İSTANBUL BİLGİ UNIVERSITY
INSTITUTE OF GRADUATE PROGRAMS
ELECTRICAL & ELECTRONICS ENGINEERING MASTER
DEGREE PROGRAM

# TIME-DELAYED TELEOPERATION
# WITH
# VIRTUAL ENVIRONMENT RECONSTRUCTION

Hafiz Huzaifa Azeem
119815001

Asst. Prof. Dr. Abdurrahman Eray Baran

İSTANBUL
2021

TIME-DELAYED TELEOPERATION
WITH
VIRTUAL ENVIRONMENT RECONSTRUCTION

YENİDEN OLUŞTURULMUŞ SANAL ORTAM İLE
ZAMAN GECİKMELİ TELEOPERASYON

Hafiz Huzaifa Azeem
119815001

| | | |
|---|---|---|
| **Thesis Supervisor** | **:** Asst. Prof. Dr. Abdurrahman Eray Baran (İstanbul Bilgi University) | ..... |
| **Member of a Jury** | **:** Asst. Prof. Dr. Yeşim Öniz (İstanbul Bilgi University) | ..... |
| **Member of a Jury** | **:** Asst. Prof. Dr. Teoman Naskali (Galatasaray Üniversitesi) | ..... |

**Date of approval:** 03.12.2021

**Total number of pages:** 83

**Keywords (English):**

**1)** Environment Estimation
**2)** Long Short Term Memory
**3)** Recursive Least Squares
**4)** Bilateral Teleoperation
**5)** Virtual Environment

**Keywords (Turkish):**

**1)** Ortam Kestirimi
**2)** Uzun Kısa Süreli Bellek
**3)** Yinelenen En Kücük Kareler
**4)** Çift Yönlü Teleoperasyon
**5)** Sanal Ortam

# ACKNOWLEDGEMENTS

I would like to express my special appreciation to Asst. Prof. Dr. Abdurrahman Eray Baran, my thesis supervisor, for his continuous mentoring and guidance. He inspired me time and time again with his positive feedback and helpful suggestions about my research. His in-depth reviews and prognostic evaluations of my work assisted me in achieving this objective. His support during the pandemic and post pandemic is much appreciative. I thank him from the bottom of my heart.

I would like to express my sincere thanks to the jury members Asst. Prof. Dr. Yeşim Öniz and Asst. Prof. Dr. Teoman Naskali.

I would like to thank Asst. Prof. Kansu Oğuz Canbek for his guidance. His help and support was very important for me in learning.

I would like to express my gratitude towards my lab colleague Fatima Jabbar Majeed, her sincere help during this project was of great importance. I would also like to thank my lab colleague Ilkay Turac Ozcelik who also helped me again and again through this research.

Finally I would like to give my very special regards to my family for their constant support and encouragement. They have played an instrumental role in letting me achieve my dream. Their love, affection and belief in me has helped me in completion of this research.

# ABSTRACT

Teleoperations have achieved a milestone in the field of robotics, performing extremely difficult tasks in hazardous environments such as undersea, nuclear sites, space exploration and tele-surgery. Their ability to perform tasks previously performed by a master device and to execute commands through a distant slave has opened many doors in the field of advanced robotics. When the distance between the master and the slave is long, the communication channel is subjected to time delays.

In this thesis, our motivation was to find a way to reduce the effects of the time delay problem by estimating the parameters of remote environment contact force and rendering a virtual environment on the master side in a bilateral teleoperation system. First, Deep Learning Algorithm, Long Short Term Memory (LSTM) was used to estimate the parameters of the remote environment. After careful consideration, it was concluded that LSTM was not suitable for real-time implementation.

Following the progress, the use of Recursive Least Squares (RLS) is demonstrated for estimating the parameters of remote environment contact force and rendering a virtual environment on the master side in a bilateral teleoperation system. Proper and fast estimation of the remote environment impedance plays a crucial role in the realization of local force controller for time delayed teleoperation systems.

Three different variants of RLS estimator were implemented and compared against three different impedance models. The algorithms were initially tested in the simulation environment making use of a recorded real experiment data set. The force reconstruction performances are compared to evaluate the implemented models and estimators. Based on the simulation results, one of the estimators and one of the models are selected for experimental validation on a single degree of freedom motion control system.

In a set of real experiments performed, the estimated force was rendered on the master side as a virtual environment, this way a local force feedback control loop was established. The local force control loop was based on the predicted environment using the estimated parameters of RLS running in real time. Therefore, the master side was able to feel same force felt at the slave side without any time delay.

# ÖZET

Teleoperasyon uygulamaları robotik alanında deniz altı, nükleer alanlar, uzay araştırmaları ve robotik cerrahi müdahaleler uygulamaları gibi tehlikeli ve önemli operasyonları başararak bir dönüm noktası olmuştur. Daha önce yapılmış olan uygulamaları efendi sistem aracılığıyla köle sisteme ne kadar uzak dahi olsa iletebilmesi gelişmiş robotik alanında birçok kapı açmıştır. Efendi ve köle sistemler arasındaki mesafe uzak oldukça iletişim hattında zaman gecikmeleri oluşmaktadır.

Bu tezdeki motivasyon, uzaktaki ortamın temas kuvveti parametrelerini tahmin ederek ve köle sistemin temas ettiği ortamı sanal bir şekilde efendi sisteme çift yönlü kontrol ile iletmek ve efendi-köle sistemler arasında oluşan zaman gecikmesini azaltmaktır. Öncelikle, derin öğrenme algoritmaları ve Uzun Kısa Süreli Bellek (LSTM) algoritmaları uzaktaki ortamın parametrelerini tahmin etmek için kullanldı. Dikkatli incelemeler sonucunda LSTM algoritmasının gerçek zamanlı kontrolcüler için kullanılamayacağına karar verildi.

Yukarıda belirtilen işlemler sonucunda, Yinelenen En Küçük Kareler (RLS) algoritması uzaktaki ortamın temas kuvveti parametrelerini tahmin etmek ve çift yönlü kontrolcüde, uzaktaki ortamı efendi sistemde sanal olarak yaratmak için kullanıldı. Zaman gecikmeli tele operasyon sistemlerinde, uzaktaki ortamın empedansını düzgün ve hızlı bir şekilde tahmin etmek yerel kuvvet kontrolcüsünün gerçekleşmesi için çok önemli bir rol oynar.

Üç farklı empedans modeli için üç farklı RLS yöntemi denendi ve gerekli karşılaştırmaları yapıldı. Algoritmalar öncelikle daha önceden gerçek zamanlı kontrolcü tarafından kaydedilmiş bilgileri kullanan simülasyonlar ile denendi. Kuvveti yeniden oluşturma performansları, uygulanmış modeller ve kestirim yöntemleri ile karşılaştırıldı. Simülasyon sonuçlarına göre, bir tane model ve bir tane kestirimci, bir serbestlik dereceli hareket kontrolü sistemi ile deneysel doğrulama için seçildi.

Gerçekleştirilen gerçek zamanlı deneylerde, tahmin edilen kuvvet efendi sistemde sanal olarak yaratıldı ve böylece yerel kuvvet kontrolcüsü kuruldu. Yerel kuvvet kontrolcüsü, RLS algoritmasının gerçek zamanlı olarak tahmin ettiği parametrelere kestirimi yapılan ortam parametrelerini kullanarak çalışıyor. Böylece, efendi sistem tarafında, köle sistemin hissettiği kuvvetlerin tamamı zaman gecikmesi olmadan hissedilebiliyor.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF SYMBOL

| | |
|---|---|
| $q_s$ | Slave Displacement |
| $q_m$ | Master Displacement |
| $\dot{q}_s$ | Slave Velocity |
| $\dot{q}_m$ | Master Velocity |
| $\ddot{q}_s$ | Slave Acceleration |
| $\ddot{q}_m$ | Master Acceleration |
| $f_s$ | Slave Force |
| $f_m$ | Master Force |
| $\dot{f}_s$ | Derivative of Slave Force |
| $\dot{f}_m$ | Derivative of Master Force |
| $\ddot{f}_s$ | Second Derivative of Slave Force |
| $\ddot{f}_m$ | Second Derivative of Master Force |
| $\epsilon_q$ | Position Error for Bilateral Control |
| $\epsilon_f$ | Force Error for Bilateral Control |
| $\dot{\epsilon}_q$ | Derivative of Position Error for Bilateral Control |
| $\dot{\epsilon}_f$ | Derivative of Force Error for Bilateral Control |
| $\ddot{\epsilon}_f$ | Second Derivative of Force Error for Bilateral Control |
| $C$ | Weight of position in position error for Bilateral Control |
| $K_q$ | Convergence coefficient of position error for Bilateral Control |
| $K_f$ | Convergence coefficient of Force error for Bilateral Control |
| $\tau_{ref}$ | Input Torque |
| $\tau_{dis}$ | Disturbance Torque |
| $A_n$ | Nominal Plant Inertia |
| $K_n$ | Nominal Torque Constant |
| $i_{ref}$ | Input Current |
| $f_e$ | Environment Force |
| $m_e$ | Environment Mass |
| $K_e$ | Spring Coefficient |

