CONTROL OF A TELEOPERATING ROBOTIC ARM SUBJECT TO TIME DELAYS

by

SARDA PANKAJ PRAKASH

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2012

To my parents Geeta and Prakash Sarda.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitutde to my thesis advisor and mentor Dr. Kamesh Subbarao who stood besides me, motivated me at all times and had faith in me. His experience and invaluable assistance helped me during the course of my research. I could not have imagined having a better advisor and mentor for my Masters thesis.

I would like to thank my committee members Dr. Panayiotis Shiakolas and Dr. Alan Bowling for their precious time, encouragement and insightful comments for improving this document. My sincere thanks also goes to the wonderful MAE staff for their support and help.

I thank my fellow labmates and good friends Mousumi Ahmed, Laura Henderson, Pavan Kumar Nuthi, Ghassan Atmeh, Carlos Tule and Alok Rege for their invaluable comments, motivation and having patience in listening to me even in the middle of night.

Last but not the least, I would like to express my deepest gratitude to my sister Preeti Sarda and my parents Prakash Sarda and Geeta Sarda to whom I owe my existence. They always motivate me to keep learning in life. Its because of them that has made me a good human being and I have been able to move in a positive direction during good and bad times through their guidance.

December 18, 2012

ABSTRACT

CONTROL OF A TELEOPERATING ROBOTIC ARM SUBJECT TO TIME DELAYS

SARDA PANKAJ PRAKASH, M.S.

The University of Texas at Arlington, 2012

Supervising Professor: Dr. Kamesh Subbarao

Robotics in Minimally Invasive Surgeries (MIS) can be explored to its full potential if there is complete transparency between the surgeon and the virtual environment. In the setup, haptic device which conveys force from the surgeon to the surgery site and vice verse acts as the master station and robotic arm at the surgery site acts as the slave robot. In surgeries carried out remotely, time delays in the communication channels can cause commanded velocity/force to overshoot before the surgeon can even pull out during a task causing temporary or permanent damage during the surgery.

Since the physical capability of the transmission channels cannot be improved beyond a certain point there will always be time delays no matter how small. Therefore the work presented attempts to minimize the effects of time delays for robot manipulators having more than single degree of freedom. This will allow more delicate and complex surgeries to be performed with increased precision, dexterity and control. The controller uses a stable Lyapunov based backstepping design with tuning functions to filter out high frequency signals and smooth out the control forces.

Neural networks will be used for estimating the rate of change of external forces and nonlinear system dynamics. This design guarantees stability and inherently adds robustness to account for unknown environment.

Simulations are carried out for a 1-DOF and 2-DOF robot manipulators and the results show a substantial improvement in the performance over the direct force application. The controller also performs reasonably well with higher time delays compared to direct force method where large delays destabilize the robot.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

1.1 Haptics and its History

The word haptic (haptikos) from Greek relates to sense of touch which originates from the verb haptesthai (meaning to contact or to touch) in [3]. Haptic Technology is a tactile feedback technology which uses the notion of sense of touch to produce forces, vibrations or motions on to the user. This mechanical stimulation can be used for research and development of control techniques by creating a virtual environment and testing the system behavior before actually implementing it. This technology has been in use since early $1950'$s during which a teleoperator was built at the Argonne National Laboratory for handling radioactive substances [4, 5]. Machines which are operated remotely by humans are called as teleoperators and the technique is called teleoperation. Teleoperation which includes haptic technology is known as haptic teleoperation. Majority of times the machines in teleoperation are either fixed or mobile robots but not limited to it.

Another earliest application of haptic technology was seen in light aircraft [6]. When no servo systems were installed in light aircraft, the pilot could feel the vibrations at the control stick due to aerodynamic forces as the aircraft approached stall. This was not seen when the servo systems were installed which lead to dangerous situation because the pilot had no clue about the aircraft stalling. This problem was overcome by applying vibrations to the control stick produced by an unbalanced rotating mass which was activated by angle of attack measurements when close to stall.

1.2 Application Areas

Haptic technology has very diverse applications some of which include remotely handling radioactive substances, in bike racing games where the user can feel the road and obstacles, in aircraft to operate the control surfaces, vibration response in cellular devices, touch pad of a laptop, remotely controlled space and underwater exploration devices [4] and in Minimally Invasive Surgeries (MIS)

Scientists and engineers have achieved a milestone when it comes to research in robotics. They have successfully developed artificial humanoid robots which can feel the environment and react to it as humans tend to do. Another area where there has been a substantial progress around recent years is in surgeries carried out by robotic arm. The first successful robot-assisted surgery was carried out in 1985 when a Puma-560 robotic arm was used in a delicate neurosurgical biopsy, a non-laparoscopic surgery [7]. The success of the first procedure led to another robot assisted laparoscopic surgery, a cholecystectomy, in 1987. A laparoscopic cholecystectomy is a procedure in which 5-10 mm diameter instruments are inserted by the surgeon through small incisions into the abdomen through trocars. In 2008, over 750,000 and in 2011 over a million patients underwent cholecystectomy [8, 9] in US in which 96% are done laparoscopically which explains the important role of robotics in these surgeries. Looking at the growing number of surgeries and the potential precision of robots, the Food and Drug Administration (FDA) had cleared the da Vinci System in 2000 for surgical procedures in [7].

1.3 Need for Haptics

Haptic devices integrated with robotics can create a technological revolution in the medical field. With the increasing number of surgeries done each year there is

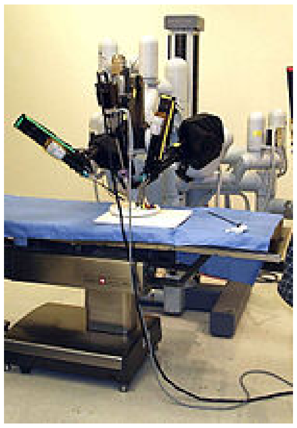

Figure 1.1. da Vinci Surgical System Manufactured by Intuitive Surgical used in Surgeries via Minimally Invasive Approach (MIS) (Image Courtesy: www.absoluteastronomy.com).

more demand of highly trained surgeons but the number of doctors graduating each year is not sufficient to meet the demands specially in developing countries. Therefore highly skilled and trained doctors flying across the globe for carrying out surgeries has become a common practice. But technological advances in haptic teleoperation have enabled highly skilled surgeons to setup a central workstation to perform operations remotely. It is because of the haptic feedback that the surgeons can now feel the environment, as if operating directly on the patient.

This technology has also helped in training students in medical institutes and recently graduated doctors. A realistic environment and a realistic/holographic organ is created for the simulated surgeries so that the surgeon gets enough practice and feel of the environment before he actually starts operating on anyone.

1.4 Literature Review

Richert, Macnab and Pieper in [1, 10, 11] developed an adaptive backstepping control with tuning functions for a 1-DOF robot-assisted endoscopic task to minimize the effects of time delay. Results were compared with H_∞ control and direct force control techniques. Richert, Beirami and Macnab in [12] used a novel neural-adaptive method, which uses an estimate of inertia matrix to supervise the training of the neural network, to control a 2-DOF 2-link flexible joint robot manipulator. The approach shows improved performance over forward dynamics and is robust under payload. Macnab, D'Eleuterio and Meng in [13] derived a robust adaptive backstepping control with weighted tuning functions to enhance the performance of the neural networks used for estimating dynamics. Results for 2-link flexible joint robot show good performance with less number of learning repetitions as compared to direct adaptive control. Wang, Tuer, Rossi, Ni and Shu in [14] investigated the advantages of haptics and effects of time delays with experiments. They proposed a latency compensation module which would consider the measure of network latency and prediction horizon to modify the input command to compensate for the time delay. This method greatly suffers from overshoots, noise effects and its performance depends on the prediction horizon.

Arioui, Kheddar and Mammar in [15] combined the wave variable theory and Smith prediction principle to minimize the degradation performance of force feedback systems in presence of large and varying time delays. This approach needs only haptic device model to be completely known. Munir and Book in [16] also used wave theory principle with modified Smith predictor, Kalman filter and an energy regulator to improve the performance of teleoperation over internet. Although the technique is robust in wave domain a mismatched plant model could destabilize the system if predicted outside the wave domain. Experiments for 2-DOF parallel link robot show improved performance. Aziminejad, Tavakoli, Patel and Moallem in [17] proposed a 4-channel architecture based on wave theory to achieve transparency close to ideal along with absolute stability when subjected to time delay. Although good in performance the implementation of this method is highly complex. 3-channel architecture is less complex to implement but trades off between transparency and stability. Shahin and Shahdi in [18, 19] propose an Linear Quadratic Gaussian (LQG) controller which attempts to minimize the disturbance inputs on the states due to time delays. But the method needs switching between the controllers for free motion/soft contact and contact with rigid environments. Also these model-based controllers are sensitive to uncertainties in modeling master/slave dynamics. In [20] they have eliminated the dependency on master/slave parameters by local Lyapunov-based nonlinear adaptive controllers. Then a robust H_∞ control is applied to minimize the effects of known constant time delays. The controller becomes quite complex when the complete model, delay and uncertainties are accounted.

Park, Cortesao and Khatib in [21, 22, 23] developed a 2-channel adaptive decoupled force controller for Puma-560 robotic manipulator and extended their work to accommodate short time delays presented in [24]. It uses Active Observer (AOB) design to achieve robustness and on-line stiffness adaptation based on force data only to navigate through changing environments. Rigorous on-line filtering and off-line analysis for tuning the stiffness estimating parameters is required for high performance. Lawrence in [25] investigated teleoperation schemes which include 2-channel position-position, position-force, passivated position-force and transparency optimized architectures by comparing stability and transparency in time delayed teleoperation. Niemeyer and Slotine in [26] investigated wave theory, passivity concepts, force reflecting systems and successfully implemented the wave theory for time delayed telemanipulation. Nohmi, Ando and Bock in [27] introduced force reflection algorithm at the master station to compensate for communication delay. Experiments reveal that operator can feel the environment more accurately if force reflection is calculated based on telemetry of force sensor and if velocity is commanded instead of position.

1.5 Thesis Outline

The thesis is divided in six chapters, first chapter being the introduction. The second chapter describes system formulation and defines each block of the system and the task to be carried out in our research. The third chapter sheds light on neural network theory and its application for controlling the robot manipulator subjected to unknown external disturbances and shows how it captures the effects of time delays indeed. Having discussed the problem in the previous chapters we work towards deriving a stable neural-adaptive backstepping control law with tuning functions in chapter 4 for 1-DOF robot manipulator with an objective to minimize the effects of time delay. The system is simulated and results are plotted against time and position. Note that the controller developed needs no switching between free space/soft contact and contact with rigid environment and also no prior training for neural networks is needed. The control law is then extended to 2-DOF robot manipulator and simulated with showing results in Chapter 5. Simulation results are compared with one more technique to validate the performance of the controller. In Chapter 6 conclusions are drawn from the results and the end of the chapter discusses the future scope of the work done.

CHAPTER 2

SYSTEM FORMULATION AND TASK DESCRIPTION

2.1 System Setup

The block diagram of a 2-channel haptic teleoperation system in [1] is shown in Figure (2.1). Surgeon and haptic device together are considered as master and the robot operating under environment acts as the slave station. The surgeon commands position/force signals through the haptic device by means of communication channels to the slave robot. Sensors and encoders mounted on the robot give its position, velocity or force information back to the surgeon through the haptic device which then completes the loop.

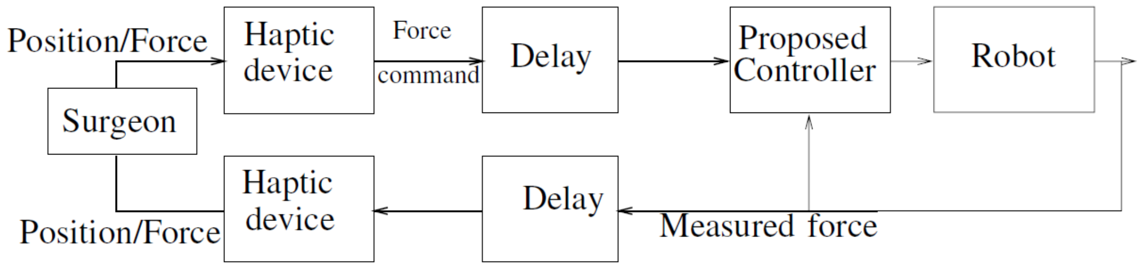

Figure 2.1. Block Diagram of the Master/Slave Teleoperator System by Macnab R. et al. in [1].

Depending on the distance between the master and the slave station there can be communication delays T (*sec*) in the forward as well as feedback loop as shown in Figure (2.1). The delay can be constant or variable. There is no way one can avoid delays in the communication channels. But a good design of closed loop system and controller can minimize the effects of this delay. The case where there is no controller either at the slave or master station is called Direct Force Application. In this thesis the performance of the proposed controller and the direct force application for 1-DOF and 2-DOF robot manipulators will be compared.

2.2 Modeling of Master Station

Human arm dynamics is difficult to model precisely in general due to human variation. They have been modeled in literature as a spring-mass-damper system [28, 29, 30] due to the fact that it closely resembles the arm behavior and that it can be simulated in real time with any common computer hardware. For research, the arm dynamics of the surgeon and the dynamics of the haptic device are modeled together as a PI filtered transfer function [1]. For 1-DOF robot manipulator constrained to move in x-direction, the haptic force f_h is given as

$$f_h(s) = \left(\frac{25s + 50}{s^2 + 25s}\right) e_v(s) \tag{2.1}$$

and

$$e_v = \dot{x} - \dot{x}_d \tag{2.2}$$

where $\dot{x}$ is the velocity of the robot felt by the surgeon at the haptic device and $\dot{x}_d$ is the desired velocity commanded by the surgeon. For a 2-DOF robot manipulator acting in x-y plane, two PI filtered transfer functions given in Equation (2.1) model the human and the haptic device in x and y direction independently. This is an assumption since all serial manipulators having more than 1-DOF have their dynamics coupled. The output from the haptic device for a 2-DOF system is given as

$$\mathbf{F_h} = \begin{bmatrix} f_{hx} \\ f_{hy} \end{bmatrix} \tag{2.3}$$

where e_{vx} and e_{vy} are velocity errors in x and y direction and

$$f_{hx}(s) = \left(\frac{25s + 50}{s^2 + 25s}\right) \; e_{vx}(s) \tag{2.4}$$

$$f_{hy}(s) = \left(\frac{25s + 50}{s^2 + 25s}\right) \; e_{vy}(s) \tag{2.5}$$

2.3 Modeling of Robot Manipulator

The Puma 560 is a widely used robotic arm due to its high precision, accuracy, repeatability, complex maneuverability and reach of its end effector. It is a serial manipulator with six degrees of freedom/joints. For research purpose however main focus is on improving the control technique by validating the approach for 1-DOF and 2-DOF robotic manipulators. The development is general enough to be extended for multi degree of freedom system. The manipulator dynamics of robot in general interacting with the environment in its standard form is given in [31] as follows

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \tau - \mathbf{J}(\mathbf{q})^T\mathbf{F_e} \tag{2.6}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the inertia matrix, $\mathbf{q} \in \mathbb{R}^{n\times 1}$ is the vector of joint angles, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ is the matrix containing coriolis and centripetal components of forces, $\mathbf{G}(\mathbf{q}) \in \mathbb{R}^{n\times 1}$ is the vector of gravitational forces, $\tau \in \mathbb{R}^{n\times 1}$ is the vector of control torques, $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the configuration dependent Jacobian matrix and $\mathbf{F_e} \in \mathbb{R}^{n\times 1}$ is the vector of environmental forces, where n is the number of joints.

In applications where the robot is interacting with the external environment it is best suited to work in the task space or so called Cartesian space. The robot dynamics in the Cartesian space can be written as

$$\mathbf{M_r}(\mathbf{q})\ddot{\mathbf{r}} + \mathbf{C_r}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{r}} + \mathbf{G_r}(\mathbf{q}) = \mathbf{F_c} - \mathbf{F_e} \tag{2.7}$$

where

$$\mathbf{M_r}(\mathbf{q}) = \mathbf{J}^{-T}(\mathbf{q})\mathbf{M}(\mathbf{q})\mathbf{J}^{-1}(\mathbf{q}) \tag{2.8}$$

$$\mathbf{C_r}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{J}^{-T}(\mathbf{q})[\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) - \mathbf{M}(\mathbf{q})J^{-1}(\mathbf{q})\dot{\mathbf{J}}(\mathbf{q})\dot{\mathbf{q}}] \tag{2.9}$$

$$\mathbf{G_r}(\mathbf{q}) = \mathbf{J}^{-T}(\mathbf{q})\mathbf{G}(\mathbf{q}) \tag{2.10}$$

$$\mathbf{F_c} = \mathbf{J}^{-T}(\mathbf{q})\tau \tag{2.11}$$

and $\mathbf{r} \in \mathbb{R}^{n\times 1}$ denotes the end-effector position and orientation in the Cartesian space.

$1 - DOF\ Robot\ Manipulator$:

The dynamic model for 1-DOF system used for our research is reduced from Equation (2.7) in Cartesian space and given as

$$m\ddot{x} + c\dot{x} = f_c - f_e \tag{2.12}$$

where x is the position of the robot, $m = 0.5kg$ is the mass of the robot, $c = 0.1Ns/m$ is the damping/friction of the robot, f_c is the control force which is same as the haptic force in direct force application and f_e is the force exerted by the environment on the robot end-effector which is measured by a sensor [11]. Figure (2.2) shows the 1-DOF robot manipulator.

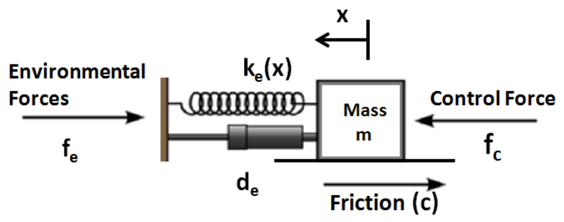

Figure 2.2. 1-DOF Robot Manipulator.

$2 - DOF\ Robot\ Manipulator$:

The dynamic model for a 2-DOF system in Cartesian space from Equation (2.7) is given as follows

$$\mathbf{M_r}(\mathbf{q})\ddot{\mathbf{r}} + \mathbf{C_r}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{r}} + \mathbf{G_r}(\mathbf{q}) = \mathbf{F_c} - \mathbf{F_e} \tag{2.13}$$

where $\mathbf{r} \in \mathbb{R}^{2\times1}$ denotes the end-effector position in Cartesian space, $\mathbf{M_r}(\mathbf{q}) \in \mathbb{R}^{2\times2}$ is the inertia matrix, $\mathbf{q} \in \mathbb{R}^{2\times1}$ is the vector of joint angles, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{2\times2}$ is the matrix containing coriolis and centripetal components of forces, $\mathbf{G}(\mathbf{q}) \in \mathbb{R}^{2\times1}$ is the vector of gravitational forces, $\mathbf{F_c} \in \mathbb{R}^{2\times1}$ is the vector of control forces, $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{2\times2}$ is the configuration dependent Jacobian matrix and $\mathbf{F_e} \in \mathbb{R}^{2\times1}$ is the vector of environmental forces. Figure 2.3 shows a 2-DOF 2 link planar robot manipulator in [2]. The robot parameters like the link lengths, inertia, coriolis and gravity terms are listed in Appendix A.

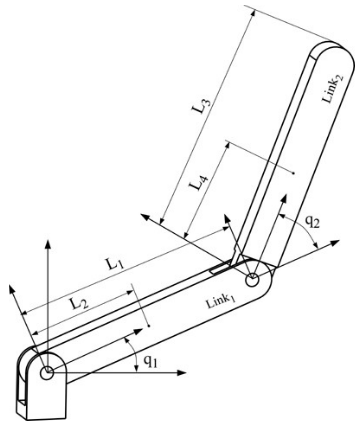

Figure 2.3. 2-DOF Robot Manipulator by Bowling A. et al. in [2].

