transfer functions are arranged in a 2×2 matrix $H(s)$ called *hybrid matrix* [72] and is illustrated in Figure 4.3. As shown, the human operator and the environment are modeled as the impedances Z_h and Z_e, respectively. The negative sign associated with V_e indicates that the actual direction is opposite to what is considered here.

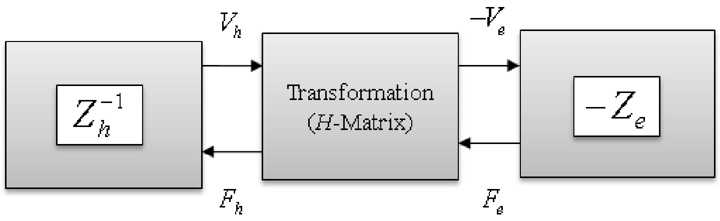

Figure 4.3 Bilateral teleoperation system.

The hybrid matrix $H(s)$ is defined as

$$\begin{bmatrix} F_h(s) \\ -V_e(s) \end{bmatrix} = \begin{bmatrix} h_{11}(s) & h_{12}(s) \\ h_{21}(s) & h_{22}(s) \end{bmatrix} \begin{bmatrix} V_h(s) \\ F_e(s) \end{bmatrix} \tag{4.1}$$

where s is the Laplace variable. The description of all four transfer functions in $H(s)$ is given in Table 4.1.

Table 4.1 Description of the hybrid matrix, *H(s).*

$h_{ij}(s)$	Description
$h_{11}(s) = \left.\frac{F_h(s)}{V_h(s)}\right\rvert_{F_e=0}$	Impedance felt when slave arm is in free motion
$h_{12}(s) = \left.\frac{F_h(s)}{F_e(s)}\right\rvert_{V_h=0}$	Force feedback gain when master arm is not moving
$h_{21}(s) = \left.\frac{-V_e(s)}{V_h(s)}\right\rvert_{F_e=0}$	Forward velocity gain when slave is in free motion

$h_{22}(s) = \left.\frac{-V_e(s)}{F_e(s)}\right\|_{V_h=0}$	Impedance felt by environment when master is not moving

The goal of any teleoperation system design is to achieve an ideal hybrid matrix, H_{ideal}, defined as

$$H_{\text{ideal}} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{4.2}$$

This states that if the slave arm is not interacting with the environment, there should not be any impedance felt by the operator; hence, $h_{11}(s) = 0$. This element characterizes the impedance transmitted to the operator by the MSN when the slave is in free motion. Also, $h_{12}(s) = 1$ states that the desired force must be displayed transparently to the operator; $F_h = F_e$, while he is interacting with the environment and holding the master arm firmly ($V_h = 0$). This element characterizes the force scaling during interaction. Furthermore, $h_{21}(s) = -1$ states that $V_e = V_h$; hence, the slave must perfectly track the desired trajectory that is received from the master side. This element characterizes the slave's motion tracking capability while its not in interaction with the environment. Similarly, the last element $h_{22}(s) = 0$ states that there shouldn't exist any activity (interaction) between the slave arm and the environment if there is no such command received from the master side ($V_h = 0$). This element characterizes the admittance transmitted by the MSN to the environment when the master is clamped.

Various measures of system stability and transparency are then derived based on the hybrid matrix formulation. Other matrix representations of MSN, such as impedance, admittance, and inverse hybrid, also exist in the literature, which are different only with regard to the choice of input-output variables on the left and right hand sides of Equation 4.1. The essence, however remains the same. Reference [73] presents and extends Llwellyn’s absolute stability criterion, which renders the two-port network absolutely stable. The criterion is given in terms of the two-port network impedance parameters. In [74], the authors have extended the criterion to a non-passive operator.

Apart from stability, an important design goal for teleoperation systems is transparency, which is defined as a match between the positions and forces of the master and the slave manipulators. The two elements, $h_{11}(s)$ and $h_{21}(s)$, combined, can be used as a measure of the system’s transparency. In addition to the position and force correspondence, a transparent teleoperation system also aims at ensuring that the *impedance match.* According to this, the impedance presented to the operator by the MSN during interaction, Z_{to}, is the same as that of the environment, $Z_{\mathrm{to}} = Z_e$. It means that the operator feels the same (and undistorted) environmental impedance without feeling any unwanted dynamics of the master and slave manipulators.

The transmitted impedance Z_{to} can be computed as [75],

$$Z_{\mathrm{to}} = \frac{F_h(s)}{V_h(s)} = \frac{h_{11}(s) + \left(h_{11}(s)h_{22}(s) - h_{12}(s)h_{21}(s)\right)Z_e}{1 + h_{22}(s)Z_e} \tag{4.3}$$

For the extreme conditions when $Z_e = 0$, the slave moves freely without touching the environment, and when $Z_e = \infty$, the slave interacts with an extremely stiff environment. The corresponding expressions for Z_{to} are,

$$Z_{\mathrm{to,stiff}} = \frac{h_{11}(s)h_{22}(s) - h_{12}(s)h_{21}(s)}{h_{22}(s)} \tag{4.4}$$

$$Z_{\text{to,free}} = h_{11}(s) \tag{4.5}$$

which are the maximum and minimum values of transmitted impedance Z_{to}. From these values another important performance measures, Z_{width}, can be derived as

$$Z_{\text{width}} = Z_{\text{to,stiff}} - Z_{\text{to,free}} \tag{4.6}$$

As can be noticed, Z_{width} is the range of impedances that the operator can experience during interaction. Two other transparency measures that have been derived in [75] from the $Z_{\text{to,stiff}}$ and $Z_{\text{to,free}}$ are the apparent stiffness and damping of the teleoperation system.

$$k_{\text{to,stiff}} = \lim_{s\to 0} sZ_{\text{to,stiff}} \tag{4.7}$$

$$b_{\text{to,free}} = \lim_{s\to 0} Z_{\text{to,free}} \tag{4.8}$$

4.2.1 Delay Effect on Stability

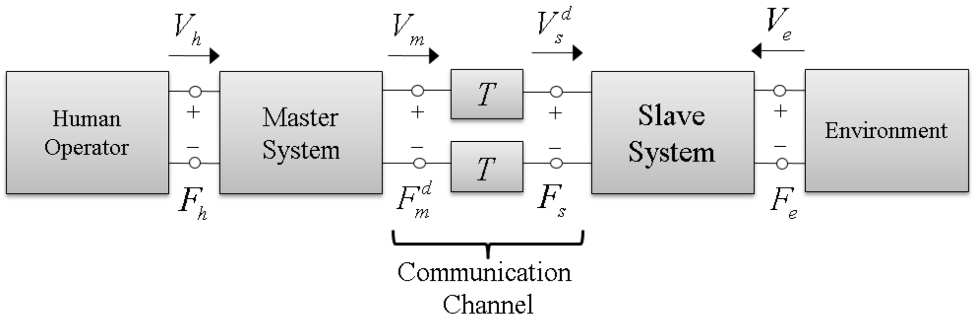

Figure 4.4 Teleoperation system as an interconnection of 2-port network elements.

Figure 4.4 shows an expanded version of Figure 4.2 where the communication channel has now been considered. The major impact that the communication channel has on the performance and stability of the teleoperation system is that it induces a time delay T in the information that is being sent through it. If a symmetric time delay T is assumed, then

$$F_h = F_e(t-T) \tag{4.9}$$

$$V_e = V_h(t-T) \tag{4.10}$$

Using the hybrid matrix notation for the communication channel that is modeled as a 2-port network,

$$\begin{bmatrix} F_h(s) \\ -V_e(s) \end{bmatrix} = \begin{bmatrix} 0 & e^{-sT} \\ -e^{-sT} & 0 \end{bmatrix} \begin{bmatrix} V_h(s) \\ F_e(s) \end{bmatrix} \tag{4.11}$$

Scattering theory has been widely used for the stability analysis of force reflecting teleoperators with time delay. According to the scattering theory, a scattering operator, S, is defined as a mapping between the difference and the sum of force and velocity:

$$f - v = S(f + v) \tag{4.12}$$

They the stability criterion of a 2-port network is given by

$$\|S(s)\| = \sup_{\omega} \lambda^{1/2}\left(S^*(j\omega)S(j\omega)\right) \le 1 \tag{4.13}$$

where λ is the eigenvalue of $S(s)$. The "*" here represents the transpose conjugate of $S(s)$.

Equation 4.13 is also a definition of a passive system. The criterion states that in order for a system to be passive and therefore stable, the norm of its scattering matrix, which is the maximum eigenvalue of $S^*(j\omega)S(j\omega)$, must not exceed unity. This, for example, is not true for the communication channel that is represented using the hybrid notation in Equation 4.11 for which the scattering representation and the scattering matrix are given as

$$\begin{bmatrix} F_h(s) - V_h(s) \\ F_e(s) + V_e(s) \end{bmatrix} = S(s) \begin{bmatrix} F_h(s) + V_h(s) \\ F_e(s) - V_e(s) \end{bmatrix} \tag{4.14}$$

$$S(s)=\begin{bmatrix}-\tanh(sT) & \operatorname{sech}(sT)\\ \operatorname{sech}(sT) & \tanh(sT)\end{bmatrix} \tag{4.15}$$

and its norm is given as

$$\|S(s)\|=\sup_{\omega}\left(|\tan(\omega T)|+|\sec(\omega T)|\right)=\infty \tag{4.16}$$

Time delay, therefore renders the teleoperation system unstable because of its non-passivity.

For any 2-port network, the scattering matrix can be derived from a given hybrid matrix $H(s)$ as,

$$S(s)=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\left(H(s)-I\right)\left(H(s)+I\right)^{-1} \tag{4.17}$$

Niemeyer and Slotine in [76][39] introduced the concept of *wave variables* to guarantee stability of force reflecting systems under unknown but constant transmission delays. In wave-based teleoperation, wave variables u_m and v_s are sent over the communication channel instead of the original variables V_h and F_e, as illustrated in Figure 4.5. The details of “Wave Variables Transformation" block will be explained in the next section (Section 4.2.2).