| | |
|---|---|
| $D_e$ | Damping Coefficient |
| $W$ | Weights Of The Input |
| $U$ | Recurrent Connections |
| $b$ | Biases |
| $F_t$ | Forget Gate |
| $I_t$ | Input Gate |
| $C_t^*$ | Candidate Value |
| $C_t$ | Cell State |
| $O_t$ | Output Gate |
| $h_t$ | Hidden Output |
| $\sigma$ | Sigmoid Function |
| $Tanh$ | Activation Function |
| $\odot$ | Hadamard Product |
| $P$ | Covariance Matrix, |
| $\phi_N$ | Nth Row Of The System Matrix |
| $K$ | Gain |
| $\lambda$ | Forgetting Factor |
| $m$ | Mass |
| $b$ | Damping Coefficient |
| $k$ | Linear Spring Coefficient |
| $\nu$ | Non-Linear Spring Coefficient |
| $F_{env}$ | Force of Environment |
| $x_r$ | End Effector Position |
| $x_e$ | Environment Contact Position |
| $\Phi$ | State Transition Matrix |
| $\theta$ | Vector of Unknowns to be Estimated |
| $Y$ | Vector of Output Measurement |
| $K$ | Gain Vector |
| $\lambda$ | Forgetting Factor |
| $r$ | Window Size |
| $P$ | Covariance Matrix |

| | |
|---|---|
| $ref$ | Reference |
| $res$ | Response |
| $B$ | Viscous Friction Coefficient |
| $G$ | Gravitation Effect |

# LIST OF ACRONYMS/ABBREVIATIONS

| | |
|---|---|
| DOB | Disturbance Observer |
| CDOB | Communication Disturbance Observer |
| RFOB | Reaction Force Observer |
| PD | Proportional Derivative |
| DOF | Degree of Freedom |
| MIMO | Multiple Input Multiple Output |
| PLC | Programmable Logic Controller |
| PLA | Thermoplastic Polyester |
| NN | Neural Network |
| RNN | Recurrent Neural Network |
| LSTM | Long Short Term Memory |
| Deep LfD | Deep learning from demonstration |
| dVRK | da Vinci Research Kit |
| RBF | Radial Basis Function |
| DCAE | Deep Convolutional Auto-Encoders |
| RFOB | Reaction Force Observer |
| BPTT | Back Propagation Through Time |
| LMS | Least Mean Squares |
| LS | Least Squares |
| RLS | Recursive Least Squares |

# 1. INTRODUCTION

### 1.0.1. Teleoperation

The prefix tele which comes from Greek means at a distance, thus teleoperation by definition means operation at a distance. Teleoperations allow humans to manipulate things at a distance by presenting the operator with identical conditions to those at the The plant's total disturbance torque, nominal torque constant, and nominal plant in- ertia are denoted by dis(t), Kn and An, respectively. When the measurement channel has a delay Dm and the slave plant's position and velocity are accessible at a remote site,Only the delayed position and velocity (i.e. q(t −Dm)) and ˙q(t −Dm)) are accessible to the controller. As a result, an observer must be utilized to offer synchronized, that is, without delays, predictions of remote plant position and velocity. In that way, the objective might be defined as designing an observer based on existing measurements(i.e. q(t −Dm)) and ˙q(t −Dm)) and Kniref is the control input, and it has a zero estimate error. Once the slave plant's actual output has been approximated, it may be utilized in another controller to create the remote plant's input. 4.3.1. Construction of Motion Controller With Delay A second observer may be added to the network to produce an initial estimate of the slave plant velocity, assuming that the slave plant shows nominal behavior with integrated DOB and that the entire impact of measurement delay is regarded as a network disturbance working on the estimator. nwdis (t) = Kniref (t) −An ¨q(t −Dm) (4.19) Like the usual disturbance observer structure, this network disturbance may be esti- mated using delayed slave plant velocity and a low pass filter. The estimated network disturbance represents the torque that is supposed to act on the slave plant during the measurement delay. Since the slave plant is required to respond nominally with DOB, the anticipated network disturbance may be routed via the slave plant's nominal inertia and integrated to create the velocity difference that is projected to exist during the delay period. This can be stated mathematically as: Δ ˙q(t) = ∫ 1 An nwdis () d (4.20) 20 icon = −(Kcpqs + Kcd ˙qs) (4.26) where Kcp and Kcd are convergence parameters. The addition of con-

vergence terms to both the estimator and the slave plant ensures that the estimation of the remote plant response is synchronized on time with the actual slave output. The only thing left to do now is build a controller that will create the control current, which will then be transferred to the slave system after a control channel delay Dc. When employing a PD controller to track master position reference, the following control current may be used as the control input to both the slave system and the observer. iref (t) = Kvel( ˙qm(t) −ˆ ˙qs(t)) + Kpos(qm(t) −ˆqs(t)) (4.27) Now we have the whole observer-controller structure for position control with a time delay. Figure 4.4: Block Diagram of Motion Control with Delay. 22 4.4. Motion Control System With Delay and Local Force Feedback In this chapter bilateral controller with time delay and local force feedback is focused. The force felt on the slave system is fed back to the master system in order to create a local force loop. When the slave makes a contact with environment, the estimated force calculated from the environment parameters render on the master side as virtual environment reconstruction hence creating a close loop system. The overall structure of the system seems similar motion control system without delay but here instead of slave position and velocity, estimated position and velocity are used and the force felt is also not the slave but the estimated force that will render on the master system. 4.4.1. Construction of Bilateral Controller With Delay and Local Force Feedback For such a close loop system, tracking error can be defined as a linear combination of errors in position and velocity tracking using the position and velocity measurements of master and slave systems, position tracking error can be stated as: q(t) = ( ˙qm(t) −ˆ ˙qs(t)) + C(qm(t) −ˆqs(t)) (4.28) where ˆ ˙qs(t) and ˆqs(t) refers to the estimated velocity and position. The following error dynamics can be applied to enforce exponential decay for this error: ˙q(t) + Kqq(t) = 0 (4.29) ( ¨qm(t) − ¨qs(t)) + (C + Kq)( ˙qm(t) −ˆ ˙qs(t)) + CKq(qm −ˆqs) (4.30) 23 Where, Kq refers to the exponential decay rate of the position error and C refers to the weight of position in position error The force tracking error can be expressed as f (t) = (fm(t) + ˆfs(t)) (4.31) by enforcing the following exponentially decaying dynamics we get ¨f (t) + Ky ˙f (t) + Kf f (t) = 0 (4.32) Without loss of generality, one can assume a pure spring environment for the interaction force (i.e. fm(t) = Keqm(t))Hence, the expression given in (4.32) can be recast as follows: Ke( ¨qm(t) + ¨qs(t)) + KeKy(