2.4 Sensor Model

A force sensor at the tool of the robot end-effector is used to measure the external forces during the task. The sensor equation for 1-DOF robot model is given as

$$f_e = k_e(x)x + d_e\dot{x} \tag{2.14}$$

where x is the position of the tool, f_e is the environmental force measured by the sensor. $k_e(x)$ is the stiffness of the sensor and d_e is the damping in the sensor. For a 2-DOF robot model the sensor equation is given as

$$\mathbf{F_e} = \mathbf{K_e}(\mathbf{r})\mathbf{r} + \mathbf{D_e}\dot{\mathbf{r}} \tag{2.15}$$

where $\mathbf{r} \in \mathbb{R}^{2\times1}$, $\dot{\mathbf{r}} \in \mathbb{R}^{2\times1}$ is a vector of Cartesian position and velocity respectively, $\mathbf{F_e} \in \mathbb{R}^{2\times1}$ is a vector of external forces on the tool, $\mathbf{K_e}(\mathbf{r}) \in \mathbb{R}^{2\times2}$ is the task and position specific stiffness matrix and $\mathbf{D_e} \in \mathbb{R}^{2\times2}$ is the task specific damping matrix.

2.5 Task Description

With growing number of surgeries done via Minimally Invasive approach especially laparoscopic cholecystectomy, endoscopic tools are used on a regular basis to guide such surgeries which is a challenging task. Therefore we consider an endoscopic task where the robot pushes the tool in x-direction. During the task the tool enters free space from medium puncturing the tissue and will collide with a solid object probably a bone.

For 1-DOF control problem the robot starts from zero with a commanded velocity of 0.001 m/sec, punctures through a tissue at 0.0075 m from medium to free space and collides with a solid object at 0.015 m. After first collision the surgeon commands a constant force $f_h = 0.005N$ before commanding it to stop. The stiffness of the medium, free space and the solid object is modeled as

$$\begin{aligned} k_{\text{medium}} &= 0.01\ N/m \\ k_{\text{free space}} &= 0\ N/m \\ k_{\text{solid object}} &= 150\ N/m \end{aligned} \tag{2.16}$$

and a plot of stiffness force is plotted against position of the tool in Figure 2.4. The damping of the medium is $d_e = 0.5Ns/m$ and $d_e = 0Ns/m$ elsewhere [1].

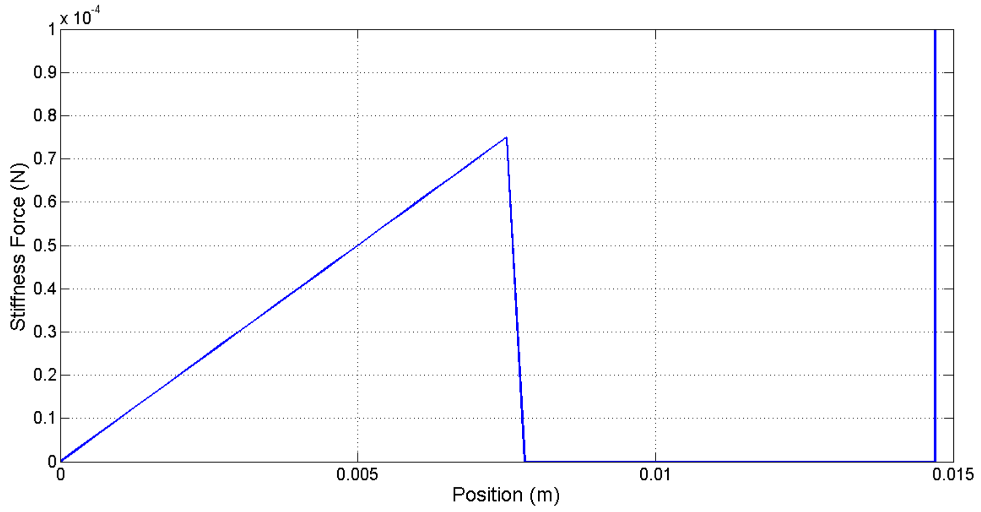

Figure 2.4. Stiffness Profile: The end-effector tries to move from left to right with constant velocity before stopping at the surface.

For 2-DOF control problem it can be seen that the workspace of the robot manipulator is a circle of radius 1.5 m as from the link lengths in Appendix A. Therefore to clearly see the comparison between the two techniques, the task problem is divided into two parts without changing the system formulation stated in Case I and Case II. Also plots for the combined task will be provided in Case III.

Case I: Puncture to Free Space from Medium at 1.1 m.

The task is carried out at constant velocity of 0.1 m/sec in x-direction and 0 m/sec in y-direction.

Case II: Collision from Free Space to Solid Object at 1 m.

After the first collision takes place the surgeon commands a constant force of 0.2 N in x-direction and 0 N in y-direction.

Case III: Puncture from Medium to Free Space at 1 m and Collision with a Hard Surface at 1.1 m.

Surgeon commands a constant velocity of 0.1 m/sec in x-direction and 0 m/sec in y-direction both in medium and free space. After first collision with the solid surface a force of 0.2 N is commanded in x-direction and 0 N is commanded in y-direction.

In all the cases, robot starts from [0.85,0] coordinates in x and y-direction respectively. The parameters of external forces acting on the robot tool in x-direction is the same as stated in 1-DOF problem. There is no external force acting in the y-direction.

2.6 Controller Design

Since the robot manipulator is performing a task in unknown environment and under time delay it is necessary to adapt and learn the uncertainties. Adaptive controllers are formulated by separating the unknown parameters from the known ones in the robot dynamic equation [32]. In adaptive approach the controller tries to learn the uncertain parameters of the system and over the time improving its accuracy with the help of information on tracking errors. If properly designed, adaptive controllers can be the best among all methods for robotic manipulators.

A 2-stage adaptive backstepping control with tuning functions will be derived in the proceeding chapters for 1-DOF and 2-DOF robot manipulator. The backstepping design allows smooth control forces to be applied. Neural networks are used for learning the model uncertainties and the change in the environmental model with time. It will be explained in detail in the next chapter with an example. The tuning functions helps in reducing the approximation and modeling error of the neural network hence increasing the performance [13, 33].

CHAPTER 3

NEURAL NETWORK APPROXIMATION

3.1 Introduction to Neural Network

The term neural network primarily evolves from the medical world referring to a network or circuit of biological neurons. In the world of control theory, they are often called as artificial neural network which consists of artificial neurons. They have been used in developing intelligent robots for mapping, localizing and navigation, which is called as Artificial Intelligence in [34]. Their successful applications are also speech recognition, image analysis and adaptive control [35]. The role of neural network in adaptive control is to learn or adapt to unknown and nonlinear function like the uncertainties in the dynamics due to friction, backlash or external disturbances with a certain degree of accuracy. A simple neural network is shown in Figure 3.1 which consists of a input layer, a single hidden layer and an output layer.

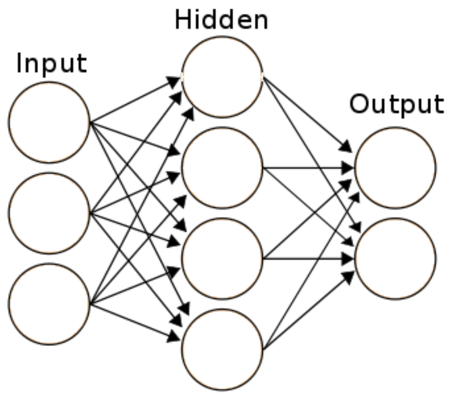

Figure 3.1. A simple model of an Artificial Neural Network.

3.2 Radial Basis Functions (RBF)

Radial basis functions are typically used in function approximations. The most commonly used functions are Gaussian, Multiquadric, Inverse Quadratic, Inverse Multiquadratic, Polyharmonic Spline and Thin Plate Spline [36]. This paper uses Gaussian functions as the radial basis functions. The advantage of using Gaussian Radial Basis Functions Network is that it can be differentiated infinite number of times and it can uniformly estimate complex functions which are continuous but not necessarily smooth. The Gaussian function is defined as

$$\phi(\bar{\mathbf{x}}) = \exp\left(-(\bar{\mathbf{x}} - \mu)^T \mathbf{R}^{-1} (\bar{\mathbf{x}} - \mu)\right) \tag{3.1}$$

where $\bar{\mathbf{x}} \in \mathbb{R}^m$ is a column vector of $\mathbf{m}$ inputs, $\mu \in \mathbb{R}^m$ is a vector of $\mathbf{m}$ expected values. $\mathbf{R} \in \mathbb{R}^{m \times m}$ is the covariance matrix associated with the RBF. To simply the analysis in this research we assume that $\mathbf{R}$ is a diagonal matrix and all the variances are identical, denoted by σ^2. The function approximation problem can be stated as follows

Given $f(x) \in \mathbb{R}$, a continuous function defined on $x \in \mathbb{R}$ and n basis functions $\mathbf{\Phi}(\bar{\mathbf{x}}) = [\phi_1 \ \phi_2 \ \dots \ \phi_n]^T$ the neural network based function approximation problem translates to finding appropriate weights $\hat{\mathbf{w}} = [w_1 \ w_2 \ \dots \ w_n]^T$ such that the square error between the actual function and the output from the neural network given by

$$\hat{y} = \mathbf{\Phi}^T(\bar{\mathbf{x}})\hat{\mathbf{w}} = \sum_{i=1}^{n} \phi_i \hat{w}_i \tag{3.2}$$

is minimized. In the process, we define the following,

$$\tilde{\mathbf{w}}_i = \mathbf{w}_i - \hat{\mathbf{w}}_i \tag{3.3}$$

where $\mathbf{w}_i$ are the ideal weights of the neural network which would produce the function $f(x)$ with a approximation error $d(x)$ with

$$|d(x)| \leq d_{max} \qquad \forall \quad x \in \mathbb{R} \tag{3.4}$$

The function that the ideal network would produce thus takes the form of

$$f(x) = \hat{y} + d(x) \tag{3.5}$$

On the other hand, $\hat{\mathbf{w}}_i$ are the estimated weights obtained from an online adaptation algorithm.

3.3 Application to Robotic Manipulators

To demonstrate the functioning of the neural network, we consider an example of a simple 2 link planar robot whose dynamical equation in the task space is given as

$$\mathbf{M_r}(\mathbf{q})\ddot{\mathbf{r}} + \mathbf{C_r}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{r}} + \mathbf{G_r}(\mathbf{q}) = \mathbf{F_c} - \mathbf{F_e} \tag{3.6}$$

where each term is described in Chapter 2. For the demonstration purpose it is said that the external disturbing force is known to us but unknown to the neural network. Let us consider two cases where

Case I :

$$\mathbf{F_e} = \mathbf{K} \begin{bmatrix} x \\ z \end{bmatrix} \tag{3.7}$$

Case II :

$$\mathbf{F_e} = \mathbf{K} \begin{bmatrix} x + \dot{x}z + 2\dot{z}x + z^2 \\ z + \dot{z}x + 2z\dot{x} + x^2 \end{bmatrix} \tag{3.8}$$

where $\mathbf{K} \in \mathbb{R}^{2\times 2}$ matrix of positive constants. Let us say that the known external forces $\mathbf{F_e}$ will be approximated by a set of linearly combined Gaussian functions $\phi(\bar{\mathbf{x}})$ and ideal weights w_i, which gives

$$\mathbf{M_r}(\mathbf{q})\ddot{\mathbf{r}} + \mathbf{C_r}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{r}} + \mathbf{G_r}(\mathbf{q}) = \mathbf{F} - \mathbf{\Phi_1^T w_1} + \mathbf{d_1} \tag{3.9}$$

where $\mathbf{d}_1$ is the approximation error. The task is to make the end-effector of the robot to follow a desired trajectory $[x_d,\ z_d]$ in the Cartesian space. Let us denote the position of the end-effector as $[x,\ z]$ which is different than the desired position due to approximation errors and unmodeled dynamics. The error due to this is given as

$$\mathbf{e} = \mathbf{r} - \mathbf{r_d} \tag{3.10}$$

where $\mathbf{r}$ is a column vector of Cartesian position. For achieving velocity control as well we choose an auxiliary error as

$$\mathbf{s} = \dot{\mathbf{e}} + \mathbf{\Lambda e} \tag{3.11}$$

where $\mathbf{\Lambda} \in \mathbb{R}^{2\times 2}$ is a positive definite matrix and proper selection can create a balance between position and velocity control. For simulating the system let us derive the control law by choosing a positive definite Lyapunov like function candidate as

$$V_1 = \frac{1}{2}\mathbf{s}^T\mathbf{s} + \frac{1}{2\beta_1}\tilde{\mathbf{w}}_1^T\tilde{\mathbf{w}}_\mathbf{1} \tag{3.12}$$

where β_1 is called the adaptation rate. Taking the time derivative gives

$$\begin{aligned} \dot{V}_1 &= \mathbf{s}^T\dot{\mathbf{s}} - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 \\ &= \mathbf{s}^T(\ddot{\mathbf{e}} + \mathbf{\Lambda}\dot{\mathbf{e}}) - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 \end{aligned} \tag{3.13}$$

Using Equation (3.9) and Equation (3.10) gives

$$\dot{V}_1 = \mathbf{s}^T[\mathbf{M_r^{-1}}(\mathbf{F} - \mathbf{\Phi_1^T w_1} + \mathbf{d_1} - \mathbf{C_r}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{r}} - \mathbf{G_r}(\mathbf{q})) - \ddot{\mathbf{r}}_\mathbf{d} + \mathbf{\Lambda}\dot{\mathbf{e}}] - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 \tag{3.14}$$

where $\mathbf{d}_1$ is the bounded approximation error. Using Equation (3.3)

$$\tilde{\mathbf{w}}_1 = \mathbf{w_1} - \hat{\mathbf{w}}_1 \tag{3.15}$$

Substituting Equation (3.15) into Equation (3.14) gives

$$\begin{aligned}
\dot{V}_1 &= \mathbf{s}^T[\mathbf{M_r^{-1}}(\mathbf{F} - \mathbf{\Phi_1^T}\hat{\mathbf{w}}_\mathbf{1} + \mathbf{d_1} - \mathbf{C_r}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{r}} - \mathbf{G_r}(\mathbf{q})) - \ddot{\mathbf{r}}_\mathbf{d} + \mathbf{\Lambda}\dot{\mathbf{e}}] \\
&\quad + \mathbf{s}^T\mathbf{M_r}^{-1}\mathbf{\Phi_1^T}\tilde{\mathbf{w}}_\mathbf{1} - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 \qquad (3.16) \\
&= \mathbf{s}^T[\mathbf{M_r^{-1}}(\mathbf{F} - \mathbf{\Phi_1^T}\hat{\mathbf{w}}_\mathbf{1} + \mathbf{d_1} - \mathbf{C_r}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{r}} - \mathbf{G_r}(\mathbf{q})) - \ddot{\mathbf{r}}_\mathbf{d} + \mathbf{\Lambda}\dot{\mathbf{e}}] \\
&\quad + \tilde{\mathbf{w}}_1^T[\mathbf{\Phi_1}\mathbf{M_r}^{-T}\mathbf{s} - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1] \qquad (3.17)
\end{aligned}$$

Leading to the choice of control law as

$$\mathbf{F} = \mathbf{\Phi_1^T}\hat{\mathbf{w}}_\mathbf{1} + \mathbf{C_r}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{r}} + \mathbf{G_r}(\mathbf{q}) - \mathbf{M_r}(\mathbf{\Lambda}\dot{\mathbf{e}} - \ddot{\mathbf{r}}_\mathbf{d}) - \mathbf{M}_r)\mathbf{G_1}\mathbf{s} \tag{3.18}$$

where $\mathbf{G}_1 = \mathbf{G}_1^T > 0$ is a positive definite gain matrix and the corresponding weight update law is chosen to be,

$$\dot{\hat{\mathbf{w}}}_1 = \beta_1\mathbf{\Phi_1}\mathbf{M_r}^{-T}\mathbf{s} \tag{3.19}$$

Substituting Equation (3.17) and Equation (3.18) in Equation (3.17) gives

$$\dot{V}_1 = -\mathbf{s^T}\mathbf{G_1}\mathbf{s} + \mathbf{s^T}\mathbf{M_r^{-1}}d_1 \tag{3.20}$$

Thus,

$$\dot{V}_1 \leq -\lambda_{min}(\mathbf{G}_1)\|\mathbf{s}\|^2 + \lambda_{max}(\mathbf{M}_r^{-1})\|\mathbf{s}\|d_{max} \tag{3.21}$$

where, $\lambda_{min}(\mathbf{G}_1)$ is the minimum eigenvalue of $\mathbf{G}_1$, $\lambda_{max}(\mathbf{M}_r^{-1})$ is the maximum eigenvalue of $\mathbf{M}_r^{-1}$ and d_{max} is the upper bound on the approximation error $\mathbf{d}_1$. The equation above can be re-arranged as

$$\begin{aligned} \dot{V}_1 &\leq -\lambda_{min}(\mathbf{G}_1)\left(\|\mathbf{s}\|^2 - \frac{\lambda_{max}(\mathbf{M}_r^{-1})}{\lambda_{min}(\mathbf{G}_1)}\|\mathbf{s}\|d_{max}\right) && (3.22) \\ &\leq -\lambda_{min}(\mathbf{G}_1)\left(\|\mathbf{s}\|^2 - 2\frac{\lambda_{max}(\mathbf{M}_r^{-1})}{2\lambda_{min}(\mathbf{G}_1)}\|\mathbf{s}\|d_{max} + \left(\frac{\lambda_{max}(\mathbf{M}_r^{-1})}{2\lambda_{min}(\mathbf{G}_1)}d_{max}\right)^2\right) \\ &\quad + \left(\frac{(\lambda_{max}(\mathbf{M}_r^{-1}))^2}{4\lambda_{min}(\mathbf{G}_1)}{d_{max}}^2\right) && (3.23) \\ \therefore \dot{V}_1 &\leq -\lambda_{min}(\mathbf{G}_1)\left(\|\mathbf{s}\| - \frac{\lambda_{max}(\mathbf{M}_r^{-1})}{2\lambda_{min}(\mathbf{G}_1)}d_{max}\right)^2 + \left(\frac{(\lambda_{max}(\mathbf{M}_r^{-1}))^2}{4\lambda_{min}(\mathbf{G}_1)}{d_{max}}^2\right) && (3.24) \end{aligned}$$

Clearly, the system is stable outside the residual set given by

$$\|\mathbf{s}\| \geq \frac{\lambda_{max}(\mathbf{M}_r^{-1})}{\lambda_{min}(\mathbf{G}_1)}d_{max} \tag{3.25}$$

Also, the system converges to the boundary of the residual set

$$\|\mathbf{s}\| = \frac{\lambda_{max}(\mathbf{M}_r^{-1})}{\lambda_{min}(\mathbf{G}_1)}d_{max} \tag{3.26}$$

asymptotically. The tracking error can be made arbitrarily small so long as the approximation error $\mathbf{d}_1$ is small and also by increasing $\lambda_{min}(\mathbf{G}_1)$.

To improve the estimation or learning performance of the neural network, the weight update is modified as in [37]

$$\dot{\hat{\mathbf{w}}}_1 = \beta_1[\mathbf{\Phi_1}\mathbf{M}^{-T}\mathbf{s} - \nu\hat{\mathbf{w}}_1] \tag{3.27}$$

where ν is a small positive leakage constant. The above update law helps in keeping the estimates bounded.

3.4 Simulation Results

For simulations we use a 2-link planar robot acting under gravity whose model is given in Appendix A. The desired trajectory to track is given by

$$\begin{aligned} x_d &= [1.1 + \sin(t)]\ m \\ z_d &= 0\ m \end{aligned} \tag{3.28}$$

where t denotes time in seconds.

Case I :

In the first case we consider the disturbance to be decoupled in x and z direction. The control law derived in the previous subsection is used and the results are plotted. In the evaluation scenario the value of gains used are $\nu = 0.001$, $\mathbf{\Lambda} = 35$, $\mathbf{G_1} = 1$, $\mathbf{K} = \text{diag}([10 \quad 10])$ and $\beta_1 = 5$.

Figure 3.2 shows that the robot closely tracks the commanded position with a maximum error of 0.0578 m and rms error of 0.0145 m in x-direction and maximum error of 0.0894 m and rms error of 0.025 m in z-direction with the error converging to zero with time. In Figure 3.3 it is seen that the neural network adapts quickly to the unknown disturbances in the system with error converging to zero.