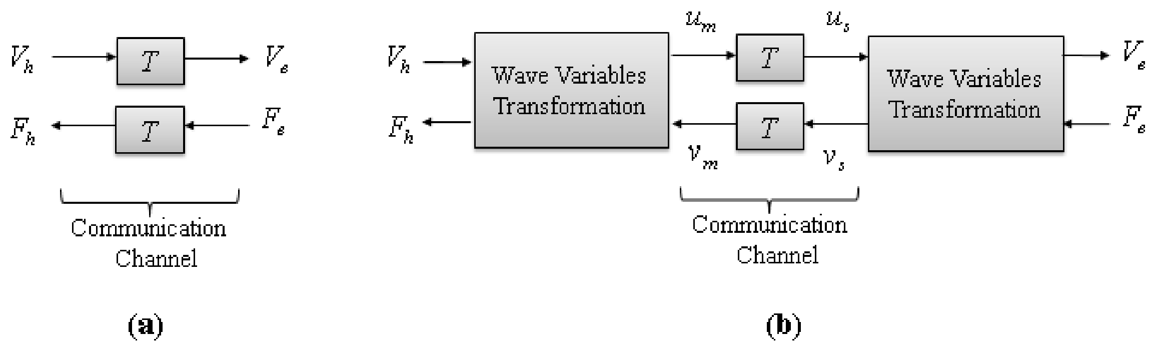

Figure 4.5 Communication (a) before wave-variables transformation, (b) after transformation.

Due to the introduction of wave variables, the resulting scattering matrix $S(s)$ comprises only the delay exponentials:

$$S(s) = \begin{bmatrix} 0 & e^{-sT} \\ -e^{-sT} & 0 \end{bmatrix} \tag{4.18}$$

which has the norm equal to one and therefore the resulting system is passive and stable despite the time delay during transmission.

4.2.2 Wave-based Haptic Teleoperation

Passivity Theory

For the last two decades, passivity-based control has provided has been a strong approach towards delay-independent stabilization of bilateral teleoperators. Pioneer work in this area came from Anderson and Spong [38] and Niemeyer and Slotine [77]. The underlying key concept is to represent a master-slave teleoperator system as an interconnection of two-port networks and then encode the velocity and force signals as so-called scattering variables before transmitting them over the network. This approach renders the communication channel passive and the entire system stable, independent of the time delay. Scattering theory uses the notion of scattering transformations which relate the sum of velocity and force to their difference, so that passivity becomes a condition on the scattering matrix (system gain), which is unaffected by the delay. The other popular approach for dealing with time delay in teleoperation, i.e., TDPC [45] [46], is also based on the passivity theory.

Passivity theory deals with the exchange of energy between interconnected systems and can be applied to both linear and nonlinear systems. In order to make use of passivity theory for bilateral teleoperation with time delay, the teleoperation system (see Figure 4.4) is modeled as an interconnection of one- or two-port networks where energy at each port is exchanged in the form of an effort-flow pair of (power) variables. For a mechanical system the effort-flow (or through-across) pair corresponds to the force-velocity pair, which for a robot manipulator may also be interpreted as the torque-angular velocity pair at each revolute joint or the force-linear velocity pair at each Cartesian axis that is assigned to the end effector. If all the systems (one- or two-port networks) in Figure 4.4 (human operator, master/slave manipulators, and the environment) are passive, their interconnection (whether in parallel or feedback) is still passive and overall stability of the complete teleoperation system is therefore guaranteed.

In the network representation of the teleoperation system (Figure 4.4), the human operator and the environment are modeled as one-port systems and can be treated as mechanical admittance and mechanical impedance, respectively. According to passivity theory, a one-port system is passive if the following condition holds:

$$E(t) = \int_{-\infty}^{t} f(\tau)\dot{x}(\tau)d\tau \geq 0 \qquad \forall t \tag{4.19}$$

Or,

$$E(t) = \int_{0}^{t} f(\tau)\dot{x}(\tau)d\tau + E(0) \geq 0 \qquad \forall t > 0 \tag{4.20}$$

Here

$$E(0) = \int_{-\infty}^{0} f(\tau)\dot{x}(\tau)d\tau \tag{4.21}$$

is the initial energy stored in the system. For a two-port network having an input conjugate pair of power variables $f_1(t), \dot{x}_1(t)$ and an output pair $f_2(t), \dot{x}_2(t)$, the condition of passivity is given as

$$E(t) = \int_0^t \left(f_2(\tau)\dot{x}_2(\tau) - f_1(\tau)\dot{x}_1(\tau) \right) d\tau + E(0) \geq 0 \qquad \forall t > 0 \tag{4.22}$$

The above two conditions state that there is no internal power generated by the system. A bounded input energy supplied to the system results in bounded output energy. The system is therefore stable in the *input-output* sense. These definitions actually originated from electrical networks, but can be analogously applied to mechanical systems.

It is a reasonable assumption to consider the human operator and the environment (which in the case of homecare may also be a human) as passive systems. Practically, this means that the humans, being part of the closed-loop, do not input random and out-of-bound motions and forces to the system. The human operator firmly holds the end effector of the haptic interface (to avoid contact instability) at all times during teleoperation and does not exceed the magnitude and frequency of arm motion beyond a certain threshold. The human's passivity assumption is valid as long as active movements of the human arm (which is the source of input motion to the teleoperation system) remain under 10Hz [78].

The master and slave manipulators, which are nonlinear systems, can be controlled in such a way that their passivity is guaranteed. Passivity-based analysis and design of nonlinear control systems have been successfully used and have demonstrated satisfactory results as can be found in the literature. A strong relationship exists between passivity and Lyapunov stability, which forms the basis for the design of robot motion control where the goal is to achieve a globally asymptotically stable system.

By treating the communication channel as a two-port network, energy stored at time t can be computed as

$$E(t) = \int_0^t \left(f_m^d(\tau)\dot{x}_m(\tau) - f_s(\tau)\dot{x}_s^d(\tau) \right) d\tau + E(0) \geq 0 \qquad \forall t > 0 \tag{4.23}$$

This condition on passivity of the communication channel does not hold true in the presence of time delay.

Wave Variables

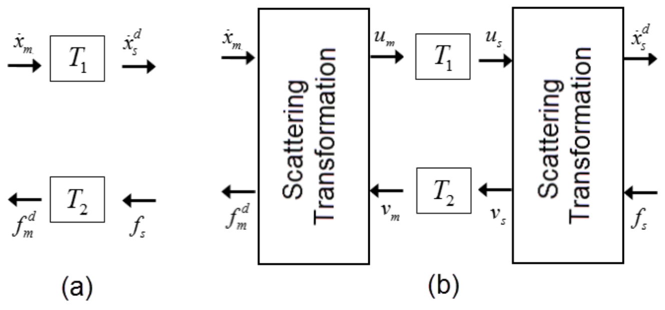

Figure 4.6 Communication channel with wave variables.

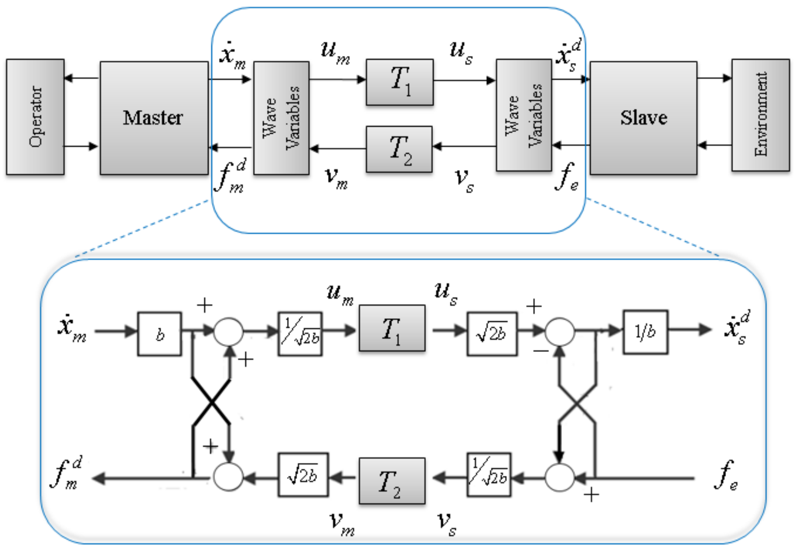

Figure 4.7 Wave-based teleoperation architecture.

Introduction of wave-variables transformation, as shown in Figure 4.6, renders the communication channel passive, even in the presence of time delay. A complete teleoperation system with wave variables is shown in Figure 4.7. It is seen that the *wave variables* u_m and v_s are exchanged between the master and the slave sides, instead of the actual power variables $\dot{x}_m$ (the velocity of the master as measured directly or derived from the measured position) and f_m^d

(the interaction force). The transformation between the wave variables and power variables is given by

$$u_m(t) = \frac{1}{\sqrt{2\text{b}}}\left(b\dot{x}_m(t) + f_m^d(t)\right) \tag{4.24}$$

$$v_s(t) = \frac{1}{\sqrt{2\text{b}}}\left(b\dot{x}_s^d(t) - f_s(t)\right) \tag{4.25}$$

where b is a positive constant referred to as the *characteristic wave impedance*, which plays an important role in determining the system response. The variable u_m is called the *forward* or *right-moving* wave as it is being sent by the master side to the slave. Similarly v_s is called the *backward* or *left-moving* wave. These (wave) variables when received by the respective ends are represented as

$$u_s(t) = u_m(t - T_1) \tag{4.26}$$

$$v_m(t) = v_s(t - T_2) \tag{4.27}$$

where T_1 and T_2 are the time delays induced by the communication channel during transmission. On receiving the transmitted wave variables, the master and the slave ends have to decode the wave variables in order to derive the corresponding actual power variables as

$$f_m^d(t) = b\dot{x}_m(t) - \sqrt{2\text{b}}v_m(t) \tag{4.28}$$

$$\dot{x}_s^d(t) = \frac{1}{b}\left(\sqrt{2b}u_s(t) - f_s(t)\right) \tag{4.29}$$

From Equations 4.24 to 4.29, the power variables can be written in terms of the wave variables as

$$f_m^d(t) = \sqrt{\frac{b}{2}}\left(u_m(t) - v_m(t)\right) \tag{4.30}$$

$$\dot{x}_m(t) = \frac{1}{\sqrt{2b}}\left(u_m(t) + v_m(t)\right) \tag{4.31}$$

$$f_s(t) = \sqrt{\frac{b}{2}}\left(u_s(t) - v_s(t)\right) \tag{4.32}$$

$$\dot{x}_s^d(t) = \frac{1}{\sqrt{2b}}\left(u_s(t) + v_s(t)\right) \tag{4.33}$$

It is possible now to transform the expression for energy stored in the communication channel (Equation 4.23) into the wave-domain as (assuming that $T_1 = T_2 = T$ for the sake of simplicity)

$$E(t) = \frac{1}{2}\int_0^t \left(u_m^2(\tau) - v_m^2(\tau) - u_s^2(\tau) + v_s^2(\tau)\right) d\tau \tag{4.34}$$

$$E(t) = \frac{1}{2}\int_{t-T}^{t} u_m^2(\tau)d\tau + \frac{1}{2}\int_{t-T}^{t} v_s^2(\tau)d\tau \geq 0 \tag{4.35}$$

This states that the passivity condition has been met and the communication channel is stable for any arbitrary time delay T.