˙qm(t) + ˆ ˙qs(t)) + Kf (fm(t) + ˆfs(t)) = 0 (4.33) Dividing this equation by the common spring constant Ke gives the following expres- sion: ( ¨qm(t) + ¨qs(t)) + Ky( ˙qm(t) + ˆ ˙qs(t)) + Kf Ke (fm(t) + ˆfs(t)) = 0 (4.34) Equations (4.30) and (4.34) set the baseline for the solution of the desired accelerations. Adding (4.30) to (4.34) and dividing by 2 yields to: 24 ¨qm(t) = −0.5(C + Kq)( ˙qm(t) −ˆ ˙qs(t)) −0.5CKq(qm(t) −ˆqs(t)) −0.5Ky( ˙qm(t) + ˆ ˙qs(t)) −0.5 Kf Ke (fm(t) + ˆfs(t)) = 0 (4.35) ¨qs(t) = −0.5Ky( ˙qm(t) + ˆ ˙qs(t)) −0.5 Kf Ke (fm(t) + ˆfs(t)) +0.5(C + Kq)( ˙qm(t) −ˆ ˙qs(t)) + 0.5CKq(qm(t) −ˆqs(t)) = 0 (4.36) Figure 4.5: Block Diagram of Motion Control with Delay and local force feedback. 25 5. ENVIRONMENT ESTIMATION METHODS 5.0.1. Deep Learning Algorithm pop5.0.1.1. Introduction to Recurrent Neural Network. A Recurrent Neu- ral Network is an artificial neural network,RNNs are a sort of neural network that is both strong and robust. They are one of the most promising algorithms in use since they are the only ones having an internal memory. They use memory by taking in- formation from previous output to influence a current input and output. This is why they're the chosen algorithm for time series, voice, text, financial data, audio, video, weather, and many other types of sequential data. RNNs have been criticized for two reasons Vanishing gradient and Exploding gradient, in vanishing gradient the weights approaches to a very small value and for exploding gradient the weight approaches to a very large value hence resulting in large error in the back-propagation. An unrolled RNN cell is shown in Figure 5.1. Figure 5.1: RNN Cell 26 pop5.0.1.2. Long Short Term Memory (LSTM). The LSTM architecture was developed by Sepp Hochreiter and Juergen Schmidhuber as a solution to the vanishing gradient problem [41]. Long short-term memory networks are a type of recurrent neu- ral network that expands the memory capacity. This memory may be thought of as a gated cell, with gated indicating that the cell selects whether or not to store or erase information (i.e., whether or not to open the gates) based on the value it gives to the data. Weights, which are also learnt by the algorithm, are used to allocate importance. This basically implies that it learns what information is important over time and what information is not. LSTM is well-suited to detect, analyze, and predict time series with unpredictable time delays. The model is trained via back-propagation. In the deep layers of the neural network, LSTMs employ cells with three gates: an input gate it, an

output gate ot, and a forget gate ft. These gates control the flow of data required for the network's output to be predicted. An unrolled LSTM cell is shown in Figure 5.2 Figure 5.2: LSTM Cell 27 The gates corresponding to LSTM Cell are as followed. ft = ∗(Wf ∗xt + Uf ∗ht−1 + bf ) (5.1) it = ∗(Wi ∗xt + Ui ∗ht−1 + bi) (5.2) c∗t = Tanh ∗(Wc ∗xt + Uc ∗ht−1 + bc) (5.3) ot = ∗(Wc ∗xt + Uo ∗ht−1 + bo) (5.4) ct = ft ⊙ct−1 + it ⊙c∗t (5.5) ht = ot ⊙(Tanh ∗ct) (5.6) In the above equations, Vectors are represented by lowercase variables. Matrices Wq and Uq contains the weights of the input and recurrent connections. The subscript it, ot, ft and ct corresponds to input gate, output gate , forget gate and memory cell depending on the activation being calculated. We will use "vector notation" in this part ct ∈Rh is not just one cell of one LSTM unit, but contains h LSTM unit's cells. The initial values are co = 0 and ho = 0 and the operator ⊙ denotes the Hadamard product (element-wise product). The subscript t indexes the time step , xt ∈Rd is the input vector and ht ∈ Rh is the hidden output. W ∈ Rh×d , U ∈ Rh×h and b ∈ Rh where the superscripts d and h refer to the number of input features and number of hidden units, respectively. Ft ∈ Rh It is the forget gate that determines whether data should be preserved or destroyed. The sigmoid function conveys information from the previous hidden state as well as current input information. The numbers range from 0 to 1. The closer you go to zero, the more you forget; the closer you get to one, the more you remember. It ∈Rh is the input gate it is used to update the values coming from sigmoid as memory and from tanh as candidate values(C∗ t ∈Rh) for updating the values. To begin, we mix 28 the prior hidden state with the current input using a sigmoid function. This controls which values will be modified by adjusting the values to be between 0 and 1. You also submit the hidden state and current input into the tanh function to compress values between -1 and 1 to assist manage the network. After that, the sigmoid output is multiplied by the tanh output. The sigmoid output will select which tanh output information should be maintained. Ct ∈ Rh is the cell state that has sufficient information to compute the cell's status. In a point-by-point manner, the cell state is multiplied by the forget vector. This has the ability to drop values in the cell state if multiplied by values near to 0. Then, on the output of the input gate, we do a point-wise addition, which changes the cell state to new values that the neural network considers relevant. As a result, a

new cell state has emerged. Ot ∈Rh is the output gate it determines the next concealed state to be. It's crucial to keep in mind that the hidden state incorporates information from previous inputs. The concealed state is also used to make predictions. To begin, we mix the prior hidden state with the current input using a sigmoid function. The tanh function receives the newly adjusted cell state. We multiply the tanh output with the sigmoid output to decide what information the hidden state should contain. The output is the hidden state. The changed cell state and hidden values are then carried over to the next time step. The core concept in LSTMs is the cell state, the horizontal line running through the top of the LSTM Cell. There are just a few small linear interactions as it passes down the entire chain which makes it really convenient for data to simply flow along it without being altered. 29 5.1. Recursive Least Squares (RLS) Gabor was the first to use a Volterra series to come up with the notion of a non-linear adaptive filter in 1954 [42] . The LMS algorithm, also known as the Widrow-Hoff rule, was developed by Widrow and Hoff in 1959 as part of their research on a pattern recognition system known as the adaptive linear element [43]. In that LMS algorithm are based on error-correction learning, the LMS algorithm is similar to Rosenblatt's perceptron [44]. During the early years of neural networks, they both appeared about the same time in the late 1950s. Today, the significance of Rosenblatt's perceptron is mostly historical. The LMS algorithm, on the other hand, has stood the test of time.Channel equal- ization, system identification, predictive deconvolution, spectrum analysis, signal de- tection, noise cancellation, and beam forming are just a few of the disciplines where adaptive filters are used.The LMS algorithm is essentially a stochastic gradient algo- rithm, which implies that the gradient of the error performance surface with respect to the free parameter vector varies randomly from iteration to iteration. Because of this stochasticity, which is compounded by the existence of non-linear feedback, a full convergence study of the LMS algorithm is a complex mathematical effort. The LMS method achieves simplicity of implementation by employing instantaneous estimations of •the input signal vector's autocorrelation matrix and •the input vector's cross-correlation vector with the intended response. The recursive least-squares (RLS) approach, on the other hand, uses continually up- dated estimates of these two parameters that go all the way back to the start of the adaptive process. As a result,

the RLS algorithm has the following characteristics. •Convergence rate that is often an order of magnitude quicker than that of the 30 distant site. A typical teleoperation system consists of a master manipulator, a slave manipulator, communication channel and control algorithm as shown in figure 1.1.

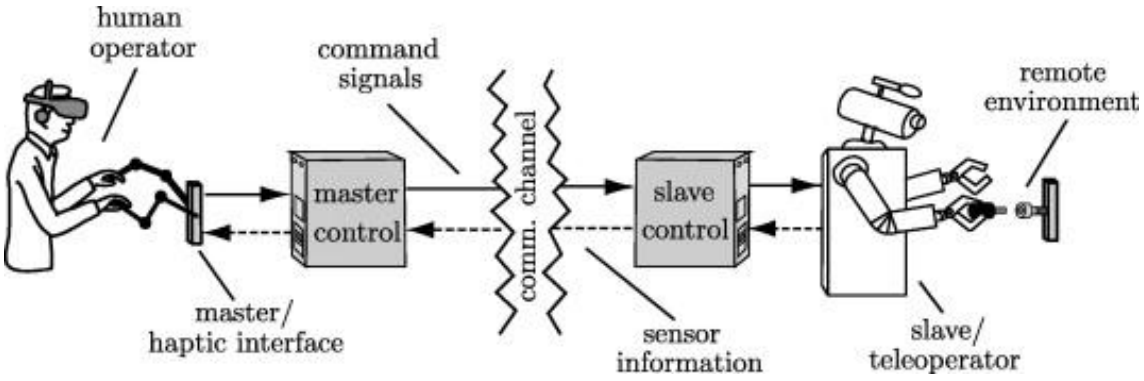



Figure 1.1: Teleoperation System [1]

A human operator physically operates a master device and a slave device is assumed to track the movement directed to master device, the objective is to transmit master positions to the slave. The master system on the other hand is meant to react to the human operator with the same force as the one caused by the slave system's contact with the remote environment, the contact force at the slave is immediately delivered back to the master as a response force.