Case II :

Due to the fact that industrial/commercial robots have more than 1-DOF and have their dynamics coupled, any disturbance force acting in x-direction also affects the position in z-direction and vice-verse. Therefore we consider a coupled nonlinear disturbance force given in Equation (3.8) to verify if the same neural network with the same set of gains used in the first case will still adapt with acceptable performance. Figure 3.4 shows that the robot closely tracks the commanded position with a maximum error of 0.13 m and rms error of 0.033 m in x-direction and maximum error of

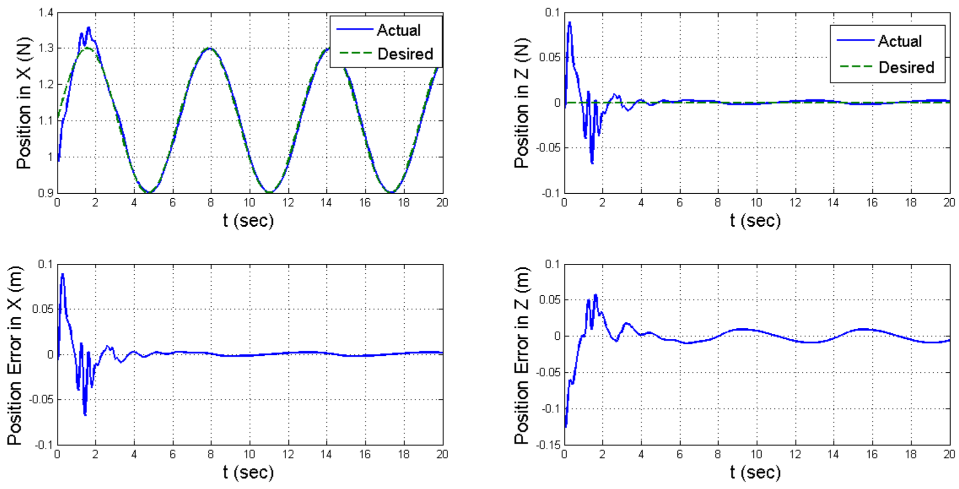

Figure 3.2. Tracking Performance in x and z-direction for *Case I*.

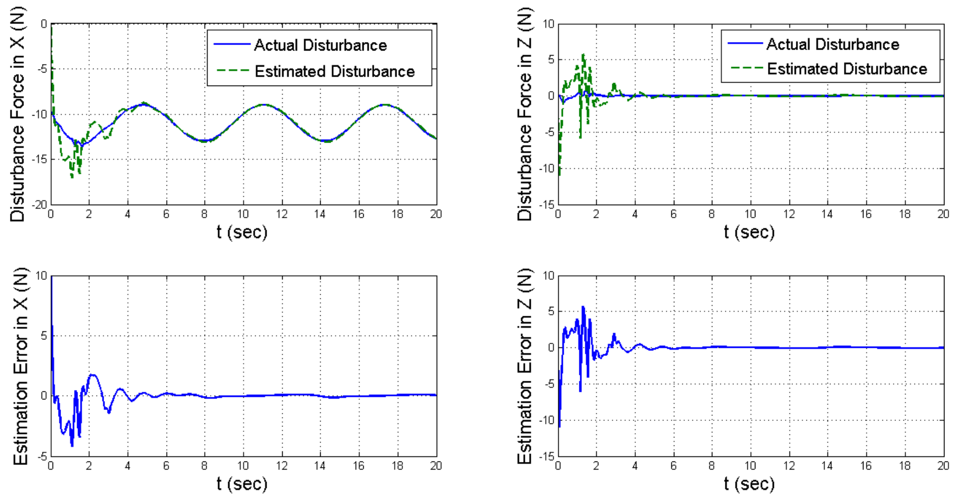

Figure 3.3. Comparison of Actual and Estimated Disturbance in x and z-direction for *Case I*.

0.0902 m and rms error of 0.027 m in z-direction with the error staying close to zero with time.

In Figure 3.5 it is seen that the neural network adapts quickly to the unknown disturbances in the system but with less accuracy than *case I*.

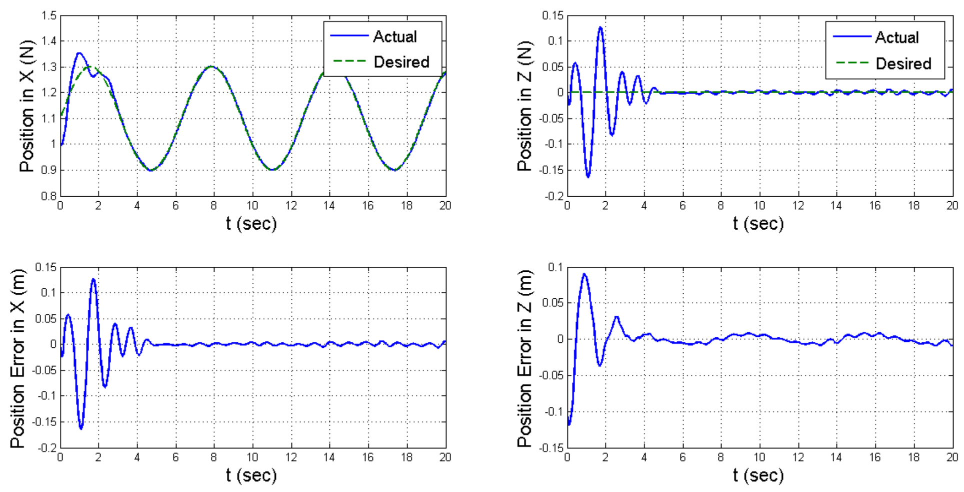

Figure 3.4. Tracking performance in x and z-direction for *Case II*.

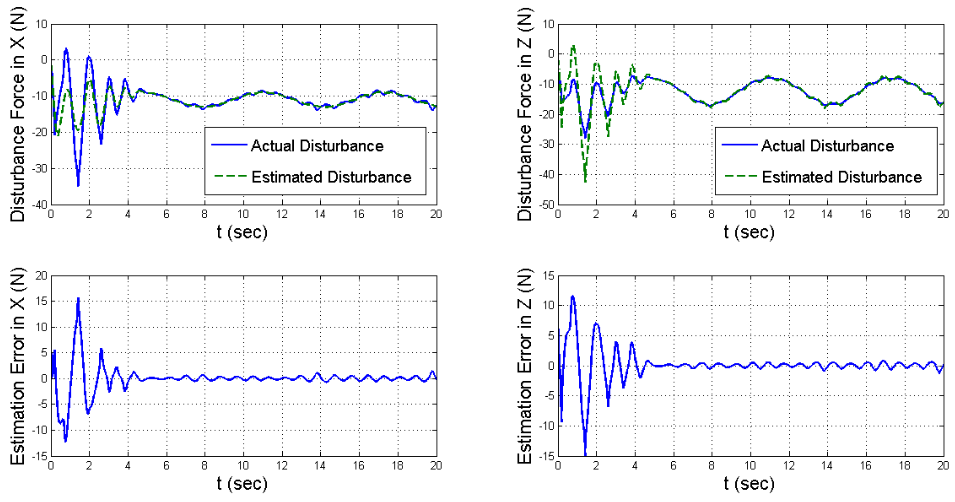

Figure 3.5. Comparison of Actual and Estimated Disturbance in x and z-direction for *Case II*.

3.5 Conclusion

Though the errors in *Case II* are slightly more than *Case I* but still the performance is acceptable. The neural network is required to coarsely estimate the disturbance rather than an accurate estimate since the adaptation rate to the output error will ensure the desired level of performance [10].

CHAPTER 4

1-DOF ROBOT MANIPULATOR CONTROL

4.1 Neural-Adaptive Backstepping Control Design with Tuning Functions

For deriving the closed loop control law, we consider the delayed haptic force $f_h(t-T)$ at the slave end as the desired force, which is operating under environmental forces $f_e(t)$ and thus defining the force error as

$$e_f(t) = f_e(t) - f_h(t-T) \tag{4.1}$$

In deriving the control law we will denote $f_h(t-T)$ as f_h and $g(t)$ as g for convenience unless required where g is any variable dependent on time. For the robot to track the velocity commanded by the surgeon along with desired force, we define the auxiliary error as

$$s = e_v + \Lambda e_f \tag{4.2}$$

where e_v is the velocity error given by

$$e_v = \dot{x}(t) - \dot{x}_d(t-T) \tag{4.3}$$

where $\dot{x}(t)$ is the Cartesian velocity of the robot end-effector, $\dot{x}_d(t-T)$ is the desired velocity at the slave robot delayed by T seconds and Λ is a positive constant which can be tuned to achieve a balance between force and velocity response depending on the application. From Chapter 2, we have

$$f_e = k_e(x)x + d_e\dot{x} \tag{4.4}$$

The time derivative of the auxiliary error is

$$\dot{s} = \ddot{x} + \Lambda(\dot{f}_e - \dot{f}_h) \tag{4.5}$$

Using Equation (4.4) in Equation (4.5) gives

$$\begin{aligned}\dot{s} &= \ddot{x} + \Lambda[\dot{k}_e(x)x + k_e(x)\dot{x} + d_e\ddot{x} - \dot{f}_h] \\ \dot{s} &= (1 + \Lambda d_e)\ddot{x} + \Lambda[\dot{k}_e(x)x + k_e(x)\dot{x}] - \Lambda\dot{f}_h\end{aligned} \tag{4.6}$$

Using Equation (2.12) gives

$$\dot{s} = (1 + \Lambda d_e)m^{-1}[f_c - f_e - c\dot{x}] + \Lambda[\dot{k}_e(x)x + k_e(x)\dot{x}] - \Lambda\dot{f}_h \tag{4.7}$$

The first neural network is used to approximate the non-linear terms

$$\dot{k}_e(x)x + k_e(x)\dot{x} - (1 + \Lambda d_e)m^{-1}(c\dot{x}) - \Lambda\dot{f}_h = \mathbf{\Phi}_1^T\mathbf{w}_1 + d_1 \tag{4.8}$$

where d_1 is a bounded approximation error as discussed in Chapter 3. Additionally we introduce another adaptive parameter, ψ to estimate

$$(1 + \Lambda d_e)m^{-1} = \psi + d_2 \tag{4.9}$$

where d_2 is also a bounded approximation error. Note, we assume that the damping d_e in Equation (4.4) is an unknown constant (slowly varying).

For deriving the control scheme and to analyze its stability we construct an "energy-like" function called the Lyapunov function and examine its time variation. The backstepping control law is designed in 2 stages. For the first stage we consider a positive-definite candidate Lyapunov function in [10] as follows

$$V_1 = \frac{1}{2}s^2 + \frac{1}{2\beta_1}\tilde{\mathbf{w}}_1^T\tilde{\mathbf{w}}_1 + \frac{1}{2\beta_2}\tilde{\psi}^2 \tag{4.10}$$

The time derivative of the function is then given as

$$\dot{V}_1 = s\dot{s} + \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\tilde{\mathbf{w}}}_1 + \frac{1}{\beta_2}\tilde{\psi}\dot{\tilde{\psi}} \tag{4.11}$$

Using the following equations

$$\begin{aligned}\tilde{\mathbf{w}}_1 &= \mathbf{w}_1 - \hat{\mathbf{w}}_1 \\ \tilde{\psi} &= \psi - \hat{\psi}\end{aligned} \tag{4.12}$$

into Equation (4.11) gives

$$\dot{V}_1 = s\dot{s} + \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T(\dot{\mathbf{w}}_1 - \dot{\hat{\mathbf{w}}}_1) + \frac{1}{\beta_2}\tilde{\psi}(\dot{\psi} - \dot{\hat{\psi}}) \tag{4.13}$$

Since $\mathbf{w}_1$ is a vector of constant weights and ψ is a constant parameter (slowly varying), $\dot{\mathbf{w}}_1 = 0$ and $\dot{\psi} = 0$. Therefore

$$\dot{V}_1 = s\dot{s} - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \frac{1}{\beta_2}\tilde{\psi}\dot{\hat{\psi}} \tag{4.14}$$

Using Equation (4.7) and Equation (4.14) gives

$$\dot{V}_1 = s[(1+\Lambda d_e)m^{-1}(f_c - f_e - c\dot{x}) + \Lambda(\dot{k}_e x + k_e\dot{x}) - \Lambda\dot{f}_h] - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \frac{1}{\beta_2}\tilde{\psi}\dot{\hat{\psi}} \tag{4.15}$$

Substituting Equations (4.8) and (4.9) into Equation (4.15) gives

$$\dot{V}_1 = s[(\psi + d_2)(f_c - f_e) + \mathbf{\Phi}_1^T\mathbf{w}_1 + d_1] - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \frac{1}{\beta_2}\tilde{\psi}\dot{\hat{\psi}} \tag{4.16}$$

$$\begin{aligned}\dot{V}_1 = s[\tilde{\psi}(f_c - f_e) + \hat{\psi}(f_c - f_e) + d_2(f_c - f_e) + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1 + \mathbf{\Phi}_1^T\hat{\mathbf{w}}_1 + d_1] \\ -\tfrac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \tfrac{1}{\beta_2}\tilde{\psi}\dot{\hat{\psi}}\end{aligned} \tag{4.17}$$

Let the desired control force (virtual control) at the robot end be denoted as α which leads to the control force error as

$$z = f_c - \alpha \tag{4.18}$$

Substituting Equation (4.18) into Equation (4.17) gives

$$\begin{aligned}\dot{V}_1 = s[\hat{\psi}(z + \alpha - f_e) + \delta_1 + \mathbf{\Phi}_1^T\hat{\mathbf{w}}_1] + \tilde{\mathbf{w}}_1^T(\mathbf{\Phi}_1 s - \tfrac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) \\ +\tilde{\psi}(s(f_c - f_e) - \tfrac{1}{\beta_2}\dot{\hat{\psi}})\end{aligned} \tag{4.19}$$

where all the uncertainties are contained in

$$\delta_1 = d_2(f_c - f_e) + d_1 \tag{4.20}$$

The weight update laws will be defined at a later stage as per the tuning function method. The first set of tuning functions are defined as

$$\begin{aligned}\tau_1 &= \mathbf{\Phi}_1 s \\ \tau_2 &= s(f_c - f_e)\end{aligned} \tag{4.21}$$

then the derivative can be written as

$$\dot{V}_1 = s[\hat{\psi}(z + \alpha - f_e) + \delta_1 + \mathbf{\Phi}_1^T \hat{\mathbf{w}}_1] + \tilde{\mathbf{w}}_1^T(\tau_1 - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \tilde{\psi}(\tau_2 - \frac{1}{\beta_2}\dot{\hat{\psi}}_2) \tag{4.22}$$

leading to the choice of virtual control as

$$\alpha = f_e - \hat{\psi}^{-1}(\mathbf{\Phi}_1^T \hat{\mathbf{w}}_1 + G_1 s) \tag{4.23}$$

where G_1 is a positive gain to stabilize the error dynamics. The derivative of the virtual control α is given as

$$\dot{\alpha} = \dot{f}_e + \hat{\psi}^{-2}\dot{\hat{\psi}}[\mathbf{\Phi}_1^T \hat{\mathbf{w}}_1 + G_1 s] - \hat{\psi}^{-1}[\frac{d}{dt}(\mathbf{\Phi}_1^T \hat{\mathbf{w}}_1) + G_1 \dot{s}] \tag{4.24}$$

A third neural network is used to estimate the change in the environmental forces f_e with respect to time given by

$$\dot{f}_e = \mathbf{\Phi}_3^T \mathbf{w}_3 + d_3 \tag{4.25}$$

where d_3 is another bounded approximation error in estimating $\dot{f}_e$. Substituting Equation (4.23) in Equation (4.22) we get

$$\dot{V}_1 = -G_1 s^2 + s\delta_1 + s\hat{\psi}z + \tilde{\mathbf{w}}_1^T(\tau_1 - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \tilde{\psi}(\tau_2 - \frac{1}{\beta_2}\dot{\hat{\psi}}) \tag{4.26}$$

The control Lyapunov function for the second stage of backstepping is chosen as

$$V_2 = V_1 + \frac{1}{2}z^2 + \frac{1}{2\beta_3}\tilde{\mathbf{w}}_3^T \tilde{\mathbf{w}}_3 \tag{4.27}$$

The time derivative is

$$\dot{V}_2 = \dot{V}_1 + z\dot{z} - \frac{1}{\beta_3}\tilde{\mathbf{w}}_3^T \dot{\hat{\mathbf{w}}}_3 \tag{4.28}$$

Substituting Equation (4.26) into Equation(4.28) gives

$$\begin{aligned}\dot{V}_2 = -G_1 s^2 + s\delta_1 + s\hat{\psi}z + \tilde{\mathbf{w}}_1^T(\tau_1 - \tfrac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \tilde{\psi}(\tau_2 - \tfrac{1}{\beta_2}\dot{\hat{\psi}}) \\ +z(\dot{f}_c - \dot{\alpha}) - \tfrac{1}{\beta_3}\tilde{\mathbf{w}}_3^T\dot{\hat{\mathbf{w}}}_3\end{aligned} \tag{4.29}$$

Substituting Equation (4.25) in Equation (4.24) gives

$$\dot{\alpha} = \mathbf{\Phi}_3^T\mathbf{w}_3 + d_3 + \hat{\psi}^{-2}\dot{\hat{\psi}}[\mathbf{\Phi}^T{}_1\hat{\mathbf{w}}_1 + G_1 s] - \psi^{-1}[\frac{d}{dt}(\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1) + G_1\dot{s}] \tag{4.30}$$

The implementable components of the derivative of the virtual control $\dot{\alpha}$ are given in $\dot{\hat{\alpha}}$ as

$$\dot{\hat{\alpha}} = \mathbf{\Phi}_3^T\hat{\mathbf{w}}_3 + \hat{\psi}^{-2}\dot{\hat{\psi}}[\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1 + G_1 s] - \psi^{-1}[\frac{d}{dt}(\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1) + G_1\dot{\hat{s}}] \tag{4.31}$$

Leading to the choice of $\dot{f}_c$ as

$$\dot{f}_c = \dot{\hat{\alpha}} - \hat{\psi}s - G_2 z \tag{4.32}$$

where G_2 is another positive gain to stabilize the error dynamics. Substituting Equation (4.32) into Equation (4.29) gives

$$\begin{aligned}\dot{V}_2 = -G_1 s^2 + s\delta_1 + s\hat{\psi}z + \tilde{\mathbf{w}}_1^T(\tau_1 - \tfrac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \tilde{\psi}(\tau_2 - \tfrac{1}{\beta_2}\dot{\hat{\psi}}) \\ +z(\dot{\hat{\alpha}} - \hat{\psi}s - G_2 z - \dot{\alpha}) - \tfrac{1}{\beta_3}\tilde{\mathbf{w}}_3^T\dot{\hat{\mathbf{w}}}_3\end{aligned} \tag{4.33}$$

$$\begin{aligned}\dot{V}_2 = -G_1 s^2 - G_2 z^2 + s\delta_1 + \tilde{\mathbf{w}}_1^T(\tau_1 - \tfrac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \tilde{\psi}(\tau_2 - \tfrac{1}{\beta_2}\dot{\hat{\psi}}) \\ -z\dot{\tilde{\alpha}} - \tfrac{1}{\beta_3}\tilde{\mathbf{w}}_3^T\dot{\hat{\mathbf{w}}}_3\end{aligned} \tag{4.34}$$

Subtracting Equation (4.30) and Equation (4.31) gives

$$\dot{\tilde{\alpha}} = \mathbf{\Phi}_3\tilde{\mathbf{w}}_3 + d_3 - \hat{\Psi}^{-1}G_1(\dot{s} - \dot{\hat{s}}) \tag{4.35}$$