From Figure 4.7 and also from Equations 4.24 to 4.25, it is noticed that a sending wave variable is computed partly based on the incoming signal (force on the master side and position on the slave slide). This forms an algebraic loop where part of the received information is reflected back to the sender. This phenomenon is known as *wave reflection* and is illustrated in Figure 4.8.

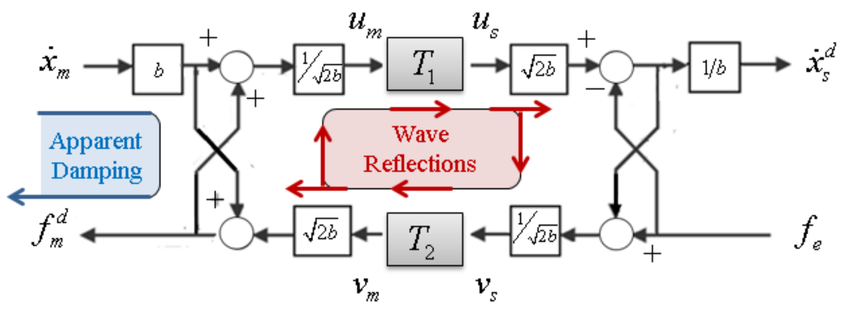

Figure 4.8: Wave reflections in wave-based teleoperation system.

Wave reflections can disorient the human operator by causing oscillatory behavior and poor transient response. In order to cancel out the reflected wave information, Niemeyer and Slotine [39] suggested that the impedances on the two ends must be matched properly. Their solution is shown in Figure 4.9 where two blocks "b" and "$\frac{1}{b}$" are appended at the right and left ends, respectively.

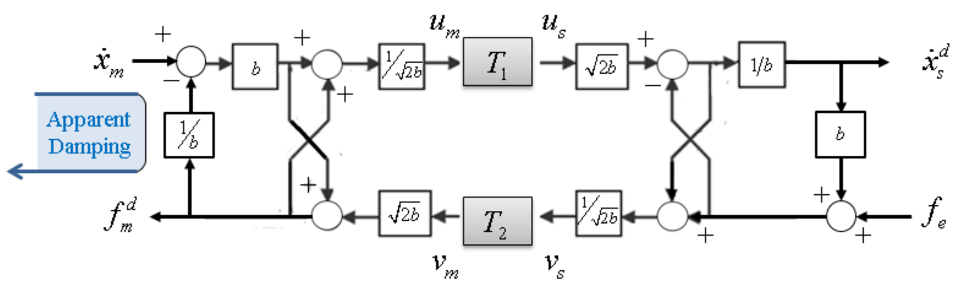

Figure 4.9 Cancellation of wave reflections in wave-based teleoperation system.

This resulted in the following re-computation of forward and backward wave-variables which does not involve a component of the reflected signal anymore:

$$u_m(t) = \sqrt{\frac{b}{2}}\dot{x}_m(t) \tag{4.38}$$

$$v_s(t) = \frac{-f_s(t)}{\sqrt{2b}} \tag{4.39}$$

The corresponding actual power variables are derived from the received wave variables as

$$\begin{aligned} f_m^d &= \frac{b}{2}\dot{x}_m - \sqrt{\frac{b}{2}}v_m \\ &= \frac{b}{2}\dot{x}_m + \frac{f_s(t-T)}{2} \end{aligned} \tag{4.40}$$

$$\begin{aligned} \dot{x}_s^d &= -\frac{1}{2b}f_s + \frac{1}{\sqrt{2b}}u_s \\ &= \frac{1}{2}\dot{x}_m(t-T) - \frac{1}{2b}f_s \end{aligned} \tag{4.41}$$

This solves the wave reflection problem, but as can be noticed from Equations (4.40) and (4.41), transparency still remains compromised. Ideally the desired signals received at the master and the slave ends should be the delayed version of what was sent by the corresponding end; i.e., $f_m^d = f_s(t-T)$ and $\dot{x}_s^d = \dot{x}_m(t-T)$. But this is not the case in Equations (4.40) and (4.41). The master has to feel an unwanted damping (as shown in Figure 4.9) and therefore the displayed force is distorted. Similarly, for the slave side the desired velocity is also not the same as sent by the master side. In summary, force and velocity tracking is not ideal.

To solve this problem and improve position and force tracking in wave-variables based teleoperation, Li and Kawashima in a recent work [79] have proposed a modification to the impedance-matched wave-variable scheme. According to their solution (shown in Figure 4.10), before retrieving the actual variables, the wave variables at the receiving end need to be modified as [79]:

$$u_s(t) = \sqrt{2\text{b}}\dot{x}_m(t-T) + \frac{f_s(t)}{\sqrt{2\text{b}}} \tag{4.42}$$

$$v_m(t) = \sqrt{\frac{b}{2}}\dot{x}_m(t) - \sqrt{\frac{2}{b}} f_s(t-T) \tag{4.43}$$

or alternatively,

$$u_s(t) = 2u_m(t-T) - v_s(t) \tag{4.42}$$

$$v_m(t) = 2v_s(t-T) + u_m(t) \tag{4.43}$$

Substitution of these new variables into Equations (4.40) and (4.41) results in the following,

$$f_m^d = f_s(t-T) \tag{4.44}$$

$$\dot{x}_s^d = \dot{x}_m(t-T) \tag{4.45}$$

These are ideal command for the master and slave controllers to track. Transparency has been improved since the unwanted terms have been removed. The new system with modified wave variables is shown in Figure 4.10.

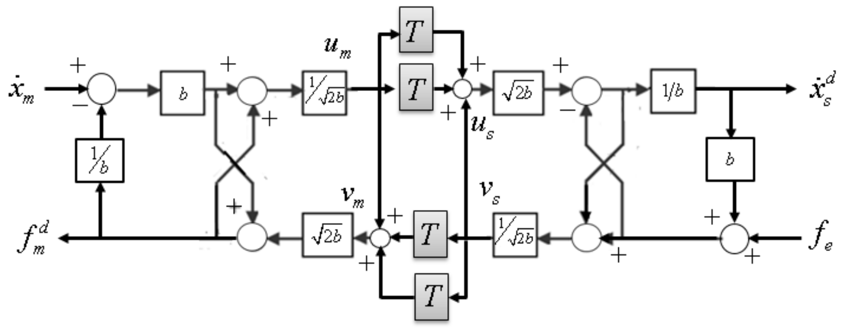

Figure 4.10 Modified wave-based teleoperation system. (*Adapted from* [42]).

Position-error Correction in Wave-based Teleoperation

Poor position tracking results in sluggish behavior of the teleoperation system and it becomes worse with the increase in time delay. This problem, known as *position drift* in bilateral teleoperation, has been widely studied by researchers.

Passivity-based control schemes such as wave-based teleoperation provide stability against time delays and guarantee perfect velocity tracking. However, these approaches fail to guarantee perfect position and force tracking in steady state. This can be attributed to the fact that only power variables (velocity and force) are exchanged between the master and the slave sides, and the position information is not explicitly encoded. It is obtained by numerical integration of velocity, which is prone to error. There may also exist an offset in the initial conditions between the two sides. Data loss during transmission further aggravates the problem. Furthermore, it is typically hard to achieve perfect impedance matching between the master and the slave manipulators, which have dissimilar kinematics and dynamics. For example, in the homecare system that is considered in this thesis, a small and lightweight haptic interface communicates with a heavy and larger industrial manipulator. Even if the nonlinearities are perfectly dealt with (removed) through feedback linearization, the impedances on the two sides cannot be perfectly matched no matter what the value of the wave impedance b is. Even if the master and the slave manipulators are similar, the commonly used control methods such as impedance control and admittance control require exclusive position information for perfect motion tracking. This was observed in Chapter 3 where the position tracking capability of an admittance controller has been improved by incorporating the desired position information into the controller along with the desired velocity. For the Homecare system developed in the thesis, the proposed solution takes into account the exchange of position information along with the wave variables.

Implementation of the wave-variable algorithm is difficult due to the inherent algebraic loops. A complete wave-variable algorithm is presented next to avoid these loops. It is assumed here that a position-force architecture is used for teleoperation. The desired force as received from the slave is directly displayed to the human operator via the haptic interface. It may be modified depending on the control scheme (such as impedance control of the master). Similarly, the slave controller is a first-order impedance controller.

The Complete Wave Variables Algorithm

At the Master Side:

1. Receive v_m from the slave
2. Modify v_m as in Equation 4.43
3. Read measured position x_m and velocity $\dot{x}_m$ from the haptic device
4. Compute f_m^d using Equation 4.40 and input it to the haptic device
5. Compute u_m using Equation 4.38 and send it via UDP to the slave side over the communication channel

At the Slave Side:

1. Receive u_s from the master
2. Modify u_s as in Equation 4.42
3. Read measured position x_s and velocity $\dot{x}_s$ from slave manipulator
4. Compute $\dot{x}_s^d$ (assuming a PD control law is used as in Equation 4.47)

$$\begin{aligned} \dot{x}_s^d &= -\frac{1}{2b}\left(k_{ps}(\mathrm{x}_s^d - x_s) + b_{ds}(\dot{x}_s^d - \dot{x}_s)\right) + \frac{1}{\sqrt{2b}}u_s \\ \dot{x}_s^d &= \frac{k_{ps}(x_s - x_s^d) + b_{ds}\dot{x}_s + \sqrt{2b}u_s}{2b + k_{ds}} \end{aligned} \tag{4.46}$$

5. Compute x_d by integrating $\dot{x}_s^d$

6. Compute the PD control force term f_s (and output it to the device)

$$f_s = k_{ps}\left(x_s^d - x_s\right) + b_{ds}\left(\dot{x}_s^d - \dot{x}_s\right) \tag{4.47}$$

7. Measure the interaction force f_e

8. Compute v_s using Equation (4.39) and send it via UDP to the master side

4.3 Experimental Results

The same experimental setup as used in Chapter 3 is employed here, with the time delays emulated.