Their ability to accomplish tasks originally performed by a master device and to execute commands via distant slave has opened numerous opportunities in the field of advanced robotics. Teleoperation achieved a milestone in the field of robotics performing extremely difficult tasks in hazardous environments such as radioactive yards, operating in areas where human life is at risk such as bomb disposal site or earth quake sites. So it is not wrong to say that bilateral teleoperations have reduced the amount of fatalities.

Teleoperation robots can go deeper into the sea where human life becomes in dange, chances of marine life exploration would become easier and safer. Assembly, mainte-

nance, and repair tasks in underwater operations will be cheaper and safer if teleoperation robots are used.

With the recent advancement in space exploration, mobile teleoperation robots can play amazing roles. The risk of losing a human will be minimised and it would help dramatically in understanding the nature of outer space. The cost of assembly, maintenance, and repair jobs in outer space operations will be reduced by using teleoperation robots instead of direct human interaction.

Furthermore, teleoperations have open doors for doctors to perform surgeries (that require particular skill and experience) from a distant location. Bringing the distant surgical room to the surgeon's fingertips would save time, money, and effort while also increasing the chances of success. Countries with poor medical facilities can benefit greatly from tele-surgery because of the lack of skill and educated doctors.

### 1.0.2. Bilateral Control

The communication channel between the two, master and slave plays a critical role in transmitting the essential signals carrying displacement and force information among either sides of the teleoperation system usually refers to bilateral controller. Two important measures can be used to assess the control theoretic performance of bilateral teleoperation systems. [2]

The controller's potential to maintain closed loop stability irrespective of the operator's movement or the environment's response. A teleoperation system must first demonstrate its stability before being used in a real-world situation.

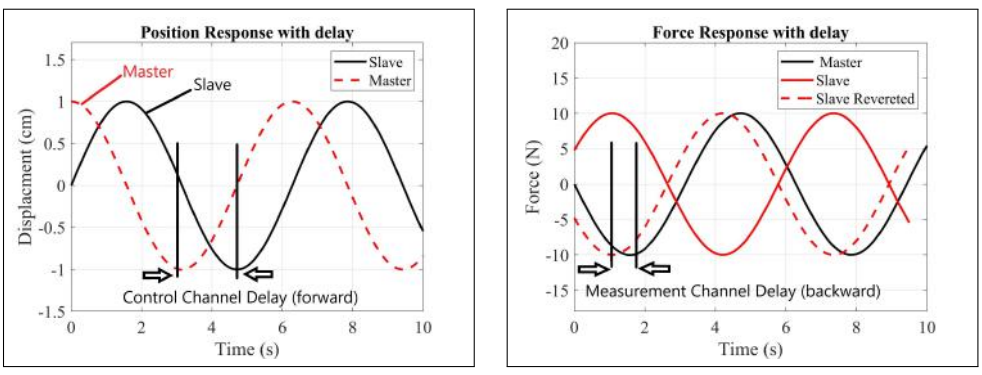
Secondly, transparency demonstrates the controller's potential to transmit operator movements and environmental pressures to the remote side and operator. When the human operator feels the same forces provided by the distant environment and the remote system produces the same movements forced by the human operator, perfect transparency is attained.

## 1.1. Problem Statement

### 1.1.1. Time Delay

When the distance between the master and the slave is long, the communication channel (i.e. transfer of position and force information) is subject to time delays. Time-delay has restricted the advancement of teleoperation in many areas of research by destabilizing the system. Latency and time delay in dynamic systems are known to degrade the control performance and make the closed-loop stabilization problem more challenging.

Imagine that you have a teleoperation system with a master and a slave that are at long distance. Despite the time delay in the system, the slave robot has been proven to be able to trace the master robot position reference and force reference [3] as indicated in the figure 1.2. At first, the delay is felt in the forward channel known as control channel delay as shown in A where the slave traces the master motion with a delay. Then the second time a delay is felt in the backward channel known as measurement channel delay as shown in B where the slave interacts with the environment first but the master feels the impact of the environment after a delay hence resulting in unstable system.




Position with delay

A

Force with delay

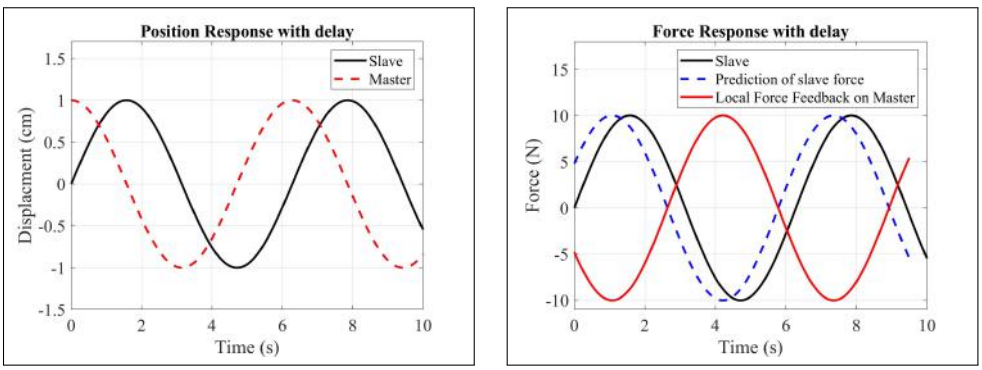
B

Figure 1.2: Problem Statement: Position and Force with Delay

## 1.2. Proposed Solution

However, until there is a virtual duplicate of the slave side environment on the master plant side, any form of local force feedback on the master side may only employ the delayed force response. The master manipulator can apply local force feedback before the real force signal is created on the slave side by using a virtual duplicate of the slave environment. This condition necessitates the use of a good environment model as well as a quick and exact estimation technique on the teleoperation network's slave side.

The goal of this research is to estimate the environment characteristics that will be supplied back to the master for the reconstruction of a virtual environment. In circumstances where the slave system's signal is absent or delayed, this will allow prediction of the actual force felt by the slave system, which should improve the system's overall teleoperation performance as shown in figure 1.3.




Position with delay Force with local force feedback

Figure 1.3: Proposed Solution: Position and Force with Local Force Feedback

# 2. LITERATURE SURVEY

Bilateral control was initially proposed in the 1940s and 1950s, when the very first mechanical master-slave manipulator was introduced by Goertz for radio-active lab-work and then later on electric-servo manipulators with force feedback were used [4].

Figure 2.1: First Mechanical Master-Slave Manipulator
[5]

The mechanically connected systems are directly responsible for the sense of force on the master side. Throughout the 1960s and 1970s, researchers began to look at the influence of delays on force projection [6] [7]. During the mid-1980s, more intelligent control theoretic approaches such as Lyapunov-based analysis began to develop [8]. Control techniques using hybrid approaches [9] and impedance representation [10] for teleoperation systems were introduced in the late 1980s, after improvements in network theory. Spong assured the stability of the closed loop time delay system by applying the passivity theory to the teleoperation for the first time [11]. However, such an energy-based strategy fails to provide sufficient visibility, which is the second condition for teleoperation system success [2].

Furthermore, the proposed passivity strategy was limited to situations with a known and constant time delay. The concept of scattering theory rather than direct power transfer over the network might help avoid wave reflection interference and improve closed loop dynamics Niemeyer and Slotine presented wave variables, it proposed a potential method for establishing both stability and transparency in the presence of continuous time delays of any size [12], [13]. Perhaps there is hardly an entirely satisfactory solution for instances including time-varying delay in bilateral teleoperation. The use of modified wave variables is suggested to offer the requisite stability in force reflecting teleoperation for unexpected transmission delays [14].

In teleoperation systems, modified wave variables have also been found to increase force tracking performance [15]. Yokokohji improved the modified wave variables approach to reduce bilateral teleoperation performance degeneration due to time-varying communication latency [16]. Other research was also done to improve the performance of wave variables. Munir and Book added predictive properties to wave-based teleoperation using a modified Smith Predictor and a Kalman Filter in their research [17]. Their work was particularly significant in terms of experimentation, as it was the first to show that intercontinental teleoperation was possible. [18] contains a complete formalism for wave-based prediction and the application of wave variables to variable time delays. However, in both circumstances, the relationship between the magnitudes of time delay and the system time constant is critical in defining the system's overall performance. Additional position controllers are employed in a recent work [19] to offer steady state position and force tracking for passivity controllers.

Furthermore, [20] looked at the benefits of using local force feedback for improved stability and performance. A few more research involving the use of control theory methodologies have also been conducted. [21] implements an adaptive rule for automated model time delay modification in a Smith Predictor. The revised structure improves the Classical Smith Predictor's performance by lowering the susceptibility to modelling mistakes. [22] presented a different adaptation approach to improve the transparency of the previously proposed impedance reflecting bilateral teleoperation [23].