Using Equation(4.7), Equation (4.8) and Equation (4.9) we can get

$$\begin{aligned}\dot{s} - \dot{\hat{s}} &= [(\psi + d_2)(f_c - f_e) - \Lambda\dot{f}_h + \mathbf{\Phi}_1^T\mathbf{w}_1 + d_1] - [\hat{\psi}(f_c - f_e) - \Lambda\dot{f}_h + \mathbf{\Phi}_1^T\hat{\mathbf{w}}_1] \\ &= \tilde{\psi}(f_c - f_e) + \delta_1 + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1\end{aligned} \tag{4.36}$$

Substituting Equation (4.36) into Equation (4.35) gives

$$\dot{\hat{\alpha}} = \mathbf{\Phi}^T{}_3\tilde{\mathbf{w}}_3 + d_3 - \hat{\psi}^{-1}G_1(\tilde{\psi}(f_c - f_e) + \delta_1 + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1) \tag{4.37}$$

Substituting Equation (4.37) into Equation (4.34) gives

$$\begin{aligned}
\dot{V}_2 &= -G_1s^2 - G_2z^2 + s\delta_1 + \tilde{\mathbf{w}}_1^T(\mathbf{\Phi}_1 s - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \tilde{\psi}(\tau_2 - \frac{1}{\beta_2}\dot{\hat{\psi}}) \\
&\quad -z[\mathbf{\Phi^T}{}_3\tilde{\mathbf{w}}_3 + d_3 - \hat{\psi}^{-1}G_1(\tilde{\psi}(f_c - f_e) + \delta_1 + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1)] - \frac{1}{\beta_3}\tilde{\mathbf{w}}_3^T\dot{\hat{\mathbf{w}}}_3 \qquad (4.38) \\
&= -G_1s^2 - G_2z^2 + s\delta_1 + z\delta_2 + \tilde{\mathbf{w}}_1^T[\mathbf{\Phi}_1(s + G_1\psi^{-T}z) - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1] + \tilde{\psi}[\tau_2 - \frac{1}{\beta_2}\dot{\hat{\psi}}] \\
&\quad +z\hat{\psi}^{-1}G_1\tilde{\psi}(f_c - f_e) + \tilde{\mathbf{w}}_3^T[\mathbf{\Phi}_3 z - \frac{1}{\beta_3}\dot{\hat{\mathbf{w}}}_3] \qquad (4.39)
\end{aligned}$$

where δ_2 is given by

$$\delta_2 = d_3 + \hat{\psi}^{-1}G_1\delta_1 \tag{4.40}$$

Thus,

$$\begin{aligned}
\dot{V}_2 &= -G_1s^2 - G_2z^2 + s\delta_1 + z\delta_2 + \tilde{\mathbf{w}}_1^T[\mathbf{\Phi}_1(s + G_1\hat{\psi}^{-T}z) - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1] \\
&\quad +\tilde{\psi}(\tau_2 + \eta_2 - \frac{1}{\beta_2}\dot{\hat{\psi}}) + \tilde{\mathbf{w}}_3^T(\mathbf{\Phi}_3 z - \frac{1}{\beta_3}\dot{\hat{\mathbf{w}}}_3) \qquad (4.41)
\end{aligned}$$

where

$$\eta_2 = z\hat{\psi}^{-1}G_1(f_c - f_e) \tag{4.42}$$

Hence choosing the weight updates as

$$\begin{aligned}
\dot{\hat{\mathbf{w}}}_1 &= \beta_1[\mathbf{\Phi}_1(s + G_1\hat{\psi}^{-T}z)] \\
\dot{\hat{\Psi}} &= \beta_2[\tau_2 + \eta_2] \\
\dot{\hat{\mathbf{w}}}_3 &= \beta_3\mathbf{\Phi}_3 z \qquad (4.43)
\end{aligned}$$

and substituting in Equation (4.41) results in

$$\dot{V}_2 = -G_1s^2 - G_2z^2 + s\delta_1 + z\delta_2 \tag{4.44}$$

Equation (4.44) can be written as

$$\dot{V}_2 = -\mu^T \mathbf{G} \mu + \mu^T \delta \tag{4.45}$$

where μ, $\mathbf{G}$ and δ are given as

$$\mu = \begin{bmatrix} s \\ z \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} G_1 & 0 \\ 0 & G_2 \end{bmatrix} \qquad \delta = \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} \tag{4.46}$$

The derivative can be upper bounded as

$$\dot{V}_2 \leq -\lambda_{min}(\mathbf{G}) \|\mu\|^2 + \|\mu\| \delta_{max} \tag{4.47}$$

where $\lambda_{min}(\mathbf{G})$ is the minimum eigenvalue of $\mathbf{G}$ and δ_{max} is the maximum approximation error. Equation (4.47) can be re-arranged as

$$\dot{V}_2 \leq -\lambda_{min}(\mathbf{G}) \left(\|\mu\|^2 - \frac{1}{\lambda_{min}(\mathbf{G})} \|\mu\| \delta_{max} \right) \tag{4.48}$$

Equation (4.48) can be written as

$$\begin{aligned} \dot{V}_2 &\leq -\lambda_{min}(\mathbf{G}) \left[\|\mu\|^2 - 2\frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \|\mu\| + \left(\frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \right)^2 - \left(\frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \right)^2 \right] \\ &\leq -\lambda_{min}(\mathbf{G}) \left(\|\mu\| - \frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \right)^2 + \frac{{\delta_{max}}^2}{4\lambda_{min}(\mathbf{G})} \end{aligned} \tag{4.49}$$

Hence clearly the system is stable outside the residual set

$$\|\mu\| \geq \frac{1}{\lambda_{min}(\mathbf{G})} \delta_{max} \tag{4.50}$$

Also the system converges to the boundary of the residual set

$$\|\mu\| = \frac{1}{\lambda_{min}(\mathbf{G})} \delta_{max} \tag{4.51}$$

asymptotically. The tracking error can be arbitrarily made small so long as the approximation error δ_{max} is small and also by increasing value of $\lambda_{min}(\mathbf{G})$.

To improve the performance of the adaptive update laws, they are augmented as in [37]

$$\begin{aligned}\dot{\hat{\mathbf{w}}}_1 &= \beta_1[\mathbf{\Phi}_1^T(s + G_1\hat{\psi}^{-T}z) - \nu_1\hat{\mathbf{w}}_1] \\ \dot{\hat{\psi}} &= \beta_2\text{proj}[\tau_2 + \eta_2 + \Upsilon(\psi_0 - \hat{\psi})] \\ \dot{\hat{\mathbf{w}}}_3 &= \beta_3[\mathbf{\Phi}_3^T z - \nu_3\hat{\mathbf{w}}_3]\end{aligned} \tag{4.52}$$

where ν_1, ν_3 are positive leakage constants and Υ is another tuning parameter which ranges between $[0,\ 1]$, ψ_0 is the estimate of the inertia/mass which is calculated a prior [12]. The Υ associated term acts as a supervisory to avoid the projection limits. The projection operator

$$\text{proj}[a] = \begin{cases} 0, & \hat{\Psi} < \|\Psi\|_{\min} \quad \text{and} \quad a < 0 \\ a, & \text{otherwise} \end{cases} \tag{4.53}$$

ensures that $\hat{\Psi}$ is invertible. All of the above modifications to the update laws help in keeping the estimates bounded.

4.2 Simulation Results

The control law derived in previous section is simulated with variable step Runge-Kutta method (ode45) for solving differential equations numerically. The time delay in the forward loop and the feedback loop is set to 0.1 *sec* (100 *ms*). Total time delay in the loop is 0.2 *sec* (200 *ms*) which is a good estimate of the delay appearing in the communication channels.

The control gains used in the simulations are $\Lambda = 1$, $G_1 = 2$, $G_2 = 20$, $\nu_1 = \nu_2 = \nu_3 = 0.1$ and $\beta_1 = \beta_2 = \beta_3 = 0.1$. Gaussian functions with equidistant means are selected. Velocity tracking by adaptive backstepping method is very good as compared to direct force application which can be seen from Figure 4.1. There is no velocity overshoot seen at the start for the proposed method and the magnitudes of the velocity overshoot at the puncture, which happens at 0.0075 m, with respect to the desired value of 0.001 m/sec are 0.000472 m/sec and 0.00026 m/sec for direct force application and adaptive backstepping method respectively. This means that there is approximately 44% reduction in the velocity overshoot with the proposed technique.

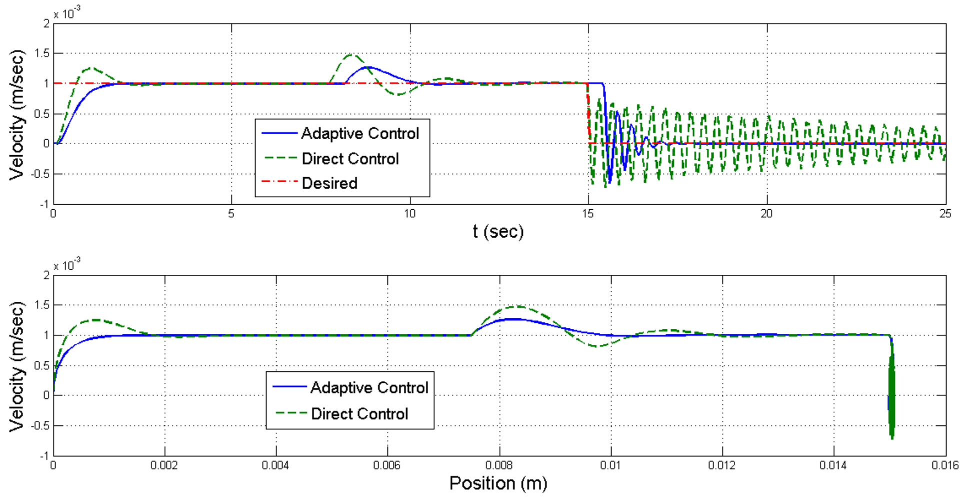

Figure 4.1. Velocity Tracking Comparison.

The values of velocity overshoots for the collision case, which happens at 0.015 m, with reference to the desired velocity of 0 m/sec are -0.72 mm/sec and -0.658 mm/sec for the direct force application and proposed method respectively, which means there is approximately a 9% reduction in the velocity overshoot at the collision in the proposed method. Also it can be seen that the settling time is much faster in case of adaptive backstepping approach as compared to direct force application.

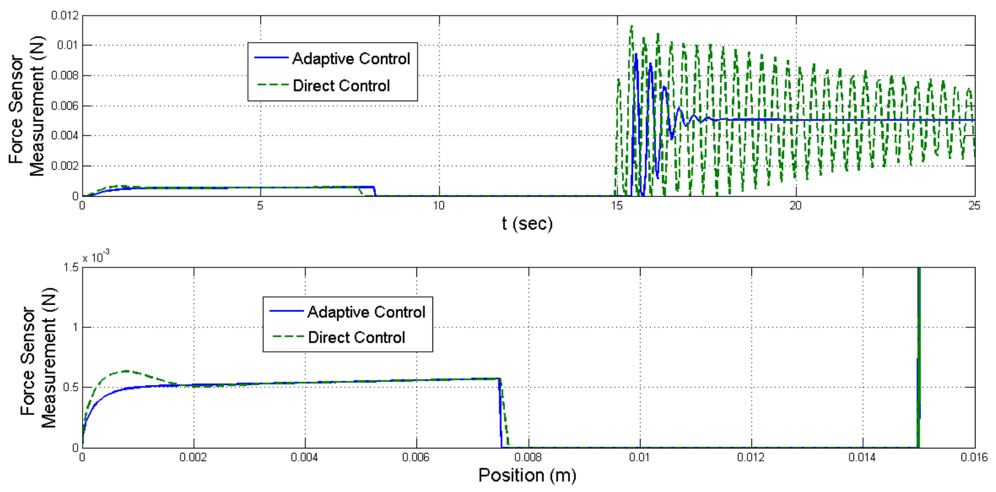

Figure 4.2. Comparison of Force Measured by the Sensor at the End-Effector.

Figure 4.2 compares the force measured by the sensor at the robot end-effector. The force measured at the puncture in both the cases are the same whose magnitude is $5.72 \times 10^{-4}N$. This can be explained as, since in both cases the velocity at the time of puncture is same and the properties of the environment do not change for both techniques the measured force remains same in both the cases.

The initial impact with the solid surface creates a force of $7.78 \times 10^{-3}N$ and $9.42 \times 10^{-3}N$ and in all a maximum force of $0.0113N$ and $9.42 \times 10^{-3}N$ with direct force application and proposed method respectively. This means that the initial value of force measured for the adaptive backstepping technique is more than that of the initial value of force measured in direct force application but is less than the maximum force measured overall. There is approximately 16% reduction in the maximum force measured in the adaptive backstepping technique with reduced oscillations at the collision. This is a very significant achievement since sensors are costly and repeated collisions may damage the sensors.

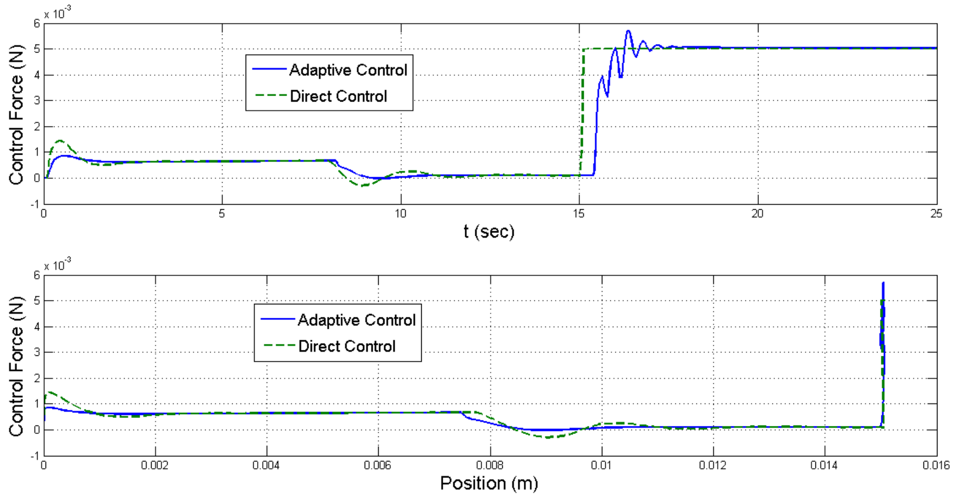

Figure 4.3. Comparison of Control Forces.

Figure 4.3 compares the control forces required to perform the task. It is important that the control forces remain smooth during the surgery which can be seen in the adaptive backstepping technique. Figure 4.4 shows the plot of weights of the first neural network given in Equation (4.8).

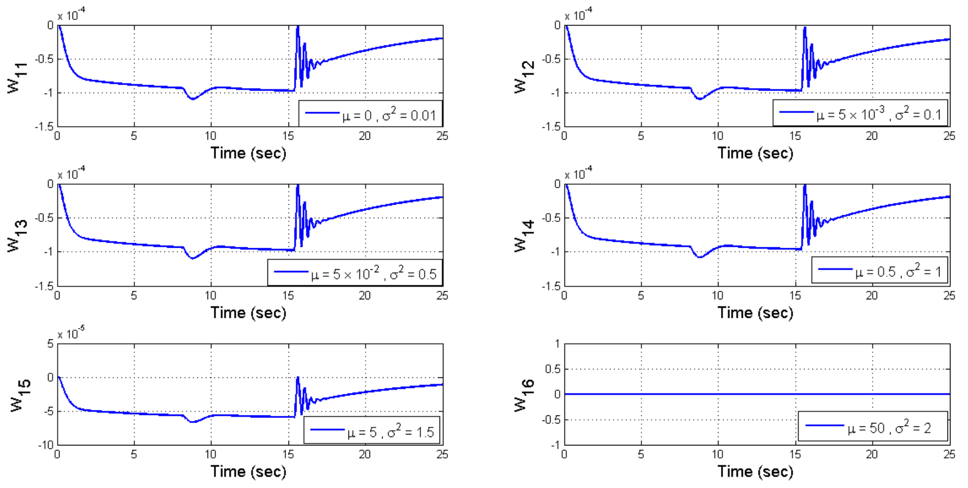

Figure 4.4. Weights of the Neural Network Estimating Nonlinear Terms in Equation (4.8).

Figure 4.5 shows the plot of the adaptive parameter which is used to estimate the terms in Equation (4.9). Figure 4.6 shows a plot of the weights of the third neural network which is used to estimate the change in environmental forces with time given by Equation (4.25).

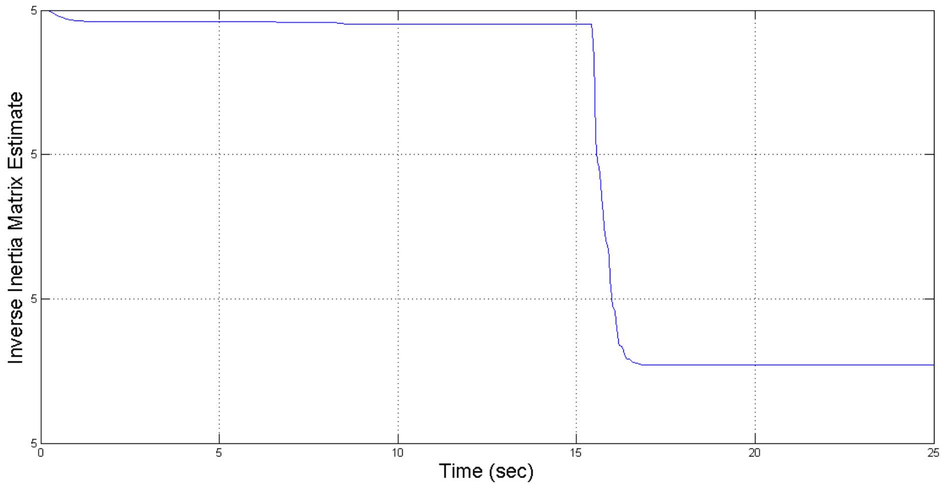

Figure 4.5. Adaptive Parameter Estimates given by Equation (4.9).

It can be seen that the Gaussian functions with their centers placed far away in the lattice have very less influence on the output of the neural network.

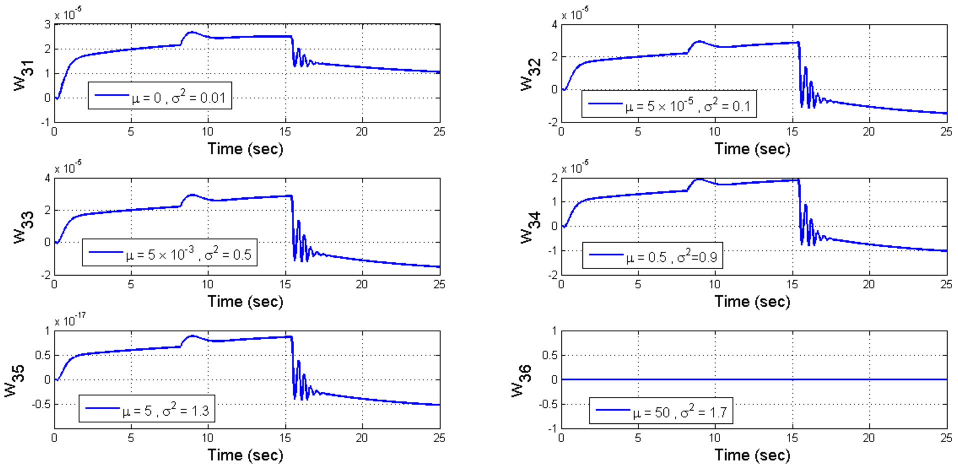

Figure 4.6. Weights of the Neural Network Estimating the derivative of the Environmental Forces as given in Equation (4.25).