Emulating Time Delay

Network emulation tools emulate Wide Area Network (WAN) characteristics using a Local Area Network (LAN). They can be used to emulate time delay (both constant and random) and other communication constraints such as packet loss, packet corruption, packet reordering, congestion, and bandwidth limitation for application development and testing in areas such as teleoperation, haptics, online video gaming, and networked control systems. Although a variety of tools exist (such as NetEm, NISTNet, WANsim, and WANEm), in this thesis WANem [80] is used because of three main reasons: (1) it is freely available, (2) it allows setting rules for both incoming and outgoing packets on the WAMem router (PC), (3) it is easily configurable and is more realistic. WANem is based on the underlying `netem` tool of the Linux kernel, which is provided by Linux for testing protocols by emulating the network properties of wide area networks. The `netem` tool models the network delay as a sum of fixed delay, called *latency*, and random delay, called *jitter*. The latter is a variation around the fixed delay and is modelled by probability density functions such as Normal or Pareto. A customized non-uniform distribution based on experimental data can also be specified.

Figure 4.11 shows the experimental setup for teleoperation with time delay where a WAN Emulator PC, with WANem installed and configured, induces controlled delays into traffic between the Master and Slave Controllers. As shown, data between the master and slave ends is transmitted in the form of UDP/IP packets over a 100Mbps Ethernet via the WAN Emulator PC. In this thesis, only a constant time delay is emulated. No jitter, packet loss, duplication, and re-ordering is considered. Extensions of the wave variables approach to incorporate the effects of these properties exist in the literature.

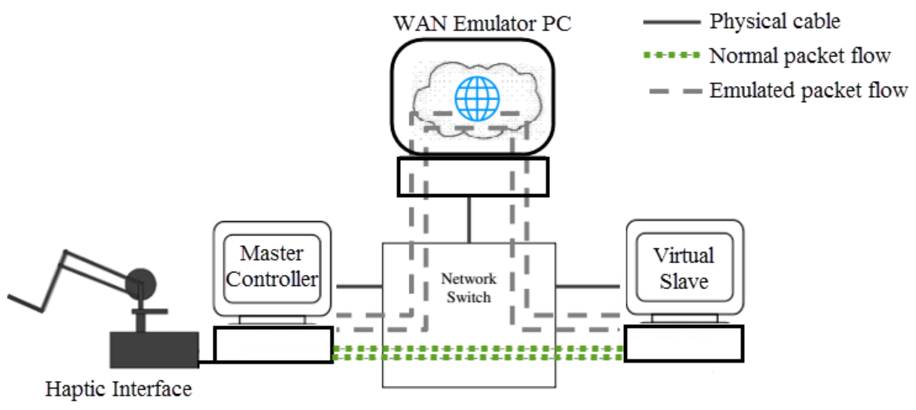

Figure 4.11 Delay emulation in haptic teleoperation through WANem.

Experiments

Two sets of experiments are performed, one for the average time delay of 200ms and the other for 400 ms. These are the magnitudes of worst-case delays for traffic over the Internet. According to the live Internet Weather Map™ [81] that displays and refreshes latency on the

Internet every 30 seconds, warnings are issued when the latency gets over 300 ms. The wave variable approach, however, can handle any arbitrary amount of delay.

Results of the experiments are given in Figures 4.12 to 4.17, which show perfect position tracking, velocity tracking, and force feedback between the master and the slave ends. Figure 4.12 and Figure 4.13 show plots of the actual and the desired positions and velocities of the slave manipulator during tracking. The slave manipulator interacts with a virtual environment and sends the resulting interaction force back to the master, which is shown in Figure 4.14.

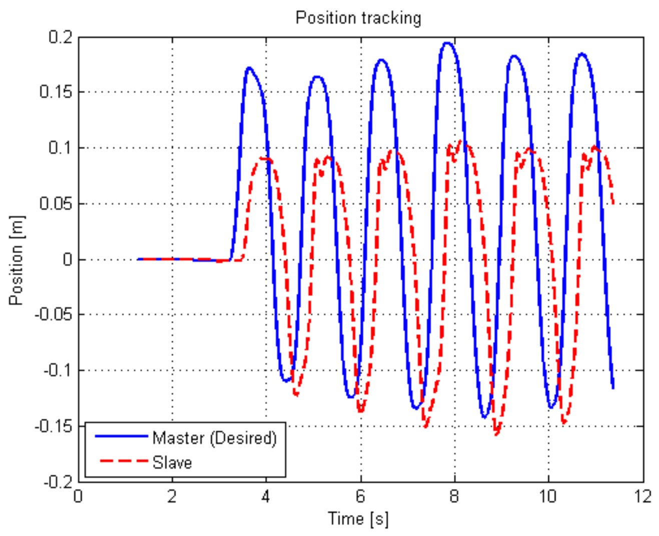

Figure 4.12 Position tracking under a time delay of 200ms.

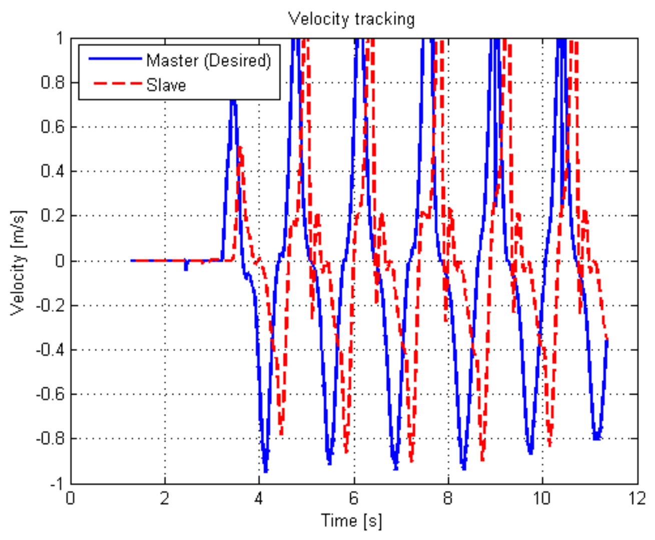

Figure 4.13 Velocity tracking under a time delay of 200ms.

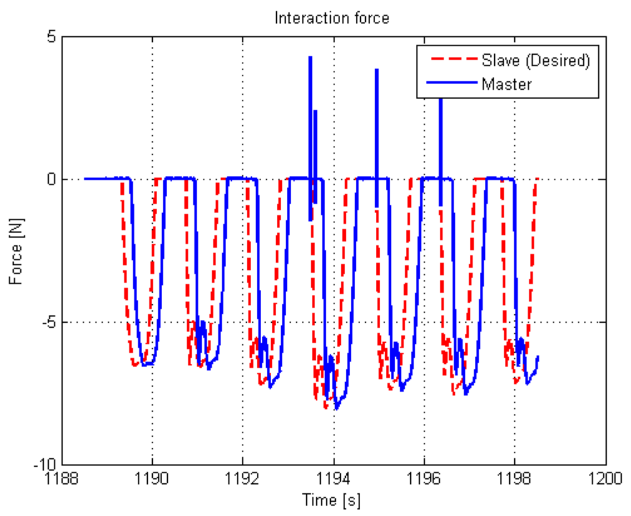

Figure 4.14 Force tracking under a time delay of 200ms.

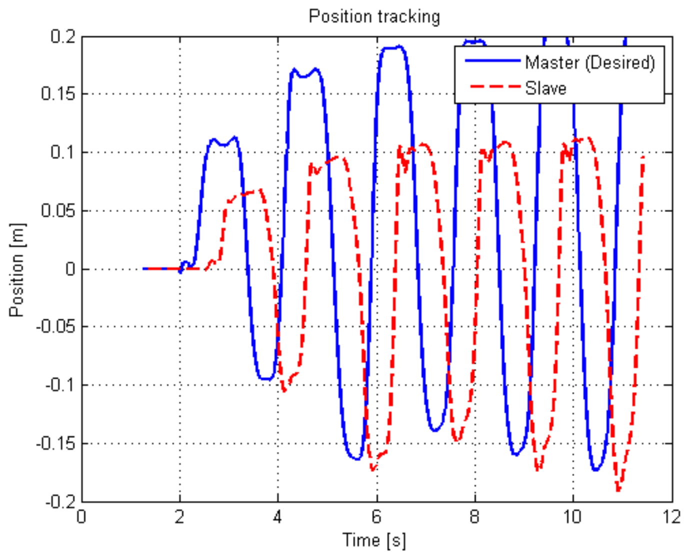

Figure 4.15 Position tracking under a time delay of 400ms.

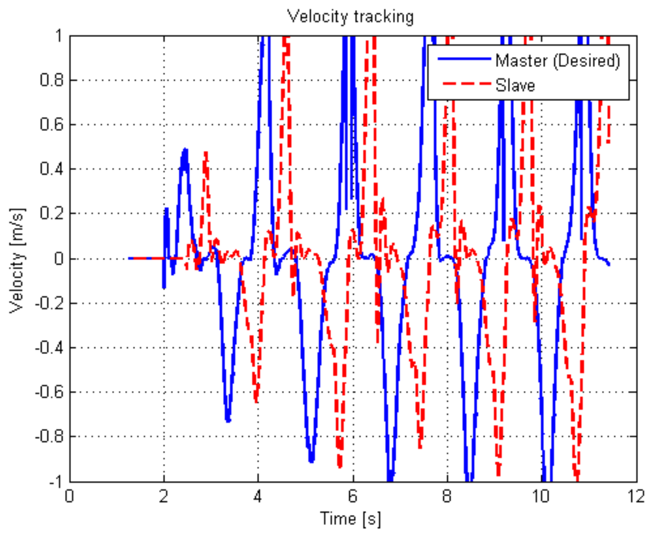

Figure 4.16 Velocity tracking under a time delay of 400ms.