Slama et al used a generalized predictive control structure to improve the performance of control based on delayed force feedback in their study [24]. In time delayed bilateral control, an optimum method based on $H\infty$ control and $\mu-$synthesis is used to maximize performance criteria [25].

Apart from the traditional power transfer and scattering theory approaches, the recently developed Communication Disturbance Observer (CDOB) appears to solve the stability problem caused by variable delays of any size [26], [27]. For systems with constant and/or time-varying delays, CDOB provides a framework for the use of disturbance observers.

CDOB was used by Ohnishi et al. to achieve bilateral teleoperation [28]. Though CDOB's effectiveness in stabilizing a network delayed motion control system has been demonstrated both theoretically and empirically, position convergence of the slave is degraded, especially when the beginning conditions of the master and slave systems are different. Sabanovic presented an observer-predictor structure with a PD convergence controller and CDOB to ensure position convergence [29]. [30] and [31] provide a quantitative and analytical assessment of the known strategies for solving the time delay problem.

Researchers have recently addressed the use of deep learning algorithms for different robotics fields, Jacky Liang used Deep learning from demonstration (Deep LfD) for automation tasks of dVRK teleoperation for industrial purposes [32].

Typically, deep LfD algorithms use data sets of

- human videos, they do not correspond to the kinematics and capabilities of the robot
- waypoints gathered via time-consuming move-and-record interfaces like teaching pendants or kinaesthetic teaching.

For time-varying delay and uncertainties Parham M. Kebria used neural network adaptive control for teleoperation system [33]. In his study, he suggested an adaptive control technique based on a radial basis function (RBF) neural network for dealing with model uncertainties as well as stabilization in the presence of time-varying delays.

Hitoe Ochi used deep learning algorithm for the scooping motion of the teleoperation, to learn motion taught by human teleoperation, the deep learning software employs a mix of Deep Convolutional Auto-Encoders (DCAE) over picture areas and Recurrent Neural Network with Long Short-Term Memory units (LSTM-RNN) over robot motor angles [34].

# 3. CONTRIBUTION OF THE THESIS

The following studies are carried out as part of this thesis:

- The potential capability of using Learning Algorithm LSTM to estimate the impedance parameters of remote environment contact force in a bilateral teleoperation system.

- The impact of using recursive least squares (RLS) to estimate the parameters of remote environment contact force and render a virtual environment on the operator side in a bilateral teleoperation system.

- Different variants of RLS estimators are implemented and compared against three different impedance models.

- Addition of local force feedback control loop based on the predicted environment using estimated parameters of RLS.

# 4. MOTION CONTROL

## 4.1. Background Algorithms

The analysis and derivations in the following sections are done on a single DOF motion control system for simplicity's sake. The resulting outcome can then be applied to MIMO systems. Plant dynamics for a single DOF motion control system could be calculated using

$$\ddot{q}(t) = \frac{\tau_{ref}(t) - \tau_{dis}(t)}{A_n} \tag{4.1}$$

Where $A_n$ is nominal plant inertia, $\tau_{ref}$ is the input torque and $\tau_{dis}$ is the disturbance torque. In 4.1 the term $\tau_{ref}(t)$ can be assumed a linear multiple of the reference current $i_{ref}(t)$ which yields to the following identity. [35]

$$\tau_{ref}(t) = K_n i_{ref}(t) \tag{4.2}$$

where $K_n$ is the nominal torque constant so by Substituting in (4.1) we get the following

$$\ddot{q}(t) = \frac{K_n i_{ref}(t) - \tau_{dis}(t)}{A_n} \tag{4.3}$$

in which all undesirable effects, including viscous friction $B(q, \dot{q})$, are grouped together by $\tau_{dis}(t)$

$$\tau_{dis}(t) = \Delta A_n \ddot{q}(t) + \Delta K_n i_{ref}(t) + B(q, \dot{q})\dot{q}(t) + G(q(t)) + \tau_{ext}(t) \tag{4.4}$$

### 4.1.1. Disturbance Observer (DOB) and Acceleration Control

The removal of the disturbance torque is critical for the use of acceleration control in the dynamic system shown in equation (4.3). To estimate and cancel the disturbance acting on the system, a disturbance observer (DOB) can be used [36]. A low-pass filter is part of the disturbance observer's internal structure. A disturbance observer with a high filter gain can be built to cancel the disturbance torque as early as feasible. The system's velocity response and current input may be used to calculate the estimated disturbance, which can then be given back to the plant. A velocity observer is used to determine the velocity response from the location data.

However, while a disturbance observer can substantially boost the resilience of a system in many situations, the disturbance may not always be properly compensated owing to the low-pass filter utilized in the structure, resulting in estimate errors. With this in mind, the motion control system in equation (4.3) can be re-formulated as follows, with the inclusion of a disturbance observer.

$$\ddot{q}(t) = \frac{K_n i_{ref}(t) - \delta\tau_{dis}(t)}{A_n} \tag{4.5}$$

where, $\delta\tau_{dis}(t)$ refers to disturbance estimation error $\tau_{dis}(t)$ - $\hat{\tau}_{dis}(t)$ The plant should act as a double integrator system under complete disturbance cancellation. However, due of the DOB output flaw, the estimation error is also doubly integrated. As a result, without a separate control loop, the velocity and position responses may differ from the relevant references, which can be summarized as follows:

$$\begin{aligned} \dot{q}^{res}(t) - \dot{q}^{ref}(t) &= \int_0^t \delta\tau(\zeta)\, d\zeta \\ q^{res}(t) - q^{ref}(t) &= \int_0^t \int_0^t \delta\tau(\zeta)\, d\zeta d\psi \end{aligned} \tag{4.6}$$

Figure 4.1 illustrates the disturbance observer's structure.

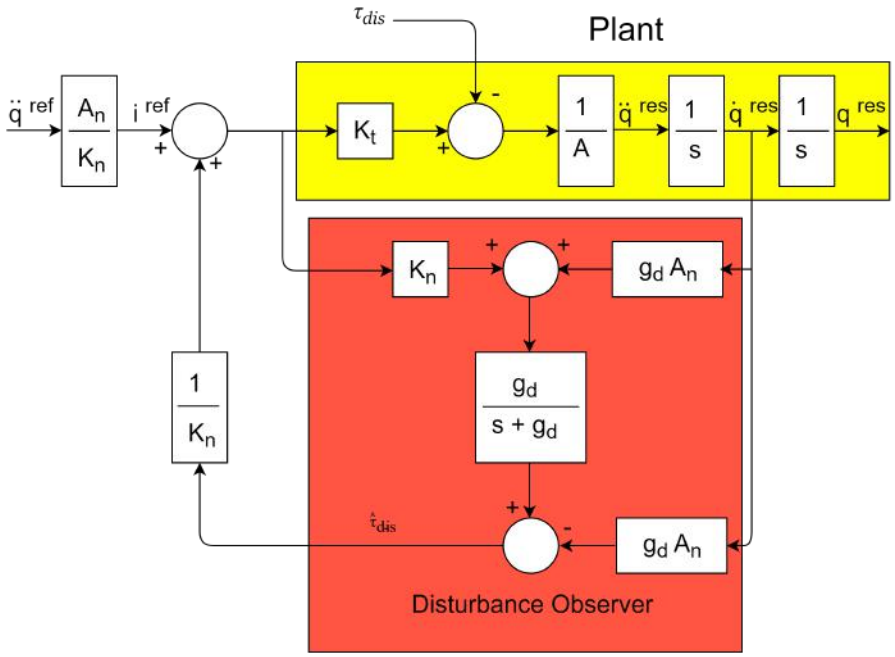



Figure 4.1: Plant with DOB.

### 4.1.2. Reaction Force Observer (RFOB)

In the previous discussion, the expected disturbance torque was supplied back to the system in order to provide stable motion control in the acceleration framework. Furthermore, if the system is well recognized, the external torque acting on it may be approximated by feeding all known torques forward [37]. The external torque exerted on the system may be calculated mathematically as:

$$\hat{\tau}_{ext}(s) = [\tau_{dis} - B(s)sq(s) + G(s)]\frac{g_r}{s + g_r} \tag{4.7}$$

Here the time domain was intentionally converted to Laplace domain for the convenience of the using a low-pass filter.