CHAPTER 5

2-DOF ROBOT MANIPULATOR CONTROL

5.1 Neural-Adaptive Backstepping Control Design with Tuning Functions

For deriving the closed loop control law, we consider the delayed haptic force $\mathbf{F}_h(t-T)$ at the robot as the desired force, which is operating under environmental forces $\mathbf{F}_e(t)$ and thus defining the force error $\mathbf{e}_f$ as

$$\mathbf{e}_f = \mathbf{F}_e(t) - \mathbf{F}_h(t-T) \tag{5.1}$$

To track a desired velocity commanded by the surgeon along with force, we define the auxiliary error as

$$\mathbf{s} = \mathbf{e}_v + \mathbf{\Lambda}\mathbf{e}_f \tag{5.2}$$

where $\mathbf{e}_v$ is the velocity error given by

$$\mathbf{e}_v = \dot{\mathbf{r}}(t) - \dot{\mathbf{r}}_d(t-T) \tag{5.3}$$

where $\dot{\mathbf{r}}(t) \in \mathbb{R}^{(2\times1)}$ is the velocity of the robot end-effector in Cartesian space. $\mathbf{\Lambda} \in \mathbb{R}^{(2\times2)}$ is a positive definite matrix which can be tuned to achieve either force/velocity control depending on the application and $\dot{\mathbf{r}}_d(t-T)$ is the delayed velocity in Cartesian space commanded by surgeon to the slave robot. In deriving the control law we will denote $\mathbf{F}_h(t-T)$ as $\mathbf{F}_h$ and $g(t)$ as g for convenience unless required, where g is any variable depending on time. From Chapter 2, we have

$$\mathbf{F}_e = \mathbf{K}_e(\mathbf{r})\mathbf{r} + \mathbf{D}_e\dot{\mathbf{r}} \tag{5.4}$$

The time derivative of the auxiliary error is

$$\dot{\mathbf{s}} = \ddot{\mathbf{r}} + \mathbf{\Lambda}(\dot{\mathbf{F}}_e - \dot{\mathbf{F}}_h) \tag{5.5}$$

Substituting Equation (5.4) in Equation (5.5) gives

$$\begin{aligned}\dot{\mathbf{s}} &= \ddot{\mathbf{r}} + \mathbf{\Lambda}(\dot{\mathbf{K}}_e(\mathbf{r})\mathbf{r} + \mathbf{K}_e(\mathbf{r})\dot{\mathbf{r}} + \mathbf{D}_e\ddot{\mathbf{r}} - \dot{\mathbf{F}}_h) \\ &= (\mathbf{I} + \mathbf{\Lambda}\mathbf{D}_e)\ddot{\mathbf{r}} + \mathbf{\Lambda}(\dot{\mathbf{K}}_e(\mathbf{r})\mathbf{r} + \mathbf{K}_e(\mathbf{r})\dot{\mathbf{r}}) - \mathbf{\Lambda}\dot{\mathbf{F}}_h \qquad (5.6)\end{aligned}$$

Using Equation (2.13) and Equation (5.6) gives

$$\dot{\mathbf{s}} = (\mathbf{I} + \mathbf{\Lambda}\mathbf{D}_e)\mathbf{M}_r{}^{-1}(\mathbf{F}_c - \mathbf{F}_e - \mathbf{B}_r\dot{\mathbf{r}} - \mathbf{G}_r) + \mathbf{\Lambda}(\dot{\mathbf{K}}_e(\mathbf{r})\mathbf{r} + \mathbf{K}_e(\mathbf{r})\dot{\mathbf{r}}) - \mathbf{\Lambda}\dot{\mathbf{F}}_h \qquad (5.7)$$

The first neural network is used to approximate the non-linear terms

$$\begin{aligned}\dot{\mathbf{K}}_e(\mathbf{r})\mathbf{r} + \mathbf{K}_e(\mathbf{r})\dot{\mathbf{r}} - (\mathbf{I} + \mathbf{\Lambda}\mathbf{D}_e)\mathbf{M}_r^{-1}[\mathbf{B}_r\dot{\mathbf{r}} + \mathbf{G}_r] - \mathbf{\Lambda}\dot{\mathbf{F}}_h &= \begin{bmatrix} \mathbf{\Phi}_{11}^T\mathbf{w}_{11} + d_{11} \\ \mathbf{\Phi}_{22}^T\mathbf{w}_{22} + d_{22} \end{bmatrix} \qquad (5.8) \\ &= \mathbf{\Phi}_1^T\mathbf{w}_1 + \mathbf{d}_1\end{aligned}$$

where $\mathbf{d_1}$ is a bounded approximation as discussed in the Chapter 3. Also, $\mathbf{\Phi}_1^T = \begin{bmatrix} \mathbf{\Phi_{11}^T} & \vdots & 0 \\ \dots & & \dots \\ 0 & \vdots & \mathbf{\Phi_{22}^T} \end{bmatrix}$, $\mathbf{w}_1 = \begin{bmatrix}\mathbf{w}_{11}^T & \mathbf{w}_{22}^T\end{bmatrix}^T$.

Additionally we need another adaptive parameter to estimate

$$(\mathbf{I} + \mathbf{\Lambda}\mathbf{D}_e)\mathbf{M}_r^{-1} = \gamma \begin{bmatrix} \mathbf{w}_2 + \frac{D_{21}}{\gamma} & \mathbf{w}_3 + \frac{D_{31}}{\gamma} \\ \mathbf{w}_4 + \frac{D_{41}}{\gamma} & \mathbf{w}_5 + \frac{D_{51}}{\gamma} \end{bmatrix} = \gamma\mathbf{W} + \mathbf{D} \qquad (5.9)$$

where $\mathbf{\Psi}$ can be written as $\gamma\mathbf{W}$ and $\mathbf{D} = [D_{21}\ D_{31}\ D_{41}\ D_{51}]^T$. For deriving the control scheme and to analyze its stability we construct a "energy-like" function following the Lyapunov function approach and examine its time variation. For the first stage of backstepping design we consider a positive-definite candidate Lyapunov function as follows

$$V_1 = \frac{1}{2}\mathbf{s}^T\mathbf{P}\mathbf{s} + \frac{1}{2\beta_1}\tilde{\mathbf{w}}_1^T\tilde{\mathbf{w}}_1 + \frac{1}{2}\sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i^2 \qquad (5.10)$$

where $\mathbf{P} = \mathbf{P}^T > 0$ is a positive definite matrix. The time derivative of the function is then given as

$$\dot{V}_1 = \mathbf{s}^T\mathbf{P}\dot{\mathbf{s}} + \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\tilde{\mathbf{w}}}_1 + \sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i\dot{\tilde{\mathbf{w}}}_i \tag{5.11}$$

Using the following equations

$$\begin{aligned} \tilde{\mathbf{w}}_1 &= \mathbf{w}_1 - \hat{\mathbf{w}}_1 \\ \tilde{\mathbf{\Psi}} &= \mathbf{\Psi} - \hat{\mathbf{\Psi}} \end{aligned} \tag{5.12}$$

into Equation (5.11) gives

$$\begin{aligned} \dot{V}_1 &= \mathbf{s}^T\mathbf{P}\dot{\mathbf{s}} + \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T[\dot{\mathbf{w}}_1 - \dot{\hat{\mathbf{w}}}_1] + \sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i[\dot{\mathbf{w}}_i - \dot{\hat{\mathbf{w}}}_i] \\ &= \mathbf{s}^T\mathbf{P}\dot{\mathbf{s}} - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i\dot{\hat{\mathbf{w}}}_i \end{aligned} \tag{5.13}$$

Substituting Equation (5.7) in Equation (5.13) gives

$$\begin{aligned} \dot{V}_1 &= \mathbf{s}^T\mathbf{P}[(\mathbf{I} + \mathbf{\Lambda}\mathbf{D}_e)\mathbf{M}_r{}^{-1}(\mathbf{F}_c - \mathbf{F}_e - \mathbf{B}_r\dot{\mathbf{r}} - \mathbf{G}_r) + \mathbf{\Lambda}(\dot{\mathbf{K}}_e(\mathbf{r})\mathbf{r} + \mathbf{K}_e(\mathbf{r})\dot{\mathbf{r}}) - \mathbf{\Lambda}\dot{\mathbf{F}}_h] \\ &\quad -\frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i\dot{\hat{\mathbf{w}}}_i \end{aligned} \tag{5.14}$$

Substituting Equation (5.8) and Equation (5.9) into Equation (5.14) gives

$$\dot{V}_1 = \mathbf{s}^T\mathbf{P}\left[(\mathbf{\Psi} + \mathbf{D})(\mathbf{F}_c - \mathbf{F}_e) + \mathbf{\Phi}_1^T\mathbf{w}_1 + \mathbf{d}_1\right] - \frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i\dot{\hat{\mathbf{w}}}_i \tag{5.15}$$

$$\begin{aligned} &= \mathbf{s}^T\mathbf{P}\left[\gamma\tilde{\mathbf{W}}(\mathbf{F}_c - \mathbf{F}_e) + \gamma\hat{\mathbf{W}}(\mathbf{F}_c - \mathbf{F}_e) + \mathbf{D}(\mathbf{F}_c - \mathbf{F}_e) + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1 + \mathbf{\Phi}_1^T\hat{\mathbf{w}}_1 + \mathbf{d}_1\right] \\ &\quad -\frac{1}{\beta_1}\tilde{\mathbf{w}}_1^T\dot{\hat{\mathbf{w}}}_1 - \sum_{i=2}^{5}\frac{1}{\beta_i}\tilde{\mathbf{w}}_i\dot{\hat{\mathbf{w}}}_i \end{aligned} \tag{5.16}$$

Let the desired control force (virtual control) at the robot end be denoted as α which leads to the control force error as

$$\mathbf{z} = \mathbf{F}_c - \alpha \tag{5.17}$$

Substituting Equation (5.17) into Equation (5.16) gives

$$\begin{aligned}\dot{V}_1 &= \mathbf{s}^T\mathbf{P}\left[\hat{\mathbf{\Psi}}(\mathbf{z}+\alpha-\mathbf{F}_e)+\delta_1+\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1-\mathbf{\Lambda}\dot{\mathbf{F}}_h\right]+\tilde{\mathbf{w}}_1^T(\mathbf{\Phi}_1\mathbf{P}\mathbf{s}-\frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1)\\ &+\left(\gamma\mathbf{s}^T\mathbf{P}\tilde{\mathbf{W}}(\mathbf{F}_c-\mathbf{F}_e)-\sum_{i=2}^{5}\frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i\tilde{\mathbf{w}}_i\right)\end{aligned}\tag{5.18}$$

Notice that $\gamma\mathbf{s}^T\mathbf{P}\tilde{\mathbf{W}}(\mathbf{F}_c-\mathbf{F}_e)=\gamma\sum_{i=2}^{5}\varepsilon_{i1}\tilde{\mathbf{w}}_i$.
Also all other uncertainties are contained in

$$\delta_1=\mathbf{D}(\mathbf{F}_c-\mathbf{F}_e)+\mathbf{d}_1\tag{5.19}$$

The weight update laws will be defined at a later stage as per the tuning function method. The first set of tuning functions is defined as

$$\begin{aligned}\tau_{11} &= \mathbf{\Phi}_1\mathbf{P}\mathbf{s}\\ \tau_{i1} &= \gamma\varepsilon_{i1}\quad i\ =2,\ldots,5\end{aligned}\tag{5.20}$$

then the derivative can be written as

$$\dot{V}_1=\mathbf{s}^T\mathbf{P}[\hat{\mathbf{\Psi}}(\mathbf{z}+\alpha-\mathbf{F}_e)+\delta_1+\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1]+\tilde{\mathbf{w}}_1^T(\tau_{11}-\frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1)+\sum_{i=2}^{5}\tilde{\mathbf{w}}_i^T(\tau_{i1}-\frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i)\tag{5.21}$$

leading to the choice of virtual control as

$$\alpha=\mathbf{F}_e-\hat{\mathbf{\Psi}}^{-1}(\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1+\mathbf{G}_1\mathbf{s})\tag{5.22}$$

where $\mathbf{G}_1=\mathbf{G}_1^T>0$ is a positive definite gain matrix selected for stabilizing the error dynamics. Substituting above equation we get

$$\dot{V}_1=-\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s}+\mathbf{s}^T\mathbf{P}\delta_1+\mathbf{s}^T\mathbf{P}\hat{\mathbf{\Psi}}\mathbf{z}+\tilde{\mathbf{w}}_1^T(\tau_{11}-\frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1)+\sum_{i=2}^{5}\tilde{\mathbf{w}}_i(\tau_{\mathbf{i1}}-\frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i)\tag{5.23}$$

The derivative of the virtual control α is given as

$$\dot{\alpha}=\dot{\mathbf{F}}_e+\hat{\mathbf{\Psi}}^{-1}\dot{\hat{\mathbf{\Psi}}}\hat{\mathbf{\Psi}}^{-1}\left[\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1+\mathbf{G}_1\mathbf{s}\right]-\hat{\mathbf{\Psi}}^{-1}\left[\frac{d}{dt}(\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1)+\mathbf{G}_1\dot{\mathbf{s}}\right]\tag{5.24}$$

A third neural network is used to estimate the non-linear dynamics of the environment $\mathbf{F_e}$ given by

$$\dot{\mathbf{F}}_e = \mathbf{\Phi}_6^T \mathbf{w}_6 + \mathbf{d}_6 \tag{5.25}$$

where, $\mathbf{\Phi}_6^T = \begin{bmatrix} \mathbf{\Phi_{16}^T} & \vdots & 0 \\ \dots & & \dots \\ 0 & \vdots & \mathbf{\Phi_{26}^T} \end{bmatrix}$, $\mathbf{w}_6 = \begin{bmatrix} \mathbf{w}_{16}^T & \mathbf{w}_{26}^T \end{bmatrix}^T$.

The control Lyapunov function for the second stage is chosen as

$$V_2 = V_1 + \frac{1}{2}\mathbf{z}^T\mathbf{z} + \frac{1}{2\beta_6}\tilde{\mathbf{w}}_6^T \tilde{\mathbf{w}}_6 \tag{5.26}$$

The time derivative is

$$\dot{V}_2 = \dot{V}_1 + \mathbf{z}^T\dot{\mathbf{z}} - \frac{1}{\beta_6}\tilde{\mathbf{w}}_6^T \dot{\hat{\mathbf{w}}}_6 \tag{5.27}$$

Substituting Equation (5.23) into Equation (5.27) gives

$$\begin{aligned} \dot{V}_2 \quad = \quad & -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} + \mathbf{s}^T\mathbf{P}\delta_1 + \mathbf{s}^T\mathbf{P}\hat{\mathbf{\Psi}}\mathbf{z} + \tilde{\mathbf{w}}_1^T(\tau_{11} - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \sum_{i=2}^{5} \tilde{\mathbf{w}}_i(\tau_{\mathbf{i1}} - \frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i) \\ & +\mathbf{z}^T(\dot{\mathbf{F}}_c - \dot{\alpha}) - \frac{1}{\beta_6}\tilde{\mathbf{w}}_6^T \dot{\hat{\mathbf{w}}}_6 \end{aligned} \tag{5.28}$$

Substituting Equation (5.25) in Equation (5.24) gives

$$\dot{\alpha} = \mathbf{\Phi}_6^T\mathbf{w}_6 + \mathbf{d}_6 + \hat{\mathbf{\Psi}}^{-1}\dot{\hat{\mathbf{\Psi}}}\hat{\mathbf{\Psi}}^{-1}\left[\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1 + \mathbf{G}_1\mathbf{s}\right] - \hat{\mathbf{\Psi}}^{-1}\left[\frac{d}{dt}(\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1) + \mathbf{G}_1\dot{\mathbf{s}}\right] \tag{5.29}$$

The implementable components of the derivative of the virtual control $\dot{\alpha}$ are given in $\dot{\hat{\alpha}}$ which is given as

$$\dot{\hat{\alpha}} = \mathbf{\Phi}_6^T\hat{\mathbf{w}}_6 + \hat{\mathbf{\Psi}}^{-1}\dot{\hat{\mathbf{\Psi}}}\hat{\mathbf{\Psi}}^{-1}\left[\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1 + \mathbf{G}_1\mathbf{s}\right] - \hat{\mathbf{\Psi}}^{-1}\left[\frac{d}{dt}(\mathbf{\Phi}_1^T\hat{\mathbf{w}}_1) + \mathbf{G}_1\dot{\hat{\mathbf{s}}}\right] \tag{5.30}$$

Leading to the choice of $\dot{\mathbf{F}}_c$ as

$$\dot{\mathbf{F}}_c = \dot{\hat{\alpha}} - \hat{\mathbf{\Psi}}^T\mathbf{P}^T\mathbf{s} - \mathbf{G}_2\mathbf{z} \tag{5.31}$$

where $\mathbf{G}_2 = \mathbf{G}_2^T > 0$ is a positive definite gain matrix chosen to stabilize the auxiliary error dynamics. Substituting Equation (5.31) into Equation (5.28) gives

$$\begin{aligned}\dot{V}_2 &= -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} + \mathbf{s}^T\mathbf{P}\delta_1 + \mathbf{s}^T\mathbf{P}\hat{\mathbf{\Psi}}\mathbf{z} + \tilde{\mathbf{w}}_1^T(\tau_{11} - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \sum_{i=2}^{5}\tilde{\mathbf{w}}_i(\tau_{i1} - \frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i) \\ &\quad + \mathbf{z}^T(\dot{\hat{\alpha}} - \hat{\mathbf{\Psi}}\mathbf{P}\mathbf{s} - \mathbf{G}_2\mathbf{z} - \dot{\alpha}) - \frac{1}{\beta_6}\tilde{\mathbf{w}}_6^T\dot{\hat{\mathbf{w}}}_6 \qquad (5.32)\\ &= -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} - \mathbf{z}^T\mathbf{G}_2\mathbf{z} + \mathbf{s}^T\mathbf{P}\delta_1 + \tilde{\mathbf{w}}_1(\tau_{11} - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \sum_{i=2}^{5}\tilde{\mathbf{w}}_i^T(\tau_{i1} - \frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i) \\ &\quad - \mathbf{z}^T\dot{\tilde{\alpha}} - \frac{1}{\beta_6}\tilde{\mathbf{w}}_6^T\dot{\hat{\mathbf{w}}}_6 \qquad (5.33)\end{aligned}$$

Subtracting Equation (5.29) and Equation (5.30) gives

$$\dot{\tilde{\alpha}} = \mathbf{\Phi}_6^T\tilde{\mathbf{w}}_6 + \mathbf{d}_6 - \hat{\mathbf{\Psi}}^{-1}\mathbf{G}_1(\dot{\mathbf{s}} - \dot{\hat{\mathbf{s}}}) \qquad (5.34)$$

Using Equation (5.7) we can get

$$\begin{aligned}\dot{\mathbf{s}} - \dot{\hat{\mathbf{s}}} &= [(\mathbf{\Psi} + \mathbf{D})(\mathbf{F}_c - \mathbf{F}_e) - \mathbf{\Lambda}\dot{\mathbf{F}}_h + \mathbf{\Phi}_1^T\mathbf{w}_1 + \mathbf{d}_1] - [\hat{\mathbf{\Psi}}(\mathbf{F}_c - \mathbf{F}_e) - \mathbf{\Lambda}\dot{\mathbf{F}}_h + \mathbf{\Phi}_1^T\hat{\mathbf{w}}_1] \\ &= \tilde{\mathbf{\Psi}}(\mathbf{F}_c - \mathbf{F}_e) + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1 + \delta_1 \qquad (5.35)\end{aligned}$$

Substituting Equation (5.35) into Equation (5.34) gives

$$\dot{\tilde{\alpha}} = \mathbf{\Phi}_6^T\tilde{\mathbf{w}}_6 + \mathbf{d}_6 - \hat{\mathbf{\Psi}}^{-1}\mathbf{G}_1(\tilde{\mathbf{\Psi}}(\mathbf{F}_c - \mathbf{F}_e) + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1 + \delta_1) \qquad (5.36)$$