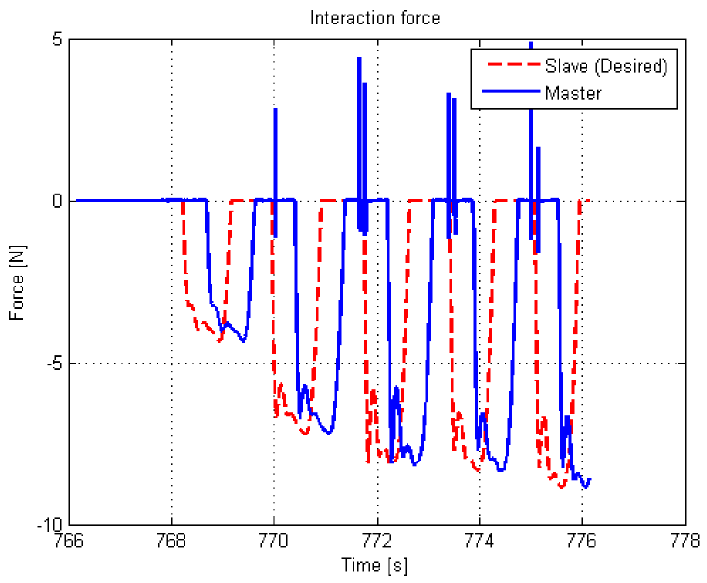

Figure 4.17 Force tracking under a time delay of 400ms.

4.4 Summary

In this chapter, wave variable-based algorithm was used to stabilize a haptic teleoperation system with time delay. Results that produced stable position and force tracking were presented. It was shown that sending the desired position along with the wave variables can solve the position drift problem. This would also allow for integration of all the control schemes that were presented in the previous chapter. The issues of wave reflection and transparency have been solved, and the controller design of both master and slave manipulators can be carried out independent of the dynamics of the communication channel and by tuning the wave impedance parameter *b*.

Chapter 5: Conclusion and Future Work

This thesis focused primarily on the use of an impedance control approach to design controllers for the master and the slave manipulators in a teleoperation system for elderly care in a home environment. An inverse dynamic model of the robot was identified using a learning-based approach that did not require the availability of knowledge about the existing model. The resulting impedance controller, which used the learned model in a feed forward manner, was implemented in real-time and was found to be capable of adapting to changes in the arm's real dynamics. The desired impedance model was adaptively selected based on the estimated impedance of the unknown environment with which the slave manipulator was in contact. A method of admittance control was designed and its merits investigated compared to the standard impedance control based on experimental results. Lastly, the transparency and position synchronization of the popular wave variable approach was improved. This allowed stable position and force tracking in teleoperation under time delay that was induced by the communication channel during the exchange of information between the master and the slave ends. All the algorithms developed were validated in the context of homecare robotics through simulation and experimentation, and the results were presented and discussed.

This chapter concludes the thesis with a description of the overall significance of the work and the key research contributions to the broader discipline of haptic teleoperation in the context of homecare robotics. Limitations and strengths of the work are discussed. Finally, possible directions for future research are reported, which may further improve the performance and effectiveness of a homecare robotic system.

5.1 Conclusions

In this thesis, interaction control algorithms were investigated, designed, analyzed, and evaluated, which would ensure safe and compliant behavior of a robotic system in a home environment in giving routine care to the elderly or the disabled and assisting in daily tasks. Pure position-, force-, or hybrid control techniques cannot achieve the simultaneous conflicting goals of controlling the position and force, and are therefore not suitable for dealing with contact situations. Stiffness control has long been tried, but it is unable to cope with the nonlinear and coupled robot dynamics. Impedance control, therefore, has been a preferred method of choice for robotic control, especially in the homecare and service industry where the robot has to coexist with humans and delicate objects in its surroundings. Due to its known advantages over other approaches, practical application of impedance control in real-time is still a challenging task and therefore many applications in industry have still not benefited from it. This thesis has attempted to address these challenges.

Chapter 2 of this thesis investigated the kinematic considerations that could ensure a successful implementation of impedance control. As the impedance control formulates the robot dynamic model in the operational space, the presence of singular configurations in a manipulator's workspace can adversely affect the performance and stability of impedance control. The operational space formulation of dynamic model makes extensive use of the manipulator's Jacobian matrix, which relates the joint velocities to the Cartesian velocities of the end-effector. At the singularity, the Jacobian matrix loses its rank which results in extremely large forces and torques, loss of stiffness, and ultimately instability of the control algorithm. Manipulability measures such as velocity and force ellipsoids were therefore used in the kinematic transformation of the master and slave motion in the teleoperation system. This helped in finding optimal configurations (a subset of the complete workspace) of the slave manipulator that are away from singularities and therefore safe. The slave manipulator, prior to making contact with the environment, was put in one of these safe regions before tracking the motion commands through the haptic interface. Complete kinematic models of an industrial manipulator (WAM) and haptic interface (PHANToM) were also derived in Chapter 2, and an experimental testbed for haptic teleoperation was developed.

A realistic implementation of impedance control is hard to achieve on manipulators that have significant dynamics. Traditionally this problem has been solved by feeding forward the inverse dynamic model, which is computed ahead of time based on the desired motion trajectory (positions, velocities, and accelerations). This results in reduced computational burden as the dynamic terms, which are now computed offline and stored in memory, just need to be fetched. However, like any feedforward control scheme, there is the inherent problem of lack of robustness with respect to parametric and structural uncertainties in the manipulator dynamics. If the inverse dynamic model is computed offline and never updated, the robustness property will not be achieved. In Chapter 3, a new impedance control algorithm was proposed which combined feedforward control with incremental online learning of the inverse dynamic model using machine learning techniques. No a-priori knowledge about the system dynamics was assumed. The learned model can not only be trained offline, but also updated in real-time and is therefore robust to any changes in system dynamics. For this purpose, in Chapter 2 of this thesis, a fast algorithm, LWPR, was used which is a statistical regression technique with learning capability to estimate the nonlinear inverse dynamic function. The learned inverse-dynamic feed-forward torque was provided to the inner torque loop of the impedance controller. The impedance control gains were then tuned to achieve the desired dynamic behavior. The resulting arm had a “soft" feel and behaved in a compliant way as demonstrated by the experimental results on a commercial 4-DOF WAM arm and a PHANToM device (a commercial haptic interface). As WAM has significant dynamics compared to the haptic interface, its dynamic model was analytically derived and results of impedance control based on both models (analytical and experimental) were presented.

Once the impedance control implementation in real-time was achieved, the next objective was to experimentally evaluate different types of impedance control that exist in the literature. In Chapter 3, three impedance control schemes were assessed in the laboratory on a commercially available industrial manipulator and a haptic interface. The evaluated methodologies were standard (second-order) impedance control, first-order impedance control, and admittance control. While first-order impedance control enjoyed easy implementation in real-time (based on feed-forward learned inverse dynamic model), it had a major drawback as observed from experimental results. The stiffness and damping parameters in the desired impedance model are normally selected in order to achieve a desired level of compliance or stiffness. In applications such as homecare robotics where a robot may encounter an environment of unknown stiffness, these parameters must be kept small in order for the robot to have an overall “soft" and safe feel in a desired Cartesian direction. However, this softness comes at the price of loosing the force- and position-tracking accuracy in that direction. On the other hand, if these controller gains are selected high in order to suppress the effect of nonlinear dynamics (which may still exist due to inaccuracy of the estimated model), the robot will feel stiff, which may be unsafe in a caregiving situation. This trade-off between compliance and position/force tracking accuracy makes the selection of the desired impedance model in standard and first-order impedance control extremely difficult. This problem was solved by using admittance control, which uses a separate inner loop for achieving motion tracking and a different outer loop for shaping the dynamic behavior of the manipulator. Experiments and numerical simulations were carried out to verify the effectiveness of admittance control as compared with the other two schemes. Algorithm and guidelines for selecting the desired impedance parameters were also given.

Finally, Chapter 4 presented an algorithm for improving transparency and tracking capabilities of the popular wave-variable approach that deals with time delay in haptic teleoperation. The wave-variable approach makes the teleoperation stable and insensitive to time delays by transmitting wave variables instead of velocities and forces through the communication channel. However, this stability comes at the price of degraded transparency where the force feedback that is displayed to the human operator on the master side is not exactly the same as intended. It was shown that the transmission of position information along with the wave variables can result in stable position and force tracking and therefore enhance transparency. This was verified through experimentation.

5.2 Contributions

This thesis aimed at solving some of the challenges in successful implementation of robotic homecare systems. Specifically, it sought to ensure that the master and the slave manipulators behaved in as much compliant manner as possible. At the same time, the accuracy of motion tracking during interaction must not be compromised. Towards these goals, this thesis has made the following four key contributions:

1. The inverse dynamic model of a robot manipulator with significant dynamics was identified using a complete data-driven learning-based technique which did not assume any a-prior knowledge of the dynamic parameters. The developed model can be used and

updated online within the real-time control loop of the manipulator and therefore solves a major problem in the implementation of impedance control in haptic teleoperation.

2. An impedance control scheme based on learned-inverse dynamic model was proposed which could achieve the desired compliant behavior despite the parametric and structural uncertainties in the robot dynamic model. Experimental validation of the proposed controller was conducted on a 4-DOF WAM arm. In addition to this, three interaction control techniques: standard impedance control, first-order impedance control and admittance control, were designed and experimentally evaluated for their performance and applicability in homecare robotics.
3. Position synchronization and transparency of the popular wave variable-based approach for time-delay compensation in haptic teleoperation was improved.
4. A complete experimental testbed for haptic teleoperation with special focus on safety and reliability was developed.

5.3 Significance of the Work

The algorithms proposed and developed in the current work are primarily intended for application in a robotic homecare system, which will become more common in the next two decades. However, the same algorithms can find applications in other fields as well.