The nominal torque constant $K_n$, as well as the nominal inertia $A_n$, are all known in advance, because the variance in inertia and torque constant is negligible (i.e. $\Delta K_n, \Delta A_n \approx 0$). The estimation accuracy, like that of the disturbance observer, is determined by the filter gain $g_r$.

Because the force estimate is based on the input current and velocity measurement in this structure, a high gain low-pass filter is used to get a very quick force estimation response. However, bandwidth constraints in force estimates are introduced by filter gain limitations [38]. Figure 4.2 illustrates the reaction force observer.

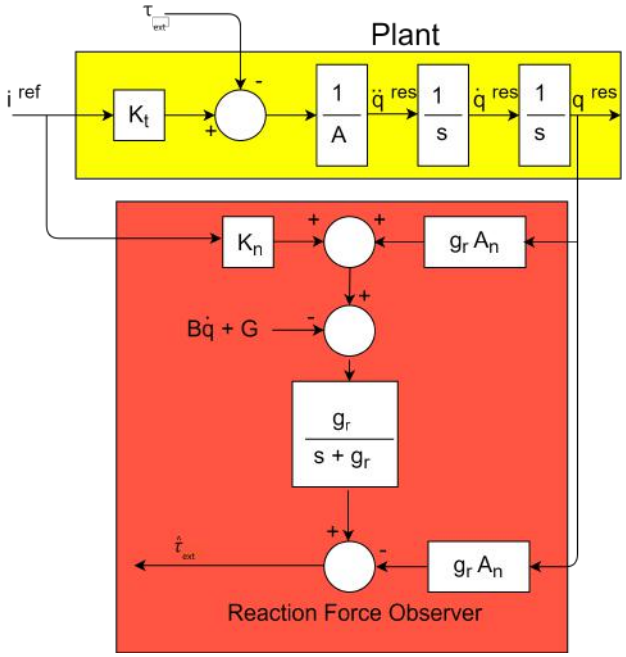



Figure 4.2: Plant with RFOB.

## 4.2. Bilateral Control Without Delay

The analysis and design of bilateral control is examined in this section without taking into account the time delay effect. The structure used in this example was first presented in [39]. The core logic of the investigated structure is based on the conservation of a nominal system, a double integrator plant capable of receiving acceleration references and controller derivation in the acceleration dimension. Robust motion control and acceleration control are being used extensively in the developing and testing.

To get at a nominal plant structure, one must ensure that any undesirable effects acting on the system are minimized. The Disturbance Observer (DOB) is used to attain disturbance rejection. Both the master and slave systems are made more resilient to disturbance in this way. After that, position and force servoing controller derivation is carried out in order to construct the appropriate acceleration references for the plants. Reaction Force Observers (RFOB) are used to monitor force without employing sensors during the implementation.

### 4.2.1. Construction of Bilateral Controller Without Delay

The goal of bilateral control, as previously stated, is for the slave system to follow the position reference provided by the master system and for the master system to reflect the forces felt by the slave system. This problem can be written as follows from a mathematical perspective:

$$q_m = q_s \tag{4.8}$$

$$f_m = -f_s \tag{4.9}$$

A bilateral controller can be developed to satisfy the conditions outlined in the above equation (4.8) and (4.9) . The tracking error can be defined as a linear combination of errors in position and velocity tracking using the position and velocity measurements of master and slave systems. To do this, we must first define an error that synchronizes the movements of the master and slave systems. Therefore, position tracking error can then be stated as:

$$\epsilon_q(t) = (\dot{q}_m(t) - \dot{q}_s(t)) + C(q_m(t) - q_s(t)) \tag{4.10}$$

Where $\epsilon_q(t)$ represent the position error between master and slave, $\dot{q}(t)$ and $q(t)$ represents velocity and displacement of master and slave system.
The following error dynamics can be applied to enforce exponential decay for this error:

$$\dot{\epsilon}_q(t) + K_q\epsilon_q(t) = 0 \tag{4.11}$$

$$(\ddot{q}_m(t) - \ddot{q}_s(t)) + (C + K_q)(\dot{q}_m(t) - \dot{q}_s(t)) + CK_q(q_m - q_s) = 0 \tag{4.12}$$

Where, $K_q$ refers to the exponential decay rate of the position error and $C$ refers to the weight of position in position error. The term $\ddot{q}(t)$ represents the acceleration of master and slave system.
The force tracking error can be expressed as

$$\epsilon_f(t) = (f_m(t) + f_s(t)) \tag{4.13}$$

where $\epsilon_f(t)$ represents the force error between master and slave, by enforcing the following exponentially decaying dynamics we get

$$\ddot{\epsilon}_f(t) + K_y\dot{\epsilon}_f(t) + K_f\epsilon_f(t) = 0 \tag{4.14}$$

Where $\ddot{\epsilon}_f(t)$ and $\dot{\epsilon}_f(t)$ represents the second and first derivative of the slave and master system's force. The coefficient $K_f$ represents the exponential decay rate of the force error. Without loss of generality, one can assume a pure spring environment for the interaction force (i.e. $f_m(t) = K_e q_m(t)$). Hence, the expression given in (4.14) can be recast as follows:

$$K_e(\ddot{q}_m(t) + \ddot{q}_s(t)) + K_e K_y(\dot{q}_m(t) + \dot{q}_s(t)) + K_f(f_m(t) + f_s(t)) = 0 \tag{4.15}$$

Dividing this equation by the common spring constant $K_e$ gives the following expression:

$$(\ddot{q}_m(t) + \ddot{q}_s(t)) + K_y(\dot{q}_m(t) + \dot{q}_s(t)) + \frac{K_f}{K_e}(f_m(t) + f_s(t)) = 0 \tag{4.16}$$

Equations (4.12) and (4.16) set the baseline for the solution of the desired accelerations. Adding (4.12) to (4.16) and dividing by 2 yields to:

$$\begin{aligned} \ddot{q}_m(t) = -0.5(C + K_q)(\dot{q}_m(t) - \dot{q}_s(t)) - 0.5CK_q(q_m(t) - q_s(t)) \\ -0.5K_y(\dot{q}_m(t) + \dot{q}_s(t)) - 0.5\frac{K_f}{K_e}(f_m(t) + f_s(t)) = 0 \end{aligned} \tag{4.17}$$

$$\begin{aligned} \ddot{q}_s(t) = -0.5K_y(\dot{q}_m(t) + \dot{q}_s(t)) - 0.5\frac{K_f}{K_e}(f_m(t) + f_s(t)) \\ +0.5(C + K_q)(\dot{q}_m(t) - \dot{q}_s(t)) + 0.5CK_q(q_m(t) - q_s(t)) = 0 \end{aligned} \tag{4.18}$$

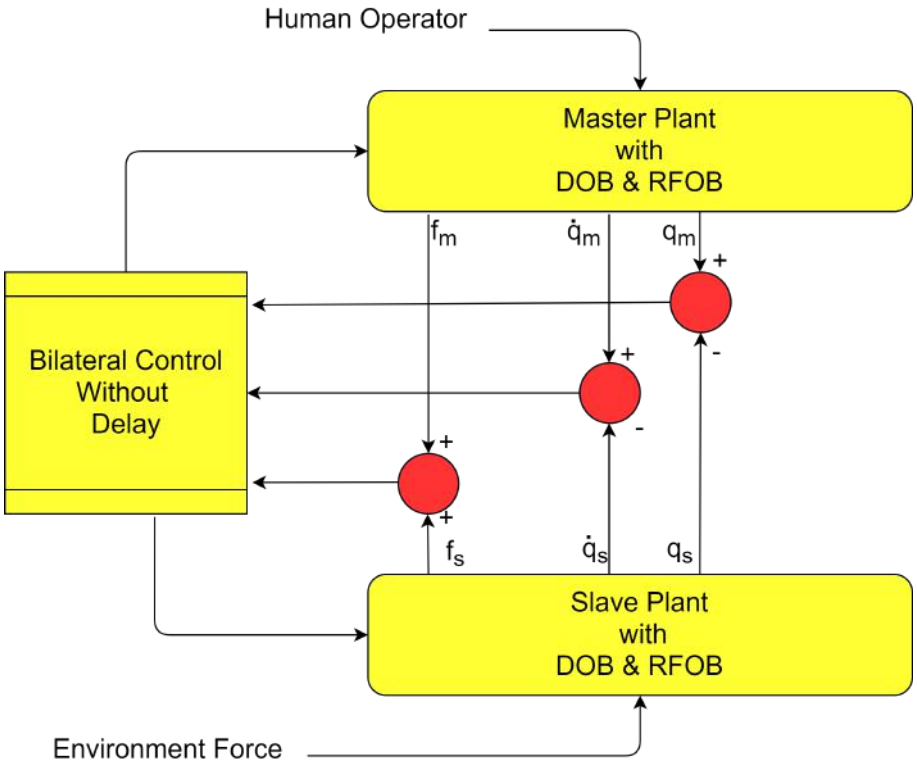



Figure 4.3: Block Diagram of Bilateral Control without Delay.