Substituting Equation (5.36) into Equation (5.33) gives

$$\begin{aligned}\dot{V}_2 &= -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} - \mathbf{z}^T\mathbf{G}_2\mathbf{z} + \mathbf{s}^T\mathbf{P}\delta_1 + \tilde{\mathbf{w}}_1^T(\mathbf{\Phi}_1\mathbf{P}\mathbf{s} - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1) + \sum_{i=2}^{5}\tilde{\mathbf{w}}_i(\tau_{i1} - \frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i) \\ &\quad - \mathbf{z}^T[\mathbf{\Phi}_6^T\tilde{\mathbf{w}}_6 + \mathbf{d}_6 - \hat{\mathbf{\Psi}}^{-1}\mathbf{G}_1(\tilde{\mathbf{\Psi}}(\mathbf{F}_c - \mathbf{F}_e) + \delta_1 + \mathbf{\Phi}_1^T\tilde{\mathbf{w}}_1)] - \frac{1}{\beta_6}\tilde{\mathbf{w}}_6^T\dot{\hat{\mathbf{w}}}_6 \qquad (5.37)\\ &= -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} - \mathbf{z}^T\mathbf{G}_2\mathbf{z} + \mathbf{s}^T\mathbf{P}\delta_1 + \mathbf{z}^T\delta_2 + \tilde{\mathbf{w}}_1^T[\mathbf{\Phi}_1(\mathbf{P}\mathbf{s} + \mathbf{G}_1\mathbf{\Psi}^{-T}\mathbf{z}) - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1] \\ &\quad + \sum_{i=2}^{5}\tilde{\mathbf{w}}_i[\tau_{i1} - \frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i] + \mathbf{z}^T\hat{\mathbf{\Psi}}^{-1}\mathbf{G}_1\tilde{\mathbf{\Psi}}(\mathbf{F}_c - \mathbf{F}_e) \qquad (5.38)\\ &\quad + \tilde{\mathbf{w}}_6^T[\mathbf{\Phi}_6\mathbf{z} - \frac{1}{\beta_6}\dot{\hat{\mathbf{w}}}_6] \qquad (5.39)\end{aligned}$$

where the uncertainties contained in δ_2 are given as

$$\delta_2 = \mathbf{d}_6 + \hat{\mathbf{\Psi}}^{-1}\mathbf{G}_1\delta_1 \tag{5.40}$$

$$\begin{aligned}\dot{V}_2 &= -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} - \mathbf{z}^T\mathbf{G}_2\mathbf{z} + \mathbf{s}^T\mathbf{P}\delta_1 + \mathbf{z}^T\delta_2 + \tilde{\mathbf{w}}_1^T\left[\mathbf{\Phi}_1^T(\mathbf{P}\mathbf{s} + \mathbf{G}_1\hat{\mathbf{\Psi}}^{-T}\mathbf{z}) - \frac{1}{\beta_1}\dot{\hat{\mathbf{w}}}_1\right] \\ &\quad + \sum_{i=2}^{5}\tilde{\mathbf{w}}_i(\tau_{i1} + \eta_{i1} - \frac{1}{\beta_i}\dot{\hat{\mathbf{w}}}_i) + \tilde{\mathbf{w}}_6^T(\mathbf{\Phi}_6\mathbf{z} - \frac{1}{\beta_6}\dot{\hat{\mathbf{w}}}_6)\end{aligned} \tag{5.41}$$

where, $\mathbf{z}^T\hat{\mathbf{\Psi}}^{-1}\mathbf{G}_1\tilde{\mathbf{\Psi}}(\mathbf{F}_c - \mathbf{F}_e) = \gamma\sum_{i=2}^{5}\eta_{i1}\tilde{\mathbf{w}}_i$ has been used.
Now we choose the weight update laws to be the following

$$\begin{aligned}\dot{\hat{\mathbf{w}}}_1 &= \beta_1[\mathbf{\Phi}_1(\mathbf{P}\mathbf{s} + \mathbf{G}_1\hat{\mathbf{\Psi}}^{-T}\mathbf{z})] \\ \dot{\hat{\mathbf{w}}}_i &= \beta_i[\tau_{\mathbf{i1}} + \eta_{i1}] \quad i \ = \ 2,\ldots,5 \\ \dot{\hat{\mathbf{w}}}_6 &= \beta_6\mathbf{\Phi}_6\mathbf{z}\end{aligned} \tag{5.42}$$

Substituting Equations (5.42) in Equation (5.40) gives

$$\dot{V}_2 = -\mathbf{s}^T\mathbf{P}\mathbf{G}_1\mathbf{s} - \mathbf{z}^T\mathbf{G}_2\mathbf{z} + \mathbf{s}^T\mathbf{P}\delta_1 + \mathbf{z}^T\delta_2 \tag{5.43}$$

The equation above can be re-written as

$$\dot{V}_2 = -\mu^T\mathbf{G}\mu + \mu^T\delta \tag{5.44}$$

where μ, $\mathbf{G}$ and δ are given as

$$\mu = \begin{bmatrix}\mathbf{s} \\ \mathbf{z}\end{bmatrix} \quad \mathbf{G} = \begin{bmatrix}\mathbf{P}\mathbf{G}_1 & 0 \\ 0 & \mathbf{G}_2\end{bmatrix} \quad \delta = \begin{bmatrix}\mathbf{P}\delta_1 \\ \delta_2\end{bmatrix} \tag{5.45}$$

The derivative can be upper bounded as

$$\dot{V}_2 \leq -\lambda_{min}(\mathbf{G})\left\|\mu\right\|^2 + \left\|\mu\right\|\delta_{max} \tag{5.46}$$

where $\lambda_{min}(\mathbf{G})$ is the minimum eigenvalue of $\mathbf{G}$ and δ_{max} is the maximum approximation error. Equation (5.46) can be re-arranged as

$$\dot{V}_2 \leq -\lambda_{min}(\mathbf{G}) \left(\|\mu\|^2 - \frac{\delta_{max}}{\lambda_{min}(\mathbf{G})} \|\mu\| \right) \tag{5.47}$$

Further

$$\begin{aligned} \dot{V}_2 &\leq -\lambda_{min}(\mathbf{G}) \left(\|\mu\|^2 - 2\frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \|\mu\| + \left(\frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \right)^2 - \left(\frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \right)^2 \right) \\ &\leq -\lambda_{min}(\mathbf{G}) \left(\|\mu\| - \frac{\delta_{max}}{2\lambda_{min}(\mathbf{G})} \right)^2 + \frac{{\delta_{max}}^2}{4\lambda_{min}(\mathbf{G})} \end{aligned} \tag{5.48}$$

Hence clearly the system is stable outside the residual set

$$\|\mu\| \geq \frac{\delta_{max}}{\lambda_{min}(\mathbf{G})} \tag{5.49}$$

Also the system converges to the boundary of the residual set

$$\|\mu\| = \frac{\delta_{max}}{\lambda_{min}(\mathbf{G})} \tag{5.50}$$

asymptotically. The tracking error can be arbitrarily made small so long as the approximation error δ_{max} is small and also by increasing value of $\lambda_{min}(\mathbf{G})$. The approximation error can be reduced using an appropriate number of the basis functions in the neural network formulation.

To improve the performance of the adaptive update laws, they are augmented as in [37]

$$\begin{aligned} \dot{\hat{\mathbf{w}}}_1 &= \beta_1[\mathbf{\Phi}_1(\mathbf{Ps} + \mathbf{G}_1\hat{\mathbf{\Psi}}^{-T}\mathbf{z}) - \nu_1\hat{\mathbf{w}}_1] \\ \dot{\hat{\mathbf{w}}}_i &= \beta_i \mathrm{proj}[\tau_{\mathbf{i1}} + \eta_{i1} + \Upsilon(g_i - \hat{\mathbf{w}}_i)] \quad i \ = \ 2, \ldots, 5 \\ \dot{\hat{\mathbf{w}}}_6 &= \beta_6[\mathbf{\Phi}_6\mathbf{z} - \nu_6\hat{\mathbf{w}}_6] \end{aligned} \tag{5.51}$$

where $\nu_1,\ \nu_6$ are positive leakage constants, Υ is another tuning parameter which ranges between $[0,\ 1]$ and g_i are the parameters representing the estimate of the inertia/mass matrix which is calculated a-priori for zero joint angles:

$$\hat{\mathbf{M}}_r^{-1} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \tag{5.52}$$

The Υ associated term acts as supervisory to avoid the projection limits. The projection operator

$$\text{proj}[a] = \begin{cases} 0, & \hat{\Psi} < \|\Psi\|_{\min} \quad \text{and} \quad a < 0 \\ a, & \text{otherwise} \end{cases} \tag{5.53}$$

All of the above modifications to the update laws help in keeping the estimates bounded.

5.2 Simulation Results

As in the previous section, variable step Runge-Kutta method (ode45) will be used in this section too for solving differential equations numerically. Time delays totaling 0.2 *sec* (200 *ms*) will be used to validate the control law. Results are plotted both against time and position. *All simulations were performed in continuous time domain using* ***Simulink*** *within the Matlab/Simulink environment.*

The control gains used in the simulations are $\nu_1 = \nu_2 = \nu_3 = 0.1$ and $\beta_1 = \beta_2 = \beta_3 = 0.1$, $\Lambda = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $G_1 = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$, $G_2 = \begin{bmatrix} 30 & 0 \\ 0 & 30 \end{bmatrix}$.

Case 1: Puncture from Medium to Free Space at 1.1 m

Velocity tracking by adaptive backstepping method subject to time delay in x and y direction by mere observation seems very good as compared to direct force application plotted in Figure 5.1. The system gradually and smoothly reaches the desired velocity inside the medium without any overshoot in case of adaptive control. On the other hand the time delay in direct force control causes the robot's velocity to oscillate around the desired velocity as a result of which it has higher velocity at the time of puncture and so the magnitudes of the velocity overshoot cannot be compared. After the puncture the direct force system becomes unstable and begins to diverge.

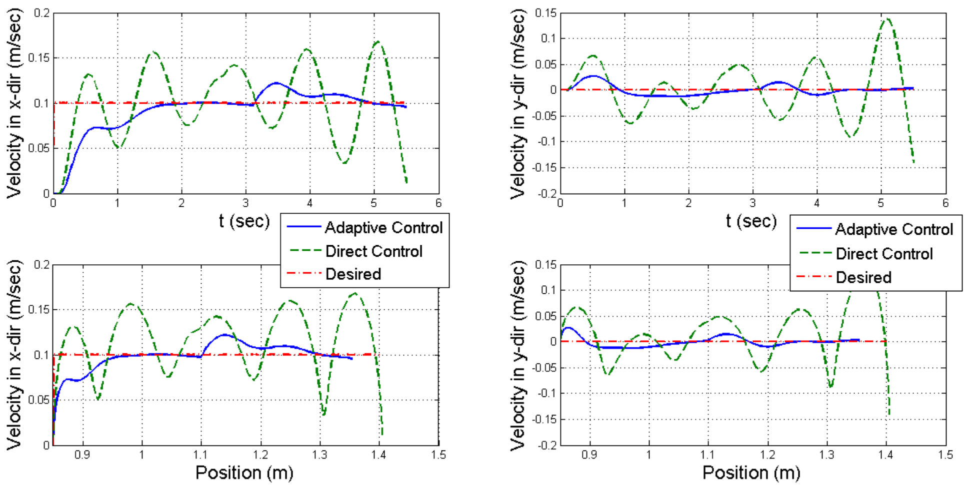

Figure 5.1. Comparison of Velocity for the Puncture Scenario.

Figure 5.2 compares the external force measured by the sensor at the end-effector. Since there is no external force acting in the y-direction the force measured is zero as seen.

Figure 5.3 compares the control forces applied to the robot by the controller in case of adaptive control and by the haptic device in case of direct force method. It can be seen that the control forces are smooth and lower in magnitude for achieving the goal in case of adaptive control as compared to direct force control which needs

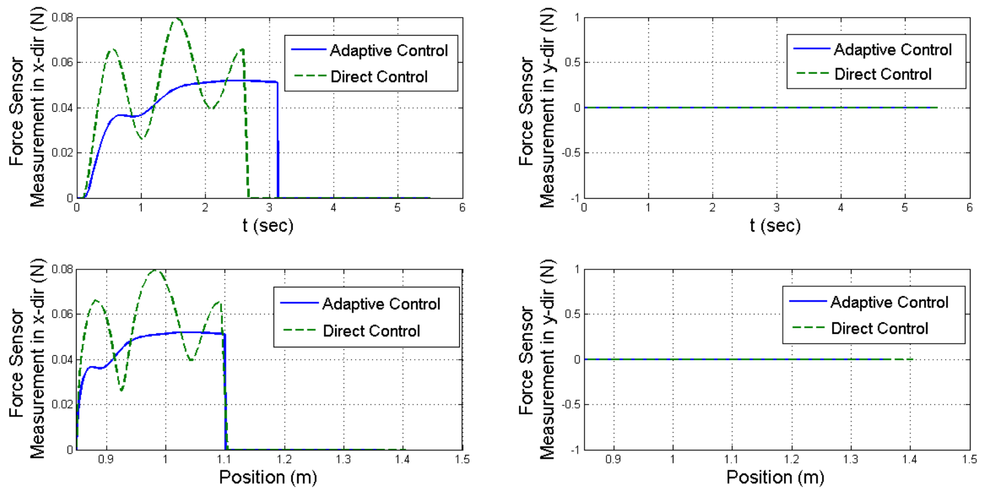

Figure 5.2. Force Measured by the Sensor during Puncture.

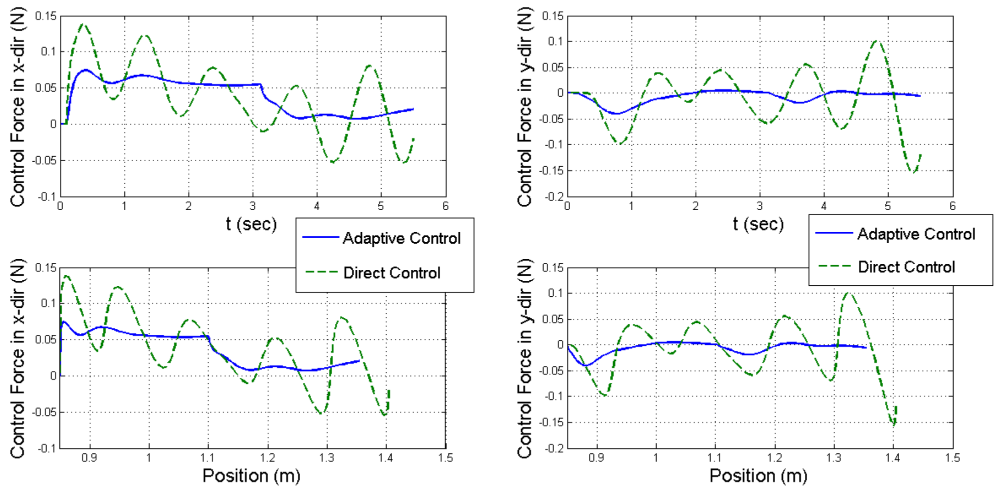

Figure 5.3. Control Force Comparison for Puncture Case.

higher control forces.

Figure 5.4, Figure 5.5 and Figure 5.6 shows the time variation of the weights in the neural network defined in Equation (5.10), Equation (5.11) and Equation (5.25) respectively for puncture scenario. It can be seen that the neural network works

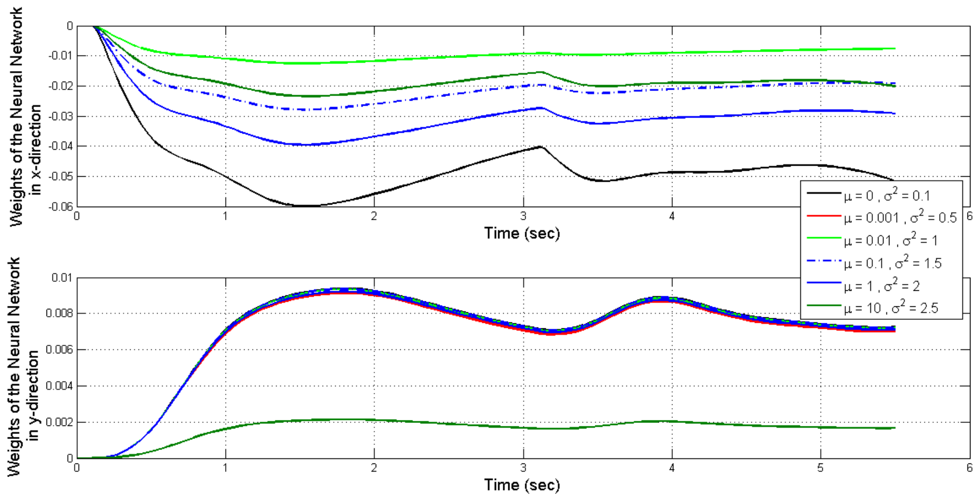

Figure 5.4. Weights of the First Neural Network given in Equation (5.8) for Puncture Case.

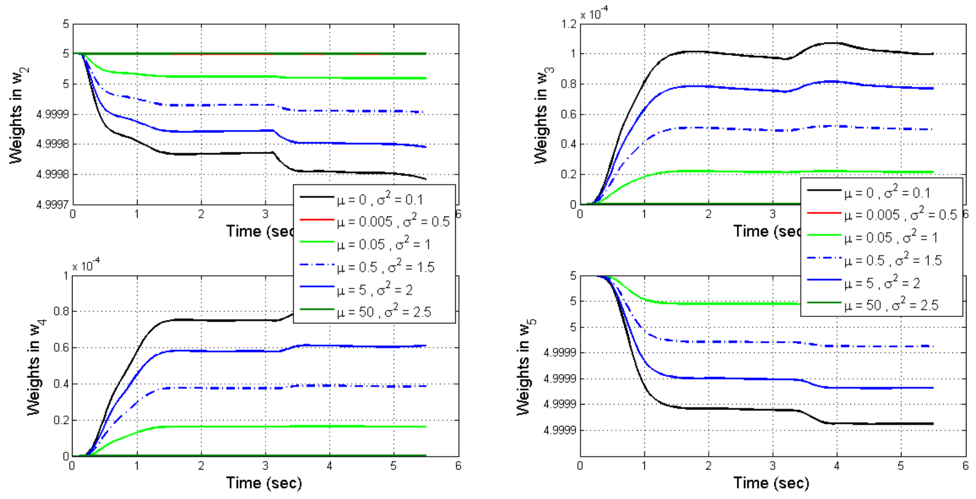

Figure 5.5. Weights of Inertia Matrix Inverse given in Equation (5.9) for Puncture Case.

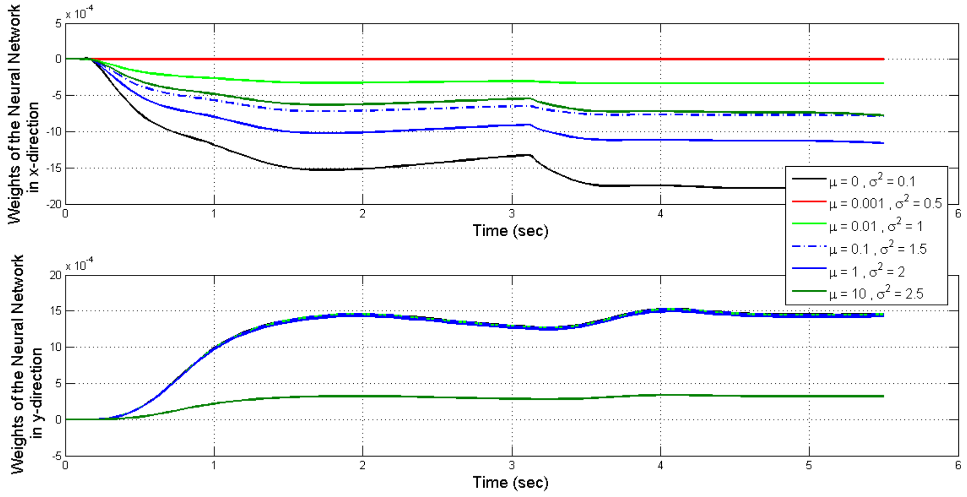

Figure 5.6. Weights of Neural Network for Estimating Derivative of Environmental Force given in Equation (5.25) for Puncture Case.

efficiently and adapts quickly to the unknown environment and nonlinear system dynamics.

Case 2: Collision with a Hard Surface from Free Space at 1 m

Figure 5.7 compares for both the methods the velocity of the end-effector with the desired velocity subject to time delay. The adaptive control tracks velocity very efficiently in both the directions before and after collision as compared to direct force application which fails to converge at all.