There is a shift in research and technology towards the development of more complex robotic systems such as humanoid robots or 2-armed robots that can have up to 40 degrees of freedom or more. Deriving kinematic and dynamic models for such complex robots analytically will not be possible. The complete data-driven learning framework developed in this thesis can therefore be a preferred solution for model-based control of such complex robotic systems.

Cooperative manipulation where several robots jointly manipulate an object is used in such applications as robotic surgical systems where humans and robots work cooperatively perform a useful task. Admittance control, which results in safe interaction, even in a stiff environment is desirable for interaction control. The desired admittance model can incorporate behavior for guided motion of cooperative manipulators.

Other application areas where the developed interaction control algorithms can be useful include space robotics, telesurgery, and remote access to hazardous or difficult environments.

5.4 Limitations and Suggested Future Work

There are several factors that limit the effectiveness of the algorithms and techniques developed in this thesis. In Chapters 2 and 3, the learned-dynamic model is expressed in the joint-space. Therefore it is not well-suited for use in an impedance control architecture. The reliance on inverse kinematics could not be eliminated. For redundant robots, the inverse kinematics is not only a challenging task, but is an ill-posed problem. A possible future topic of research would be to represent the learned model in the Cartesian space.

The online identification of the environmental model in Chapter 3 may also be achieved using a learning framework. Many tasks in a home environment are repetitive and involve interaction with the same environment (for example, opening a door, or cleaning a floor). A set of models may be developed off line, one for each environment, and stored in memory. The online execution of a task could benefit from such a model base, which could be updated at the same time. This could be a direction of future work. Furthermore, task geometry could also be taken into consideration and a sophisticated task planner module could be added to the impedance control framework. Vision sensors may be incorporated for this purpose.

In impedance control, the desired dynamic behavior has be provided in the task space. Robot manipulators, on the other hand, find it hard to ensure smooth tracking of the desired trajectory in the Cartesian space. All industrial robots achieve perfect position tracking in the joint-space. This is further improved by interpolation and blending of the desired joint motion. In a haptic teleoperation setup, where the desired Cartesian trajectory for the slave manipulator is received from the master manipulator, trajectory optimization must be done before being tracked by the slave.

Finally, in order to ensure stable and transparent force display at the haptic interface, model-mediated teleoperation [82] needs to be investigated further. Rather than directly sending the slave motion and force information to the master, a model of the environment may be transmitted to the master where it is displayed to the human operator. The slave then can receive commands from the master consistent with the model. Alternatively, the module of environmental model estimation on the slave side may be transferred to the master side, which can be used to provide the human operator with a predicted environmental force [83]. The environmental model can then be updated as and when the information resulting from interaction with the actual environment is received from the slave end. This would allow a stable and transparent teleoperation in the presence of substantial communication delays.

REFERENCES

[1] Statistics Canada, *The Canadian Population in 2011: Age and Sex, 2011 Census*. The Ministry of Industry, Canada, 2012.

[2] Statistics Canada, “Table 051-0001 - Estimates of population, by age group and sex for July 1, Canada, provinces and territories.” The Ministry of Industry, Canada, 2015.

[3] Canadian Federation of Nurses Unions, “The Nursing Workforce,” 2013. [Online]. Available: https://nursesunions.ca/sites/default/files/2013.backgrounder.nursing_workforce.e.pdf. [Accessed: 10-Feb-2015].

[4] International Federation on Ageing, “Long Term Care and Technology,” 2012. [Online]. Available: http://www.ifa-fiv.org/wp-content/uploads/2012/11/som-2012-ltc-and-technology-final-report.pdf. [Accessed: 12-Mar-2015].

[5] L. Bos, A. Dumay, L. Goldschmidt, G. Verhenneman, and K. Yogesan, Eds., *Handbook of Digital Homecare: Successes and Failures*. Springer Berlin Heidelberg, Berlin, 2011.

[6] S. Klanke, S. Vijayakumar, and S. Schaal, “A library for locally weighted projection regression,” *J. Mach. Learn. Res.*, vol. 9, pp. 623–626, 2008.

[7] S. Schaal and C. G. Atkeson, “Learning Control in Robotics,” *IEEE Robot. Autom. Mag.*, vol. 17, no. 2, pp. 20–29, Jun. 2010.

[8] N. Aghasadeghi, H. Zhao, L. J. Hargrove, A. D. Ames, E. J. Perreault, and T. Bretl, “Learning impedance controller parameters for lower-limb prostheses,” in *2013 IEEE/RSJ international conference on Intelligent robots and systems (IROS)*, 2013, pp. 4268–4274.

[9] M. D. E. R. Roberto Calandra Serena Ivaldi and J. Peters, “Learning Inverse Dynamics Models with Contacts,” in *2015 IEEE International Conference on Robotics and Automation*, 2015.

[10] C. K. I. Williams and C. E. Rasmussen, “Gaussian processes for regression,” in *Advances in Neural Information Processing Systems*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds. MIT Press, 1996, pp. 514–520.

[11] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, 2004.

[12] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: Incremental real time learning in high dimensional space," in *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000, pp. 1079–1086.

[13] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local gaussian process regression," *Adv. Robot.*, vol. 23, no. 15, pp. 2015–2034, 2009.

[14] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cogn. Process.*, vol. 12, no. 4, pp. 319–340, 2011.

[15] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Rob. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[16] O. Sigaud, C. Salaün, and V. Padois, "On-line regression algorithms for learning mechanical models of robots: A survey," *Rob. Auton. Syst.*, vol. 59, no. 12, pp. 1115–1129, 2011.

[17] C. Salaun, V. Padois, and O. Sigaud, "Learning forward models for the operational space control of redundant robots," in *From Motor Learning to Interaction Learning in Robots*, Springer, 2010, pp. 169–192.

[18] J. de la Cruz, D. Kulic, and W. Owen, "Learning inverse dynamics for redundant manipulator control," in *2010 International Conference on Autonomous and Intelligent Systems (AIS)*, 2010, pp. 1–6.

[19] J. S. De La Cruz, D. Kulić, and W. Owen, "Online incremental learning of inverse dynamics incorporating prior knowledge," in *Autonomous and Intelligent Systems*, Springer, 2011, pp. 167–176.

[20] J. de la Cruz, D. Kulić, and W. Owen, "Online Incremental Learning of Inverse Dynamics Incorporating Prior Knowledge," in *Autonomous and Intelligent Systems*, vol. 6752, M. Kamel, F. Karray, W. Gueaieb, and A. Khamis, Eds. Springer Berlin Heidelberg, 2011, pp. 167–176.

[21] D. Nguyen-Tuong and J. Peters, “Online Kernel-Based Learning for Task-Space Tracking Robot Control,” *Neural Networks Learn. Syst. IEEE Trans.*, vol. 23, no. 9, pp. 1417–1425, Sep. 2012.

[22] D. Nguyen-Tuong, J. Peters, M. Seeger, and B. Schölkopf, “Learning inverse dynamics: a comparison,” in *European Symposium on Artificial Neural Networks*, 2008, no. EPFL-CONF-175477.

[23] W. Khalil and E. Dombre, *Modeling, identification & control of robots*. Sterling, VA; London: Kogan Page Science, 2004.

[24] R. Volpe and P. Khosla, “Computational considerations in the implementation of force control strategies,” *J. Intell. Robot. Syst.*, vol. 9, no. 1–2, pp. 121–148, 1994.

[25] M. Ueberle and M. Buss, “Control of kinesthetic haptic interfaces,” in *IEEE/RSJ Int. Conf. on Intell. Robots and Syst., Workshop on Touch and Haptics*, 2004.

[26] B. Stanczyk, A. Peer, and M. Buss, “Development of a high-performance haptic telemanipulation system with dissimilar kinematics,” *Adv. Robot.*, vol. 20, no. 11, pp. 1303–1320, 2006.

[27] R. Kikuuwe, “A Sliding-Mode-Like Position Controller for Admittance Control With Bounded Actuator Force,” *IEEE/ASME Trans. Mechatronics*, vol. 19, no. 5, pp. 1489–1500, 2014.

[28] D. Feth, A. Peer, and M. Buss, “Incorporating human haptic interaction models into teleoperation systems,” in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 4257–4262.

[29] P. Hokayem and M. Spong, “Bilateral teleoperation: An historical survey,” *Automatica*, vol. 42, no. 12, pp. 2035–2057, Dec. 2006.

[30] P. Arcara, “Control schemes for teleoperation with time delay: A comparative study,” *Rob. Auton. Syst.*, vol. 38, no. 1, pp. 49–64, Jan. 2002.

[31] J. Park and O. Khatib, “A Haptic Teleoperation Approach Based on Contact Force Control,” *Int. J. Rob. Res.*, vol. 25, no. 5–6, pp. 575–591, 2006.

[32] D. R. Madill, “Modelling and Control of a Haptic Interface : A Mechatronics Approach,” University of Wateloo, 1998.

[33] K. Hashtrudi-Zaad and S. E. Salcudean, “Transparency in time-delayed systems and the effect of local force feedback for transparent teleoperation,” *IEEE Trans. Robot. Autom.*, vol. 18, no. 1, pp. 108–114, 2002.

[34] N. Hogan, “Impedance control-An approach to manipulation. I-Theory. II-Implementation. III-Applications,” *ASME Trans. J. Dyn. Syst. Meas. Control*, vol. 107, pp. 1–24, 1985.

[35] J. H. Park and H. C. Cho, “Sliding-mode controller for bilateral teleoperation with varying time delay,” in *1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 1999, pp. 311–316.

[36] N. D. Do and T. Namerikawa, “Four-channel force-reflecting teleoperation with impedance control,” *Int. J. Adv. Mechatron. Syst.*, vol. 2, no. 5–6, pp. 318–329, 2010.

[37] A. Peer, “Design and Control of Admittance-Type Telemanipulation Systems,” Technical University Munich, Germany, 2008.

[38] R. J. Anderson and M. W. Spong, “Bilateral control of teleoperators with time delay,” *IEEE Trans. Automat. Contr.*, vol. 34, no. 5, pp. 494–501, 1989.

[39] G. Niemeyer, “Telemanipulation with Time Delays,” *Int. J. Rob. Res.*, vol. 23, no. 9, pp. 873–890, Sep. 2004.