### 4.3. Motion Control System With Delay

The topic of this section is bilateral teleoperation with time delays. The medium for teleoperation, as previously indicated, is a network (an internet environment), which may create data transmission delays. Because of the network's structure, both the measurements from the distant plant and the control input to the remote side have unknown time delays. The purpose of a controller for time delayed bilateral teleoperation under these conditions should be to provide steady operation with as much transparency as feasible.

In order to have reliable position tracking under time delay, an observer must be able to overcome the measurement delay from the slave side and give on time estimation of the slave motion. The estimator result may then be used by the master side controller to construct the appropriate control input for position tracking.

The implementation of a previously proposed network control mechanism [40] is included in the following study. For this aim, the derivation of the previously described structure is done in a slightly different way. Furthermore, the entire study is performed using a single DOF motion control system. The conclusions of this structure may be applied to MIMO systems without losing their generality. The nominal plant characteristics are assumed to be known. The control input is only affected by network latency, but the measurements are affected by network non-ideality (delay and dynamic distortions).

Assume a single-degree-of-freedom motion control system is to be controlled across a network with unpredictably long time delays in both the measurement and control channels. Due to the nature of delay, both control input and plant measurements will be influenced non-linearly. Plant dynamics for such a system were shown in (4.1) and (4.3) and can be summarized as follows:

$$\ddot{q}(t) = \frac{K_n i_{ref}(t) - \tau_{dis}(t)}{A_n}$$

The plant's total disturbance torque, nominal torque constant, and nominal plant inertia are denoted by $\tau_{dis}(t)$, $K_n$ and $A_n$, respectively.
When the measurement channel has a delay $D_m$ and the slave plant's position and velocity are accessible at a remote site,Only the delayed position and velocity (i.e. $q(t - D_m)$) and $\dot{q}(t - D_m)$) are accessible to the controller.
As a result, an observer must be utilized to offer synchronized, that is, without delays, predictions of remote plant position and velocity. In that way, the objective might be defined as designing an observer based on existing measurements(i.e. $q(t - D_m)$) and $\dot{q}(t - D_m)$) and $K_n i_{ref}$ is the control input, and it has a zero estimate error.
Once the slave plant's actual output has been approximated, it may be utilized in another controller to create the remote plant's input.

### 4.3.1. Construction of Motion Controller With Delay

A second observer may be added to the network to produce an initial estimate of the slave plant velocity, assuming that the slave plant shows nominal behavior with integrated DOB and that the entire impact of measurement delay is regarded as a network disturbance working on the estimator.

$$\tau_{dis}^{nw}(t) = K_n i_{ref}(t) - A_n \ddot{q}(t - D_m) \tag{4.19}$$

Like the usual disturbance observer structure, this network disturbance may be estimated using delayed slave plant velocity and a low pass filter. The estimated network disturbance represents the torque that is supposed to act on the slave plant during the measurement delay. Since the slave plant is required to respond nominally with DOB, the anticipated network disturbance may be routed via the slave plant's nominal inertia and integrated to create the velocity difference that is projected to exist during the delay period. This can be stated mathematically as:

$$\Delta\dot{q}(t) = \int \frac{1}{A_n} \tau_{dis}^{nw}(\zeta)\, d\zeta \tag{4.20}$$

The predicted slave plant velocity is calculated by adding this velocity difference to the delayed velocity, as illustrated below:

$$\hat{\dot{q}}(t) = q(t - D_m) + \Delta\dot{q}(t) \tag{4.21}$$

This observer can follow the slave velocity without delay if the slave system's starting circumstances and the observer output are the same and complete disturbance cancellation is provided. However, the beginning circumstances are not always guaranteed to be equal, thus complete disturbance rejection is not always possible. As a result, more compensation must be included to push the predicted states (i.e. position and velocity) closer to the real slave plant output. As a result, the estimation error may be stated in the following general manner, containing the position and velocity:

$$\epsilon_{\hat{q}}(t) = (\dot{q}_m(t) - \hat{\dot{q}}_s(t)) + C(q_m(t) - \hat{q}_s(t)) \tag{4.22}$$

The following error dynamics can be imposed on the system to compel an exponential convergence of error to zero:

$$\dot{\epsilon}_{\hat{q}}(t) + K_q\epsilon_{\hat{q}}(t) = 0 \tag{4.23}$$

By replacing the error supplied in (4.22) into (4.23) and collecting the real and estimated parameters together, the following condition for the observer and estimator may be achieved:

$$(\ddot{q}_m(t) - \hat{\ddot{q}}_s(t)) + (C + K_q)(\dot{q}_m(t) - \hat{\dot{q}}_s(t)) + CK_q(q_m - \hat{q}_s) = 0 \tag{4.24}$$

Keeping in mind that a scaler can convert the acceleration reference to the current reference, the expression provided in equation (62) can be summarized as follows:

$$\hat{i}_{con} = -(K_{cp}\hat{q}_s + K_{cd}\hat{\dot{q}}_s) \tag{4.25}$$

$$i_{con} = -(K_{cp}q_s + K_{cd}\dot{q}_s) \tag{4.26}$$

where $K_{cp}$ and $K_{cd}$ are convergence parameters.

The addition of convergence terms to both the estimator and the slave plant ensures that the estimation of the remote plant response is synchronized on time with the actual slave output. The only thing left to do now is build a controller that will create the control current, which will then be transferred to the slave system after a control channel delay $D_c$.

When employing a PD controller to track master position reference, the following control current may be used as the control input to both the slave system and the observer.

$$i_{ref}(t) = K_{vel}(\dot{q}_m(t) - \hat{\dot{q}}_s(t)) + K_{pos}(q_m(t) - \hat{q}_s(t)) \tag{4.27}$$

Now we have the whole observer-controller structure for position control with a time delay.

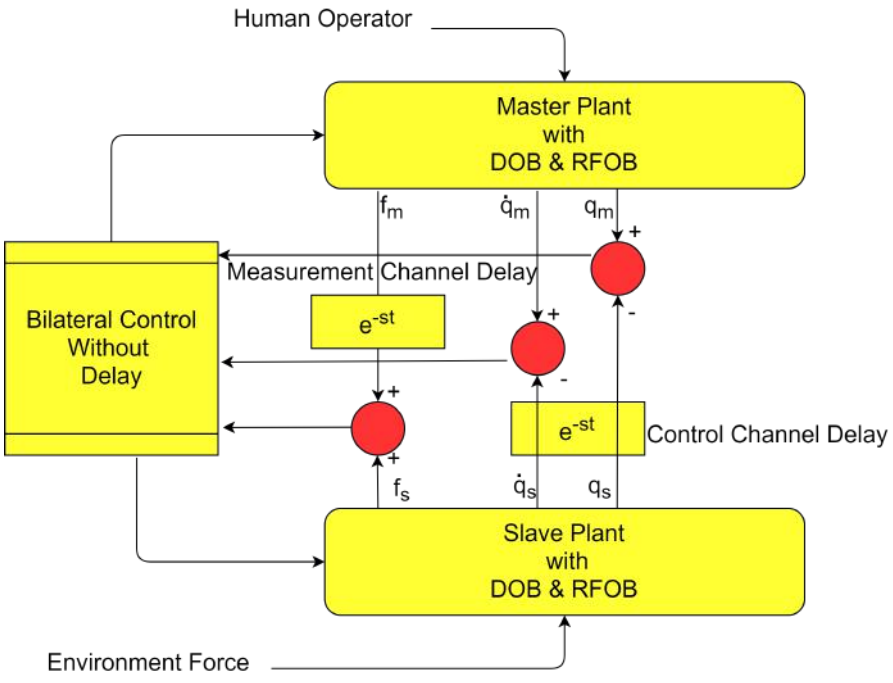



Figure 4.4: Block Diagram of Motion Control with Delay.

## 4.4. Motion Control System With Delay and Local Force Feedback

In this chapter bilateral controller with time delay and local force feedback is focused. The force felt on the slave system is fed back to the master system in order to create a local force loop. When the slave makes a contact with environment, the estimated force calculated from the environment parameters render on the master side as virtual environment reconstruction hence creating a close loop system.