Figure 5.8 compares the external force measured by the sensor at the end-effector. The first impact force with the hard surface in x-direction is much lower and damps out quickly in the case of adaptive control than direct force control where the impact force gets larger with time. Also since there is no external force acting in the y-direction the force measured is seen as zero.

Figure 5.9 compares the position of the end-effector with time. It is seen that even though the direct force control has good position tracking before collision, the

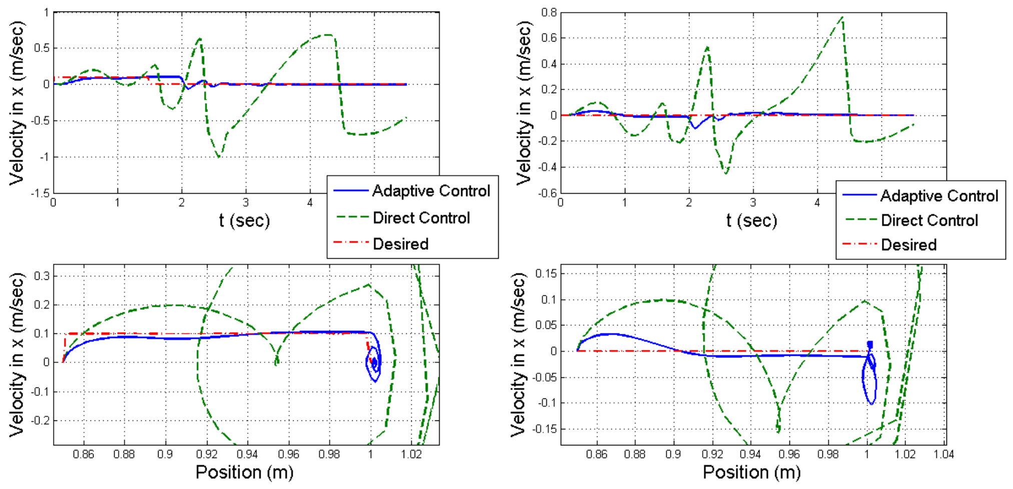

Figure 5.7. Comparison of Velocity for Collision Case.

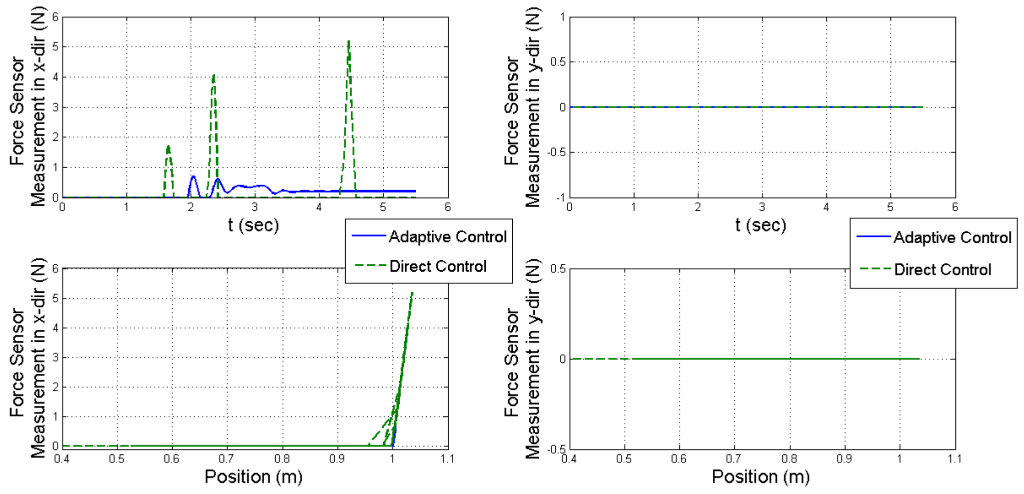

Figure 5.8. Force Measured by the Sensor during Collision.

system becomes unstable in both directions after collision. In case of adaptive control even though the response at start is sluggish but the system achieves the desired position and remains stable after collision.

Figure 5.10 compares the control forces applied to the robot by the controller in case of adaptive control and by the haptic device in case of direct force method. It

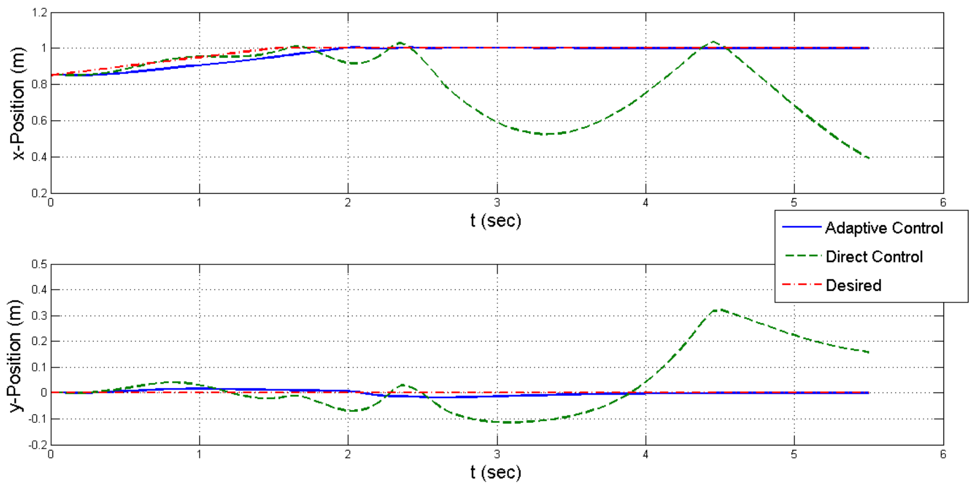

Figure 5.9. Position Comparison for Collision Case.

can be seen that the control forces are smooth and less force is required to achieve the goal in case of adaptive control as compared to direct force control which are abrupt and needs higher control forces.

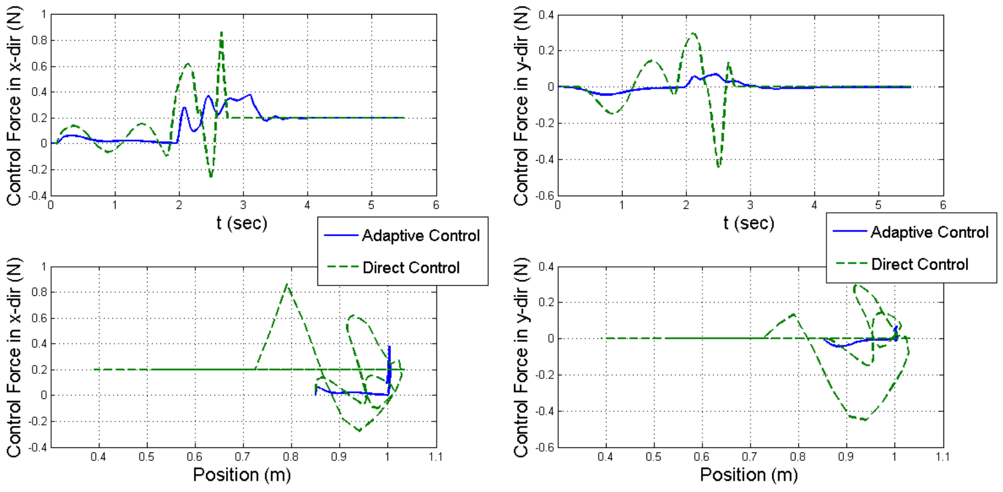

Figure 5.10. Control Force Comparison for Collision Case.

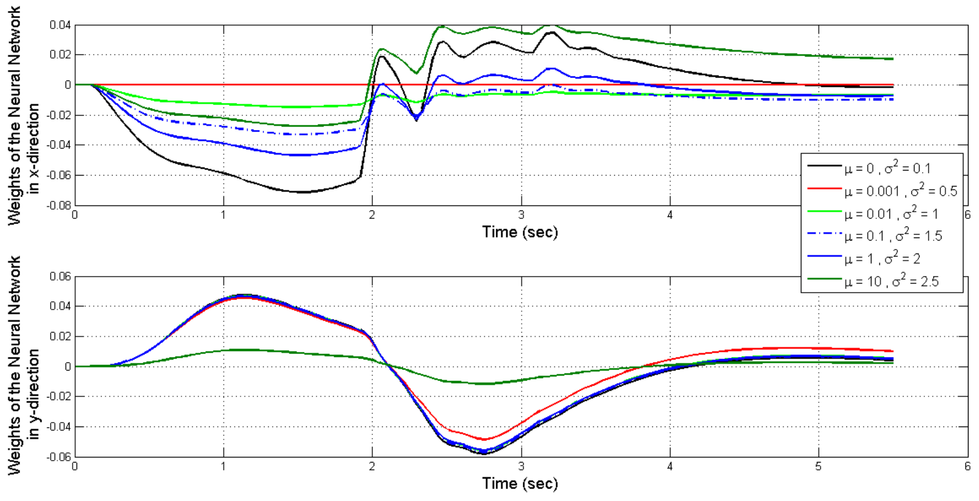

Figure 5.11. Weights of the Neural Network given in Equation (5.8) for Collision Case.

Figure 5.11 shows the time behavior of the weights in the neural network of Equation (5.10) for collision case. Figure 5.12 shows the time behavior of the weights used to estimate the inverse of inertia matrix as defined in Equation (5.11) in the collision scenario.

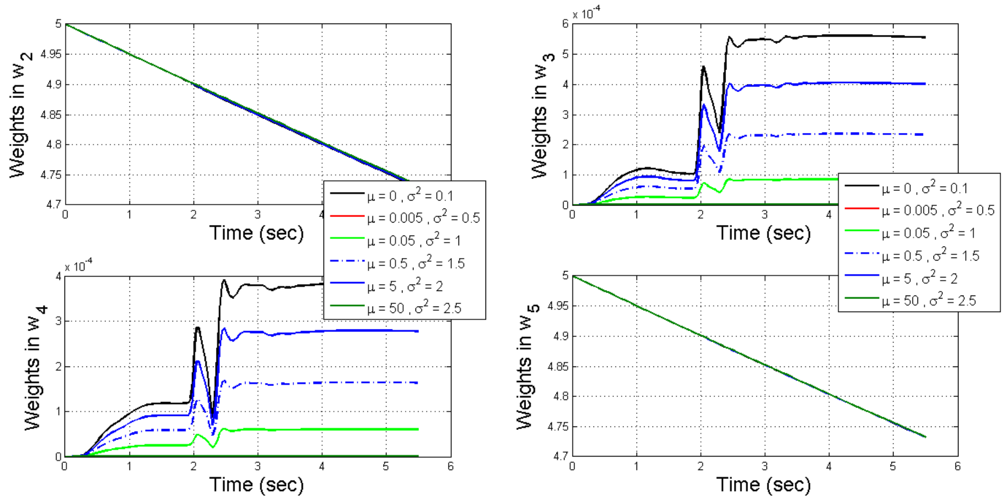

Figure 5.12. Weights of Inertia Matrix Inverse given in Equation (5.9) for Collision Case.

Figure 5.13 shows the weights of the neural network in estimating the derivative of the external forces on the robot end-effector as defined in Equation (5.25).

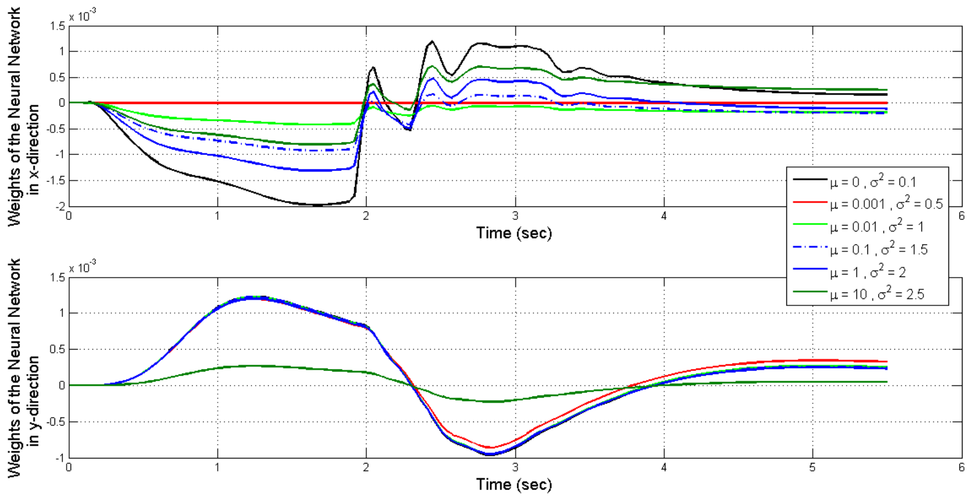

Figure 5.13. Weights of Neural Network for Estimating Derivative of Environmental Force given in Equation (5.25) for Collision Case.

Case 3: Puncture from Medium to Free Space at 1 m and Collision with a Hard Surface at 1.1 m

Figure 5.14 compares the velocity of the end-effector for both the techniques with the desired velocity subject to time delay. The adaptive control tracks velocity very efficiently in both the directions as compared to direct force application which fails to converge.

Figure 5.15 compares the external force measured by the sensor at the end-effector. The first impact force with the hard surface in x-direction is much lower in case of adaptive control than direct force control. Since there is no external force acting in the y-direction the force measured is seen as zero.

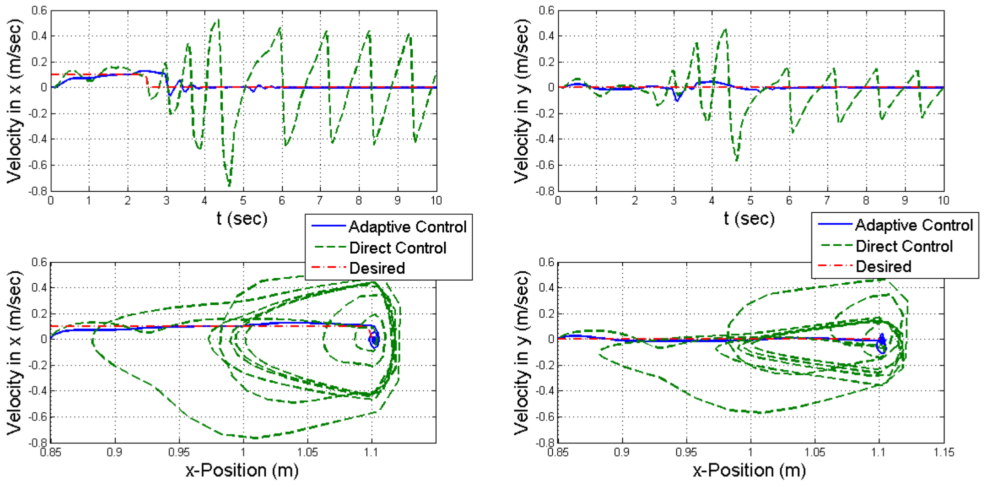

Figure 5.14. Comparison of Velocity for Combined Case.

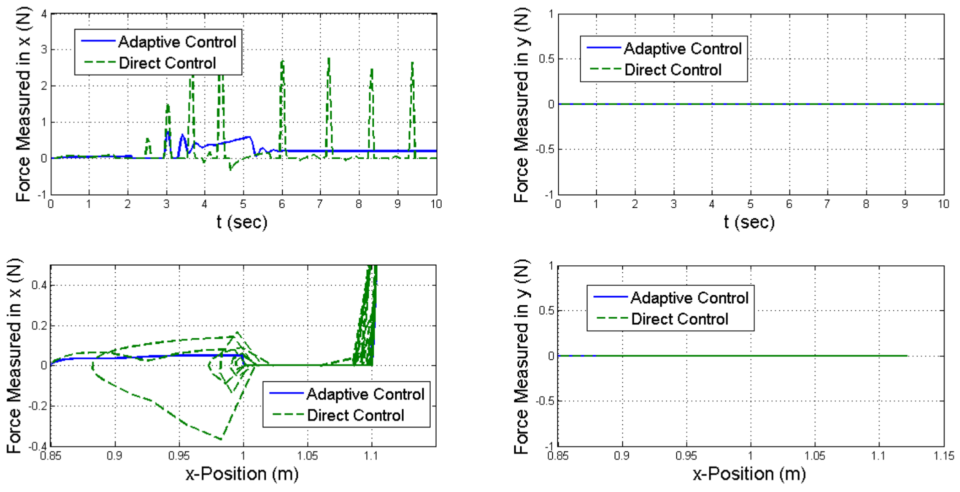

Figure 5.15. Force Measured by the Sensor for Combined Case.

Figure 5.16 compares the position of the end-effector with time. It is seen that the direct force control has good position tracking before collision but after collision the system becomes unstable in both directions. In case of adaptive control even though the response at start is sluggish but the system achieves the desired position and remains stable after collision.

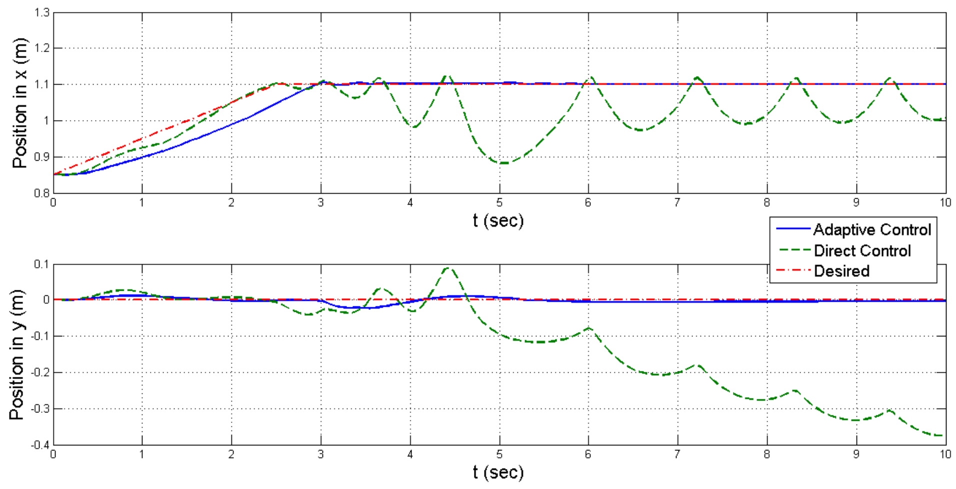

Figure 5.16. Position Comparison for Combined Case.

Figure 5.17 compares the control forces applied to the robot by the controller in case of adaptive control and by the haptic device in case of direct force method. It can be seen that the control forces are smooth and less force is required to achieve the goal in case of adaptive control as compared to direct force control which are abrupt and needs higher control forces.

Figure 5.18 shows the time behavior of the weights in the neural network of Equation (5.10). Figure 5.19 shows the behavior of the weights used to estimate the inverse of inertia matrix as defined in Equation (5.11).

Figure 5.20 shows the weights of the neural network in estimating the derivative of the external forces on the robot end-effector as defined in Equation (5.25).

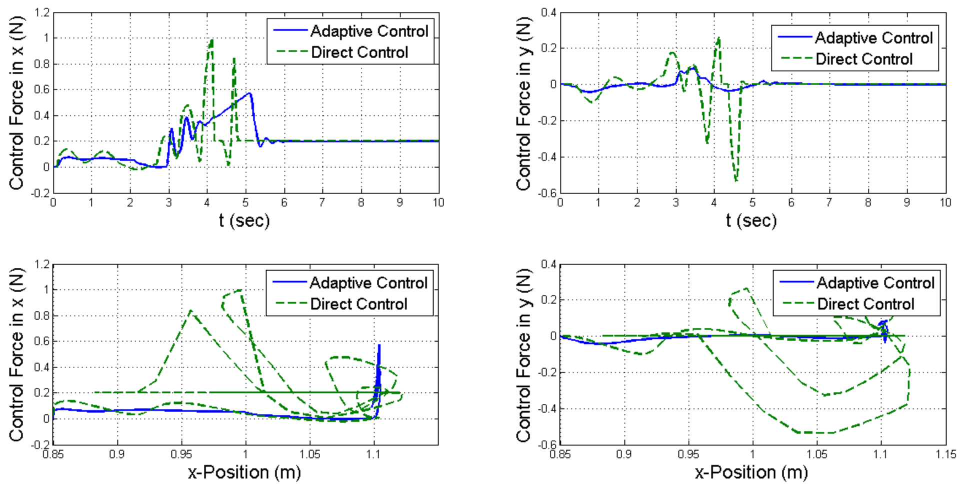

Figure 5.17. Control Force Comparison for Combined Case.