[40] N. Chopra and M. W. Spong, “Adaptive coordination control of bilateral teleoperators with time delay,” in *43rd IEEE Conference on Decision and Control (CDC)*, 2004, vol. 5, pp. 4540–4547.

[41] D. Lee and M. W. Spong, “Passive Bilateral Teleoperation With Constant Time Delay,” *IEEE Trans. Robot.*, vol. 22, no. 2, pp. 269–281, Apr. 2006.

[42] E. J. Rodríguez-Seda, “Passive Transparency Compensation for Bilateral Teleoperators with Communication Delays,” *J. Robot.*, vol. 2015, pp. 1–13, 2015.

[43] D. Sun, F. Naghdy, and H. Du, “Transparent four-channel bilateral control architecture

using modified wave variable controllers under time delays," *Robotica*, no. July 2014, pp. 1–17, 2014.

[44] H. Li and K. Kawashima, "Experimental comparison of backdrivability for time-delayed telerobotics," *Control Eng. Pract.*, vol. 28, pp. 90–96, 2014.

[45] B. Hannaford and J.-H. Ryu, "Time-domain passivity control of haptic interfaces," *IEEE Trans. Robot. Autom.*, vol. 18, no. 1, pp. 1–10, 2002.

[46] J.-H. Ryu, D.-S. Kwon, and B. Hannaford, "Stable Teleoperation With Time-Domain Passivity Control," *IEEE Trans. Robot. Autom.*, vol. 20, no. 2, pp. 365–373, Apr. 2004.

[47] H. Li and K. Kawashima, "Bilateral teleoperation with delayed force feedback using time domain passivity controller," *Robot. Comput. Integr. Manuf.*, 2015.

[48] T. B. Teleoperation and S. Member, "Time Domain Passivity Controller for," *IEEE Trans. Haptics*, vol. 8, no. 1, pp. 79–89, 2015.

[49] V. Chawda and M. K. O'Malley, "Position Synchronization in Bilateral Teleoperation Under Time-Varying Communication Delays," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 1, pp. 245–253, 2015.

[50] Barrett Technology Inc., "Libbarrett C++ library." [Online]. Available: http://support.barrett.com/wiki/Libbarrett. [Accessed: 09-Sep-2014].

[51] Barrett Technology Inc., "Whole Arm Manipulator (WAM) User Manual," 2009. [Online]. Available: http://support.barrett.com/wiki/WAM. [Accessed: 15-Mar-2015].

[52] "CHAI 3D: An Open-Source C++ Library for Haptics." [Online]. Available: http://www.chai3d.org. [Accessed: 01-Jun-2013].

[53] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*, vol. 3. Wiley New York, 2006.

[54] R. J. Schilling, *Fundamentals of Robotics: Analysis and Control*. New York: Prentice Hall, 1992.

[55] "Barrett Technology Inc. Glossary." [Online]. Available: http://www.barrett.com/glossary.htm. [Accessed: 09-Sep-2015].

[56] T. Yoshikawa, "Manipulability of robotic mechanisms," *Int. J. Rob. Res.*, vol. 4, no. 2, pp. 3–9, 1985.

[57] S. L. Chiu, "Task compatibility of manipulator postures," *Int. J. Rob. Res.*, vol. 7, no. 5, pp. 13–21, 1988.

[58] P. Corke, "Robotics Toolbox for MATLAB, release 9 [Software]." 2011.

[59] J. J. Craig, *Introduction To Robotics, Mechanics And Control*, 3rd ed. Pearson Education, 2008.

[60] E. N. Marongelli and K. A. Thoroughman, "The advantage of flexible neuronal tunings in neural network models for motor learning," *Front. Comput. Neurosci.*, vol. 7, no. 100, 2013.

[61] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Comput.*, vol. 17, no. 12, pp. 2602–2634, 2005.

[62] S. Vijayakumar, "Locally Weighted Projection Regressin (LWPR) - a users manual." 2001.

[63] M. Tufail and C. W. de Silva, "Haptic Teleoperation using Impedance Control with Application in Homecare Robotics," *Control Intell. Syst.*, vol. 41, no. 3, 2013.

[64] D. Erickson, M. Weber, and I. Sharf, "Contact Stiffness and Damping Estimation for Robotic Systems," *Int. J. Rob. Res.*, vol. 22, no. 1, pp. 41–57, 2003.

[65] H. Il Son, T. Bhattacharjee, and D. Y. Lee, "Estimation of environmental force for the haptic interface of robotic surgery," *Int. J. Med. Robot. Comput. Assist. Surg.*, vol. 6, no. 2, pp. 221–230, 2010.

[66] F. Mobasser and K. Hashtrudi-Zaad, "Transparent Rate Mode Bilateral Teleoperation Control," *Int. J. Rob. Res.*, vol. 27, no. 1, pp. 57–72, 2008.

[67] P. Ioannou and B. Fidan, *Adaptive Control Tutorial*. Cambridge University Press, Cambridge UK, 2007.

[68] W. Khalil and E. Dombre, *Modeling, identification and control of robots*. Butterworth-Heinemann, 2004.

[69] C. Hatzfeld and T. A. Kern, *Engineering Haptic Devices: A Beginner's Guide*, 2nd ed. London: Springer London, 2014.

[70] T. Imaida, Y. Yokokohji, M. Oda, T. Yoshikawa, and others, "Ground-space bilateral teleoperation of ETS-VII robot arm by direct bilateral coupling under 7-s time delay condition," *IEEE Trans. Robot. Autom.*, vol. 20, no. 3, pp. 499–511, 2004.

[71] C. R. Wagner, N. Stylopoulos, P. G. Jackson, and R. D. Howe, "The benefit of force feedback in surgery: Examination of blunt dissection," *Presence teleoperators virtual Environ.*, vol. 16, no. 3, pp. 252–262, 2007.

[72] B. Hannaford, "Stability and performance tradeoffs in bi-lateral telemanipulation," in *Proceedings IEEE International Conference on, Robotics and Automation*, 1989, pp. 1764–1767.

[73] F. B. Llewellyn, "Some fundamental properties of transmission systems," *Proc. IRE*, vol. 40, no. 3, pp. 271–283, 1952.

[74] A. Jazayeri and M. Tavakoli, "Revisiting Llewellyn's absolute stability criterion for bilateral teleoperation systems under non-passive operator or environment," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 70–75.

[75] G. A. V Christiansson and F. C. T. Van Der Helm, "The low-stiffness teleoperator slave - a trade-off between stability and performance," *Int. J. Rob. Res.*, vol. 26, no. 3, pp. 287–299, 2007.

[76] G. Niemeyer and J.-J. E. Slotine, "Stable adaptive teleoperation," *IEEE J. Ocean. Eng.*, vol. 16, no. 1, pp. 152–162, Jan. 1991.

[77] N. Diolaiti, G. Niemeyer, F. Barbagli, and J. K. Salisbury, "Stability of Haptic Rendering: Discretization, Quantization, Time Delay, and Coulomb Effects," *IEEE Trans. Robot.*,

vol. 22, no. 2, pp. 256–268, Apr. 2006.

[78] G. Burdea, *Force and touch feedback for virtual reality*. New York: John Wiley and Sons, 1996.

[79] H. Li and K. Kawashima, “Achieving Stable Tracking in Wave-Variable-Based Teleoperation,” *IEEE/ASME Trans. Mechatronics*, vol. 19, no. 5, pp. 1574–1582, 2014.

[80] H. K. Kalitay and M. K. Nambiar, “Designing WANem : A Wide Area Network emulator tool,” in *Third International Conference on Communication Systems and Networks (COMSNETS)*, 2011, pp. 1–4.

[81] “Internet Weather Map v2.8.” [Online]. Available: https://www.internetweathermap.com/. [Accessed: 10-Sep-2015].

[82] P. Mitra and G. Niemeyer, “Model-mediated Telemanipulation,” *Int. J. Rob. Res.*, vol. 27, no. 2, pp. 253–262, 2008.

[83] A. Iqbal, “Stabilization Of Delayed Teleoperation Systems Using Time Domain Passivity Control,” University of Siegen, Germany, 2007.

Appendix A

A.1 Specifications of the PHANToM Premium Device

Workspace	381 W x 267 H x 191 D mm
Footprint	330 W x 254 D mm
Range of motion	Lower arm movement pivoting at elbow
Nominal position resolution	0.03 mm
Backdrive friction	0.04 N
Maximum exertable force (nominal position)	8.5 N
Continuous exertable force (nominal position)	1.4 N
Stiffness	3.5 N mm^{-1}
Inertia (apparent mass at tip) - without encoder gimbal	< 75 g
Force feedback	x, y, z
Position sensing	x, y, z (roll, pitch, yaw optional)
Interface	Parallel port
Supported platforms	Intel-based PCs

A.2 Homogeneous Transformations and Jacobian of the PHANToM Device

Based on the DH parameters of the PHANToM device, the following homogeneous transformation matrices can be achieved

$$A_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & l_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & l_1 c_2 \\ s_2 & c_2 & 0 & l_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A_3 = \begin{bmatrix} -s_{2-3} & c_{2-3} & 0 & -l_2 s_{2-3} \\ -c_{2-3} & -s_{2-3} & 0 & -l_2 c_{2-3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_0^m = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & -l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, T_{mTool}^3 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{mTool}^m = T_0^m A_1 A_2 A_3 T_{mTool}^3 = \begin{bmatrix} c_1 & -s_1 s_3 & c_3 s_1 & x_{m1} \\ 0 & c_3 & s_3 & x_{m2} \\ -s_1 & -c_1 s_3 & c_1 c_3 & x_{m3} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where

$x_{m1} = s_1(l_1 c_2 + l_2 s_3)$

$x_{m2} = l_1 s_2 - l_2(c_3 - 1)$

$x_{m3} = l_1 c_1 c_2 - l_1 + l_2 c_1 s_3$

The Jacobian matrix J_m of PHANToM haptic interface is given below.

$$J_m(\theta) = \begin{bmatrix} j_{11} & j_{12} & j_{13} \\ j_{21} & j_{22} & j_{23} \\ j_{31} & j_{32} & j_{33} \end{bmatrix}$$

where

$$j_{11} = c_1(l_1c_2 + l_2s_3)$$
$$j_{12} = -l_1s_1s_2$$
$$j_{13} = l_2c_3s_1$$
$$j_{21} = 0$$
$$j_{22} = l_1c_2$$
$$j_{23} = l_2s_3$$
$$j_{31} = -s_1(l_1c_2 + l_2s_3)$$
$$j_{32} = -l_1c_1s_2$$
$$j_{33} = l_2c_1c_3$$

Appendix B

B.1 Specifications of the WAM Arm

Workspace	3.5 m^2
Maximum Payload	4 kg
Endtip velocity	1 m/s
Mass of robot	25.6 kg
Repeatability	50 μm
Total joint friction	3 Nm
Mechanical Stiffness	1.5 x 10^6 N/m
Control Stiffness	5000 N/m
Percent Backdrivability	> 95%
Motor Type	Neodymium Iron Boron, brushless, DC servo motors

B.2 Homogeneous Transformations and Jacobian of the WAM Arm

Based on the DH parameters of the WAM arm, the following homogeneous transformation matrices can be achieved

$$A_1 = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A_2 = \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A_3 = \begin{bmatrix} c_3 & 0 & -s_3 & a_3c_3 \\ s_3 & 0 & c_3 & a_3s_3 \\ 0 & -1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} c_4 & 0 & s_4 & -a_3c_4 \\ s_4 & 0 & -c_4 & -a_3s_4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, T_T^4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{sTool}^0 = A_1A_2A_3A_4T_{sTool}^4 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x_{s1} \\ r_{21} & r_{22} & r_{23} & x_{s2} \\ r_{31} & r_{32} & r_{33} & x_{s3} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where

$$r_{11} = -c_4(s_1s_3 - c_1c_2c_3) - c_1s_2s_4$$
$$r_{12} = -c_3s_1 - c_1c_2s_3$$
$$r_{13} = c_1c_4s_2 - s_4(s_1s_3 - c_1c_2c_3)$$
$$r_{21} = c_4(c_1s_3 + c_2c_3s_1) - s_1s_2s_4$$
$$r_{22} = c_1c_3 - c_2s_1s_3$$
$$r_{23} = s_4(c_1s_3 + c_2c_3s_1) + c_4s_1s_2$$
$$r_{31} = -c_2s_4 - c_3c_4s_2$$
$$r_{32} = s_2s_3$$
$$r_{33} = c_2c_4 - c_3s_2s_4$$
$$x_{s1} = d_3c_1s_2 - L(s_4(s_1s_3 - c_1c_2c_3) - c_1c_4s_2) - a_3s_1s_3 + a_3c_4(s_1s_3 - c_1c_2c_3) + a_3c_1c_2c_3 + a_3c_1s_2s_4$$
$$x_{s2} = L(s_4(c_1s_3 + c_2c_3s_1) + c_4s_1s_2) + a_3c_1s_3 + d_3s_1s_2 - a_3c_4(c_1s_3 + c_2c_3s_1) + a_3c_2c_3s_1 + a_3s_1s_2s_4$$
$$x_{s3} = c_3s_2s_4) + d_3c_2 - a_3c_3s_2 + a_3c_2s_4 + a_3c_3c_4s_2$$

The Jacobian matrix J_s of WAM arm is given below where j_{ik} represents the jk^{th} element of matrix J_s. For clarity, the short notations $s_i = \sin(\theta_i)$ and $c_i = \cos(\theta_i)$ $(i = 1, \ldots, 4)$ are used.

$$J_s(\theta) = \begin{bmatrix} j_{11} & j_{12} & j_{13} & j_{14} \\ j_{21} & j_{22} & j_{23} & j_{24} \\ j_{31} & j_{32} & j_{33} & j_{34} \end{bmatrix}$$

where

$$\begin{aligned}
j_{11} &= a_3c_4(c_1s_3 + c_2c_3s_1) - a_3c_1s_3 - d_3s_1s_2 - L(s_4(c_1s_3 + c_2c_3s_1) + c_4s_1s_2) - a_3c_2c_3s_1 - a_3s_1s_2s_4 \\
j_{12} &= c_1(d_3c_2 + Lc_2c_4 - a_3c_3s_2 + a_3c_2s_4 - Lc_3s_2s_4 + a_3c_3c_4s_2) \\
j_{13} &= -(c_3s_1 + c_1c_2s_3)(a_3 + Ls_4 - a_3c_4) \\
j_{14} &= a_3c_1c_4s_2 - a_3s_4(s_1s_3 - c_1c_2c_3) - L(c_4(s_1s_3 - c_1c_2c_3) + c_1s_2s_4) \\
j_{21} &= d_3c_1s_2 - L(s_4(s_1s_3 - c_1c_2c_3) - c_1c_4s_2) - a_3s_1s_3 + a_3c_4(s_1s_3 - c_1c_2c_3) + a_3c_1c_2c_3 + a_3c_1s_2s_4 \\
j_{22} &= s_1(d_3c_2 + Lc_2c_4 - a_3c_3s_2 + a_3c_2s_4 - Lc_3s_2s_4 + a_3c_3c_4s_2) \\
j_{23} &= (c_1c_3 - c_2s_1s_3)(a_3 + Ls_4 - a_3c_4) \\
j_{24} &= L(c_4(c_1s_3 + c_2c_3s_1) - s_1s_2s_4) + a_3s_4(c_1s_3 + c_2c_3s_1) + a_3c_4s_1s_2 \\
j_{31} &= 0 \\
j_{32} &= a_3c_2c_3c_4 - d_3s_2 - a_3c_2c_3 - a_3s_2s_4 - L(c_4s_2 + c_2c_3s_4) \\
j_{33} &= s_2s_3(2a_3\sin(\theta_4/2)^2 + Ls_4) \\
j_{34} &= a_3c_2c_4 - L(c_2s_4 + c_3c_4s_2) - a_3c_3s_2s_4
\end{aligned}$$

B.3 WAM Inertial Specifications and Code in the MATLAB Robotics Toolbox

```
% MDL_WAM Creates model of Barrett WAM 4DOF manipulator
%
% mdl_wam
%
% Script creates the workspace variable wam which describes the
% kinematic and dynamic characteristics of a 4-DOF WAM manipulator
% using standard DH conventions.
%
% Also define the workspace vectors:
% qz:         zero joint angle configuration, arm straight up
% qSample:    arm is at a nominal non-singular configuration
%
% Author: Muhammad Tufail, IAL, UBC Canada
%
% The following license statement is for using the Robotic Toolbox.
% Copyright (C) 1993-2014, by Peter I. Corke
%
% This file is part of The Robotics Toolbox for MATLAB (RTB).
%
% RTB is free software: you can redistribute it and/or modify
% it under the terms of the GNU Lesser General Public License as published by
% the Free Software Foundation, either version 3 of the License, or
% (at your option) any later version.
%
% RTB is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU Lesser General Public License for more details.
%
% You should have received a copy of the GNU Leser General Public License
% along with RTB.  If not, see <http://www.gnu.org/licenses/>.
%
% http://www.petercorke.com

clear L
deg = pi/180;

% From document "WAM_InertialSpecifications_AC-02.pdf"
% Link 1
L(1) = Revolute('d', 0, 'a', 0, 'alpha', -pi/2, ...
    % I is the inertia tensor = [Ixx Iyy Izz Ixy Iyz Ixz]
      'I', [134880.33e-6, 113283.69e-6, 90463.30e-6, -2130.41e-6, 685.55e-6,
-124.85e-6], ...
    % r is the center of mass = [rx ry rz]
      'r', [-4.43e-3, 121.89e-3, -0.66e-3], ...
    % m is mass of the link
      'm', 10.7677, ...
      'Jm', 0, ...
      'G',  0, ...
      'B',  0, ...
      'Tc', [0 0], ...
      % qlim are joint limits
      'qlim', [-150 150]*deg );       % joint limits
```

```
% Link 2
L(2) = Revolute('d', 0, 'a', 0, 'alpha', pi/2, ...
     'I', [21409.58e-6, 13778.75e-6, 15589.06e-6, 271.72e-6, -1819.20e-6,
24.61e-6], ...
     'r', [-2.37e-3, 31.06e-3, 15.42e-3], ...
     'm', 3.8749, ...
     'Jm', 0, ...
     'G',  0, ...
     'B',  0, ...
     'Tc', [0 0], ...
     'qlim', [-113 113]*deg );

% Link 3
L(3) = Revolute('d', 0.5500, 'a', 0.045, 'alpha', -pi/2,  ...
     'I', [59110.77e-6, 3245.50e-6, 59270.43e-6, -2496.12e-6, -17.67e-6,
7.38e-6], ...
     'r', [-38.26e-3, 207.51e-3, 0.03e-3], ...
     'm', 1.8023, ...
     'Jm', 0, ...
     'G',  0, ...
     'B',  0, ...
     'Tc', [0 0], ...
     'qlim', [-157 157]*deg );

% Link 4
L(4) = Revolute('d', 0, 'a', -0.045, 'alpha', pi/2,  ...
     'I', [18485.77e-6, 18916.58e-6, 1975.17e-6, 2.19e-6, 5.15e-6, -1608.68e-
6], ...
     'r', [10.95e-3, -0.03e-3, 140.54e-3], ...
     'm', 1.0651, ...
     'Jm', 0, ...
     'G',  0, ...
     'B',  0, ...
     'Tc', [0 0], ...
     'qlim', [-50 180]*deg);


% Link 5 (Toolplate)
L(5) = Revolute('d', 0.35, 'a', 0, 'alpha', 0,  ...
     'I', [0, 0, 0, 0, 0, 0], ...
     'r', [0, 0, 0], ...
     'm',  0, ...
     'Jm', 0, ...
     'G', 0, ...
     'B', 0, ...
     'Tc', [0 0], ...
     'qlim', [0 0]*deg);

% Define the SerialLink object
wam = SerialLink(L, 'name', 'WAM 4DOF', ...
    'manufacturer', 'Barret', 'comment', 'WAM 95, Document: D1005, 2008');

% some useful poses
%
qZero   = [0 0 0 0 0];          % zero angles, L shaped pose
```

```
qs       = [0 0 -pi/2 0 0];
qn       = [0 pi/4 pi 0 0];
qSample = [-0.0160, 1.0300, -0.0310, 1.5850 0];
clear L
```