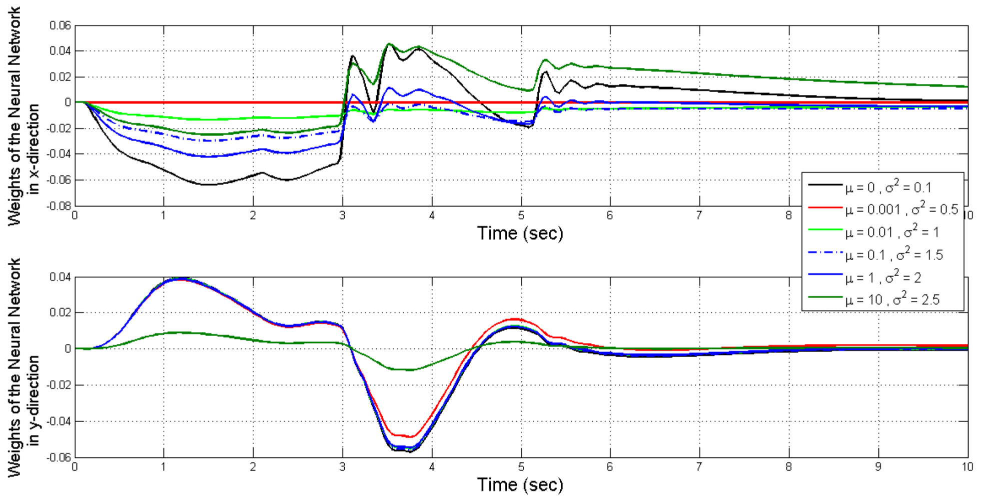

Figure 5.18. Weights of the Neural Network given in Equation (5.8).

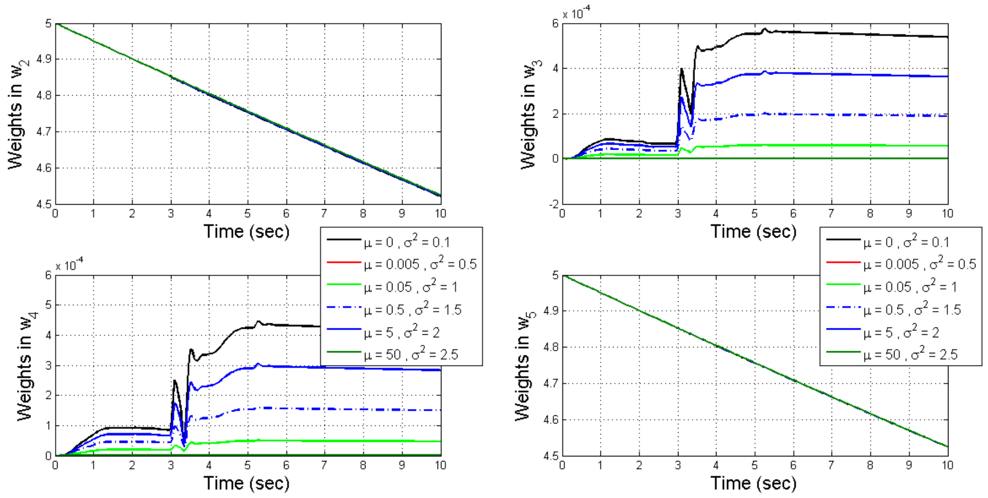

Figure 5.19. Weights of Inertia Matrix Inverse given in Equation (5.9).

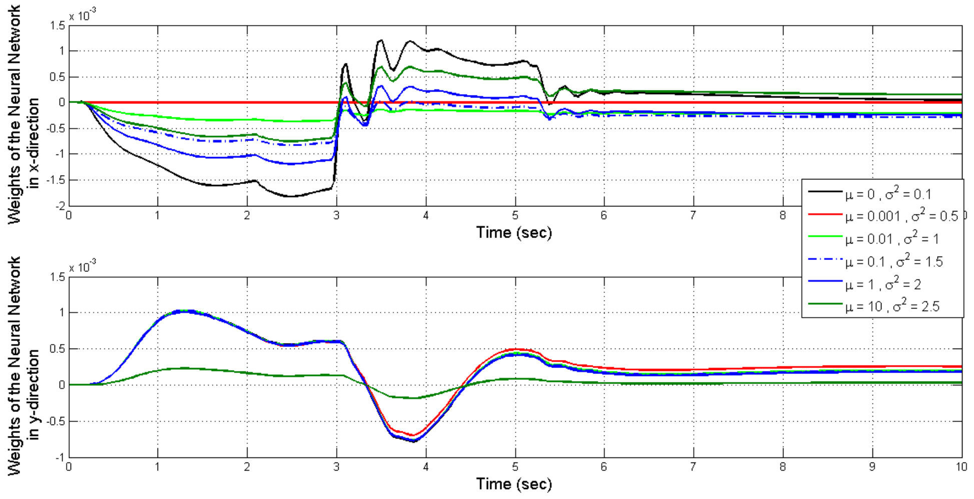

Figure 5.20. Weights of Neural Network for Estimating Derivative of Environmental Force given in Equation (5.25).

CHAPTER 6

SUMMARY, CONCLUSIONS AND FUTURE WORK

6.1 1-DOF System

The control law was successfully validated for 1-DOF robot manipulator. With the controller at the slave end there was no overshoot at start and the effect of time delay was reduced during puncture and collision with much faster settling time than the direct force application. The neural network adapted quickly to the unknown environment.

6.2 2-DOF System

The effects of time delays can be seen in the form of velocity overshoot and forces acting external to the robot end-effector which is measured by a sensor but are not accounted in the system. The proposed control law uses neural networks to estimate and to smooth out the environmental forces in order to account for the effects of the time delays.

Even though the system responses from adaptive control and direct force control for no time delay are comparable, the adaptive control completely outperforms the direct force control when subjected to 0.2 *sec* (200 *ms*) time delay. The response of the adaptive control system alone when not compared to any other method can be made more robust by tuning the PI gain values which models the human and the haptic device. Thus the control law developed in chapter 5 helps minimizing the effects of time delays such as velocity overshoot and increased collision forces by properly designing an auxiliary error which slows down the system during puncture and collision, adapts to unknown linear or nonlinear environment and achieves smoothing out of control forces.

6.3 Future Work

The work for 2-DOF control can be extended while including the actuator dynamics. There are bounded uniform approximation errors introduced due to the use of neural networks for estimation purposes. These error can be reduced by introducing robust terms in the control law. In future we will derive a robust control law based on the controller derived in Chapter 5 and test its stability and compare its performance with the one whose results are produced in this paper.

Since most robots used in the surgeries have more than 2-DOF it is necessary to test this control law on those robots. Puma-560, da Vinci Surgical System are among those robots and so our work will also be extended to Puma-560 robotic manipulator which has 6-DOF. Time delays can vary when carrying out teleoperation over internet. Therefore it is important to test the control law for varying time delay which will also be focused in our future work.

APPENDIX A

PARAMETERS OF 2-DOF ROBOT MANIPULATOR

In the following appendix, the extended form of inertia matrix, coriolis matrix, Jacobian matrix and the gravity vector denoted as $M(q)$, $C(q,\dot{q})$, $J(q)$ and $G(q)$ respectively of a 2-link planar robot which is used for the simulation are presented in [31]. Also the link properties, forward and inverse kinematic equations will be presented subsequently. Figure (A.1) shows a 2-DOF 2 link planar robot manipulator in [2].

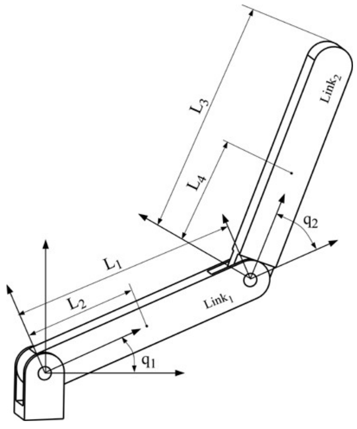

Figure A.1. 2-DOF Robot Manipulator by Bowling A. et al. in [2].

A.1 Inertia Matrix and its Elements

The inertia matrix is given as

$$M(q) = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \tag{A.1}$$

The respective elements of the Inertia matrix are

$$\begin{aligned} d_{11} &= m_1 L_2^2 + m_2(L_1^2 + L_4^2 + 2L_1L_4\cos(q_2)) + I_1 + I_2 \\ d_{12} &= d_{21} = m_2L_1L_4\cos(q_2) + m_2L_4^2 + I_2 \\ d_{22} &= m_2L_4^2 + I_2 \end{aligned} \tag{A.2}$$

A.2 Coriolis Matrix and its Elements

The coriolis matrix is given as

$$C(q, \dot{q}) = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \tag{A.3}$$

The respective elements of the coriolis matrix are

$$\begin{aligned} c_{11} &= -m_2 L_1 L_4 \sin(q_2)\dot{q}_2 \\ c_{12} &= -m_2 L_1 L_4 \sin(q_2)(\dot{q}_1 + \dot{q}_2) \\ c_{21} &= -m_2 L_1 L_4 \sin(q_2)\dot{q}_1 + m_2 L_4^2 + I_2 \\ c_{22} &= 0 \end{aligned} \tag{A.4}$$

A.3 Gravity Vector and its Elements

The gravity vector is given as

$$G(q) = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \tag{A.5}$$

The respective elements of the gravity matrix are

$$\begin{aligned} g_1 &= m_1 L_2 g \cos(q_1) + m_2 g[L_4 \cos(q_1 + q_2) + L_1 \cos(q_1)] \\ g_2 &= m_2 L_4 g \cos(q_1 + q_2) \end{aligned} \tag{A.6}$$

A.4 Jacobian Matrix and its Elements

The Jacobian matrix is given as

$$J(q) = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \tag{A.7}$$

The respective elements of the Jacobian matrix are

$$\begin{aligned} J_{11} &= -L_1 \sin(q_1) - L_3 \sin(q_1 + q_2) \\ J_{12} &= -L_3 \sin(q_1 + q_2) \\ J_{21} &= L_1 \cos(q_1) + L_3 \cos(q_1 + q_2) \\ J_{22} &= L_3 \cos(q_1 + q_2) \end{aligned} \tag{A.8}$$

A.5 Link Properties

1. Mass of Links

$$m_1 = m_2 = 0.5\ kg \tag{A.9}$$

2. Link Lengths

$$L_1 = L_3 = 0.75\ m \tag{A.10}$$

3. Center of Mass

$$L_2 = L_4 = 0.375\ m \tag{A.11}$$

4. Moment of Inertia

$$I_1 = I_2 = 0.0234\ kg\ m^2 \tag{A.12}$$

A.6 Forward Kinematics

Given joint angles (q_1, q_2) the position (x, y) of the end-effector in cartesian co-ordinates can be calculated which is known as forward kinematics. The equation are given in [38] as

$$\begin{aligned} x &= L_1 \cos(q_1) + L_3 \cos(q_1 + q_2) \\ y &= L_1 \sin(q_1) + L_3 \sin(q_1 + q_2) \end{aligned} \tag{A.13}$$

A.7 Inverse Kinematics

Given position (x, y) of the end-effector in cartesian co-ordinates the joint angles (q_1, q_2) can be calculated which is known as inverse kinematics. The equation are given in [38] as

$$\begin{aligned} C_2 &= \frac{x^2 + y^2 - L_1^2 - L_3^2}{2L_1L_3} \\ S_2 &= \pm\sqrt{1 - C_2^2} \\ q_2 &= A\tan 2(S_2, C_2) \\ q_1 &= A\tan 2(y, x) - A\tan 2(L_3\sin(q_2), L_1 + L_3\cos(q_2)) \end{aligned} \tag{A.14}$$

REFERENCES

[1] D. Richert and C. Macnab, “Direct Adaptive Force Feedback for Haptic Control with Time Delay,” in *IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*. IEEE, Sept. 2009, pp. 893–897.

[2] A. P. Bowling, J. E. Renaud, J. T. Newkirk, and N. M. Patel, “Reliability-Based Design Optimization of Robotic System Dynamic Performance,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006*, 2006, pp. 3611–3617.

[3] A. E. Saddik, “The Potential of Haptic Technologies,” *IEEE Instrumentation & Measurement Magazine*, pp. 10–17, 2007.

[4] J. Cui, S. Tosunoglu, R. Roberts, C. Moore, and D. W. Repperger, “A Review of Teleoperation System Control,” in *Proceedings of the Florida Conference on Recent Advances in Robotics*, 2003.

[5] A. K. Bejczy, “Teleoperation and Telerobotics,” in *The Mechanical Systems Design Handbook*. CRC Press, 2001, ch. 25.

[6] J. M. Wilson and M. E. Peters, “Automatic Flight Envelope Protection For Light General Aviation Aircraft,” in *Digital Avionics Systems Conference, IEEE/AIAA 28th*, 2009, pp. 1–7.

[7] J. R. Otero, P. Paparel, D. Atreya, K. Touijer, and B. Guillonneau, “History , Evolution And Application of Robotic Surgery in Urology,” *Spanish Archives of Urology*, vol. 4, pp. 335–341, 2007.

[8] A. Shamiyeh and W. Wayand, “Laparoscopic Cholecystectomy: Early and Late Complications and their Treatment.” *Langenbeck's archives of surgery / Deutsche Gesellschaft für Chirurgie*, vol. 389, no. 3, pp. 164–71, June 2004.

[9] C. A. Steiner, E. B. Bass, M. A. Talamini, H. A. Pitt, and E. P. Steinberg, “Surgical Rates and Operative Mortality for Open and Laparoscopic Choleycs-tectomy in Maryland,” *The New England Journal of Medicine*, vol. 330, no. 6, 1994.

[10] D. Richert, C. J. B. Macnab, and J. K. Pieper, “Force-Force Bilateral Haptic Control using Adaptive Backstepping with Tuning Functions,” in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics.* Ieee, July 2010, pp. 341–346.

[11] ——, “Adaptive Haptic Control for Telerobotics Transitioning Between Free, Soft, and Hard Environments,” *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 3, pp. 558–570, May 2012.

[12] D. Richert, A. Beirami, and C. J. Macnab, “Neural-Adaptive Control of Robotic Manipulators using a Supervisory Inertia Matrix,” *4th International Conference on Autonomous Robots and Agents*, pp. 634–639, Feb. 2000.

[13] C. J. B. Macnab, G. D'Eleuterio, and M. Meng, “CMAC Adaptive Control of Flexible-Joint Robots using Backstepping with Tuning Functions,” in *International Conference on Robotics & Automation*, vol. 1, 2004, pp. 2679–2686.

[14] D. Wang, K. Tuer, M. Rossi, L. Ni, and J. Shu, “The Effect of Time Delays on Tele-haptics,” *The 2nd IEEE Internatioal Workshop on Haptic, Audio and Visual Environments and Their Applications, Proceedings.*, pp. 7–12, 2003.

[15] H. Arioui, A. Kheddar, and S. Mammar, “A Predictive Wave-Based Approach for Time Delayed Virtual Environments Haptics Systems,” in *Proceedings. 11th*

IEEE International Workshop on Robot and Human Interactive Communication. Ieee, 2002, pp. 134–139.

[16] S. Munir and J. Wayne, "Internet-Based Teleoperation Using Wave Variables With Prediction," *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 2, pp. 124–133, 2002.

[17] A. Aziminejad, M. Tavakoli, R. Patel, and M. Moallem, "Transparent Time-Delayed Bilateral Teleoperation Using Wave Variables," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 3, pp. 548–555, May 2008.

[18] S. Sirouspour and A. Shahdi, "Bilateral Teleoperation under Communication Time Delay using an LQG Controller," *Proceedings of 2005 IEEE Conference on Control Applications*, pp. 1257–1262, 2005.

[19] ——, "Model Predictive Control for Transparent Teleoperation Under Communication Time Delay," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1131–1145, 2006.

[20] A. Shahdi and S. Sirouspour, "Adaptive/Robust Control for Enhanced Teleoperation under Communication Time Delay," *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2667–2672, Oct. 2007.

[21] R. Cortesão, J. Park, and O. Khatib, "Real-Time Adaptive Control for Haptic Manipulation with Active Observers," in *Intelligent Robots and Systems*, vol. 2, no. October, 2003, pp. 2938–2943.

[22] ——, "Real-Time Adaptive Control for Haptic Telemanipulation With Kalman Active Observers," *IEEE Transactions on Robotics*, vol. 22, no. 5, pp. 987–999, 2006.

[23] J. Park and R. Cortes, "Robust and Adaptive Teleoperation for Compliant Motion Tasks," in *International Conference on Advanced Robotics*, 2003.

[24] J. Park, O. Khatib, and R. Cortes, "Telepresence and Stability Analysis for Haptic Tele-Manipulation with Short Time Delay," in *International Conference on Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ*, 2005, pp. 436– 441.

[25] D. Lawrence, "Stability and Transparency in Bilateral Teleoperation," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 624–637, 1993.

[26] G. Niemeyer, "Telemanipulation with Time Delays," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 873–890, Sept. 2004.

[27] M. Nohmi, M. Ando, and T. Bock, "Contact Task by Space Teleoperation Using Force Reflection of Communication Time Delay," in *2005 International Symposium on Computational Intelligence in Robotics and Automation.* Ieee, 2005, pp. 193–198.

[28] J. E. Speich, L. Shao, and M. Goldfarb, "Modeling the Human Hand as it Interacts with a Telemanipulation System," *Mechatronics*, vol. 15, no. 9, pp. 1127–1142, Nov. 2005.

[29] M. J. Fu and M. C. Cenk, "Human Arm-and-Hand Dynamics Model with Variability Analyses for a Stylus-based Haptic Interface," *IEEE Transactions Systems, Man and Cybernetics*, vol. 42, no. 6, pp. 1633– 1644, 2012.

[30] J. J. Gil, "Influence of Vibration Modes and Human Operator on the Stability of Haptic Rendering," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 160–165, 2010.

[31] Frank L . Lewis, Darren M . Dawson, and Chaouki T . Abdallah, *Robot Manipulator Control.* CRC Press, 2003.

[32] An-Chyau Huang and Ming-Chih Chien, *Adaptive Control of Robot Manipulators.* World Scientific Publishing Co. Pvt. Ltd, 2010.

[33] M. Krstic, I. Kanellakopoulos, and P. Kokotovic, *Nonlinear and Adaptive Control Design.* John Wiley and Sons, 1995.

[34] R. Hong, "State-of-the-art of artificial neural networks and applications to mars robots," in *Electro International*, 1991, pp. 568–573.

[35] R. P. Lippmann, "An introduction to computing with neural nets," *ASSP Magazine, IEEE*, vol. 16, no. 1, pp. 7–25, Mar. 1988.

[36] B. Fornberg and C. Piret, "On Choosing a Radial Basis Function and a Shape Parameter when Solving a Convective PDE on a Sphere," *Journal of Computational Physics*, vol. 227, no. 5, pp. 2758–2780, Feb. 2008.

[37] Ioannou P. and J. Sun, *Robust Adaptive Control.* Pearson Education, Inc., 1996.

[38] J. J. Craig, *Introduction to Robotics: Mechanics and Control.* PEARSON, 2003.

BIOGRAPHICAL STATEMENT

Pankaj P Sarda was born in the city of Pune, India in 1988. He obtained his Bachelor's degree in Mechanical Engineering from University of Pune in 2010. Aspired to learn more he then came to the United States in Fall 2010 to pursue an M.S. degree program in Mechanical Engineering at the University of Texas at Arlington.

Pankaj has been in the industry for two long semesters as an Mechanical Engineering Intern assisting the lead Mechanical and Electrical engineers at Luraco Technologies, Inc., Texas, USA with Mechanical designs. His research interests include Robotics: Design & controls. He is currently working in the Aerospace Systems Lab (ASL) with Dr. Kamesh Subbarao on Control of Teleoperating Arm Subject to Time Delays.