The overall structure of the system seems similar motion control system without delay but here instead of slave position and velocity, estimated position and velocity are used and the force felt is also not the slave but the estimated force that will render on the master system.

### 4.4.1. Construction of Bilateral Controller With Delay and Local Force Feedback

For such a close loop system, tracking error can be defined as a linear combination of errors in position and velocity tracking using the position and velocity measurements of master and slave systems, position tracking error can be stated as:

$$\epsilon_q(t) = (\dot{q}_m(t) - \hat{\dot{q}}_s(t)) + C(q_m(t) - \hat{q}_s(t)) \tag{4.28}$$

where $\hat{\dot{q}}_s(t)$ and $\hat{q}_s(t)$ refers to the estimated velocity and position. The following error dynamics can be applied to enforce exponential decay for this error:

$$\dot{\epsilon}_q(t) + K_q\epsilon_q(t) = 0 \tag{4.29}$$

$$(\ddot{q}_m(t) - \ddot{q}_s(t)) + (C + K_q)(\dot{q}_m(t) - \hat{\dot{q}}_s(t)) + CK_q(q_m - \hat{q}_s) \tag{4.30}$$

Where, $K_q$ refers to the exponential decay rate of the position error and $C$ refers to the weight of position in position error
The force tracking error can be expressed as

$$\epsilon_f(t) = (f_m(t) + \hat{f}_s(t)) \tag{4.31}$$

by enforcing the following exponentially decaying dynamics we get

$$\ddot{\epsilon}_f(t) + K_y\dot{\epsilon}_f(t) + K_f\epsilon_f(t) = 0 \tag{4.32}$$

Without loss of generality, one can assume a pure spring environment for the interaction force (i.e. $f_m(t) = K_e q_m(t)$)Hence, the expression given in (4.32) can be recast as follows:

$$K_e(\ddot{q}_m(t) + \ddot{q}_s(t)) + K_e K_y(\dot{q}_m(t) + \hat{\dot{q}}_s(t)) + K_f(f_m(t) + \hat{f}_s(t)) = 0 \tag{4.33}$$

Dividing this equation by the common spring constant $K_e$ gives the following expression:

$$(\ddot{q}_m(t) + \ddot{q}_s(t)) + K_y(\dot{q}_m(t) + \hat{\dot{q}}_s(t)) + \frac{K_f}{K_e}(f_m(t) + \hat{f}_s(t)) = 0 \tag{4.34}$$

Equations (4.30) and (4.34) set the baseline for the solution of the desired accelerations. Adding (4.30) to (4.34) and dividing by 2 yields to:

$$\ddot{q}_m(t) = -0.5(C + K_q)(\dot{q}_m(t) - \hat{\dot{q}}_s(t)) - 0.5CK_q(q_m(t) - \hat{q}_s(t)) \\ -0.5K_y(\dot{q}_m(t) + \hat{\dot{q}}_s(t)) - 0.5\frac{K_f}{K_e}(f_m(t) + \hat{f}_s(t)) = 0 \tag{4.35}$$

$$\ddot{q}_s(t) = -0.5K_y(\dot{q}_m(t) + \hat{\dot{q}}_s(t)) - 0.5\frac{K_f}{K_e}(f_m(t) + \hat{f}_s(t)) \\ +0.5(C + K_q)(\dot{q}_m(t) - \hat{\dot{q}}_s(t)) + 0.5CK_q(q_m(t) - \hat{q}_s(t)) = 0 \tag{4.36}$$

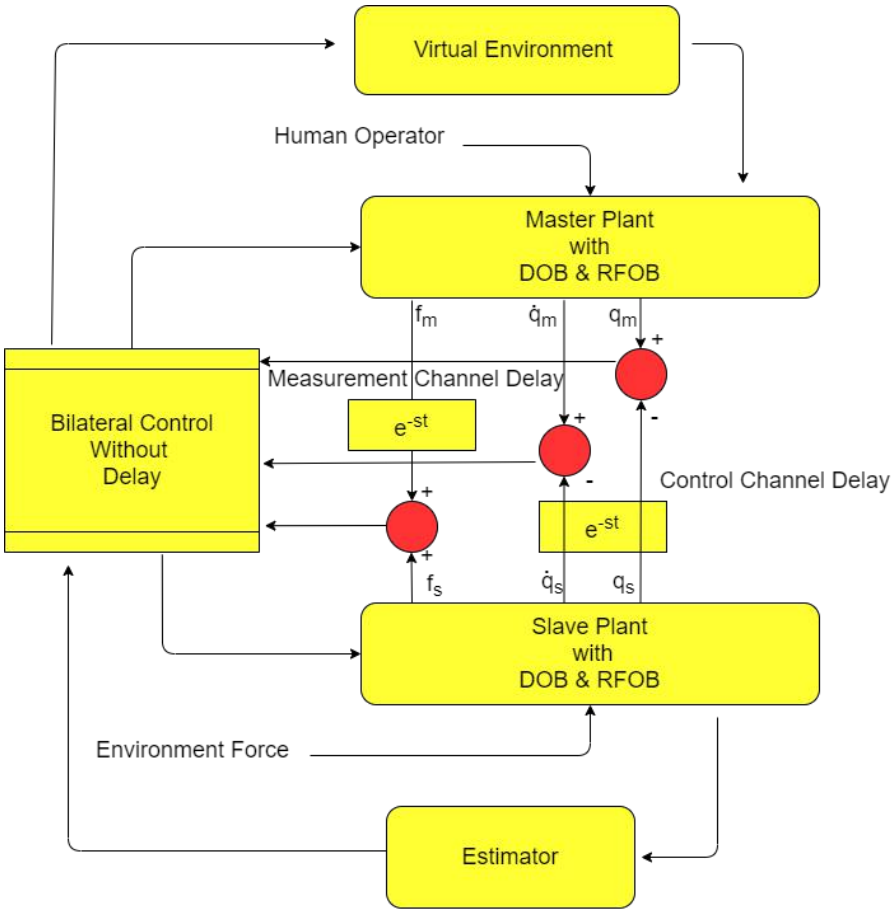



Figure 4.5: Block Diagram of Motion Control with Delay and local force feedback.

# 5. ENVIRONMENT ESTIMATION METHODS

### 5.0.1. Deep Learning Algorithm

#### 5.0.1.1. Introduction to Recurrent Neural Network.

A Recurrent Neural Network is an artificial neural network,RNNs are a sort of neural network that is both strong and robust. They are one of the most promising algorithms in use since they are the only ones having an internal memory. They use memory by taking information from previous output to influence a current input and output. This is why they're the chosen algorithm for time series, voice, text, financial data, audio, video, weather, and many other types of sequential data. RNNs have been criticized for two reasons Vanishing gradient and Exploding gradient, in vanishing gradient the weights approaches to a very small value and for exploding gradient the weight approaches to a very large value hence resulting in large error in the back-propagation. An unrolled RNN cell is shown in Figure 5.1.

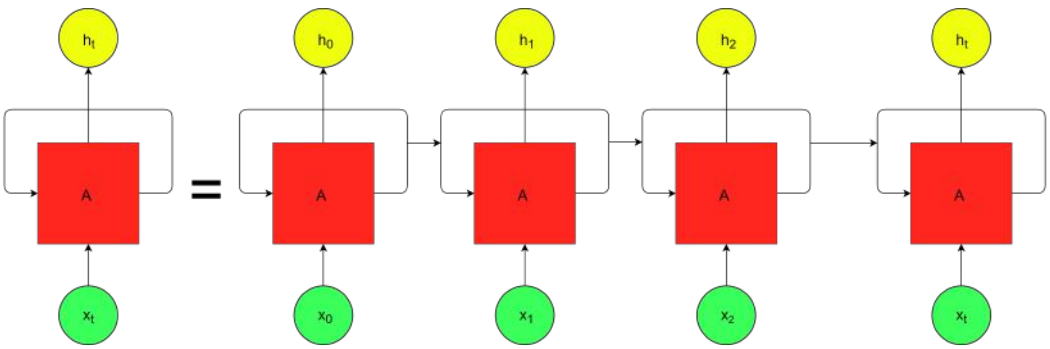



Figure 5.1: RNN Cell

#### 5.0.1.2. Long Short Term Memory (LSTM).

The LSTM architecture was developed by Sepp Hochreiter and Juergen Schmidhuber as a solution to the vanishing gradient problem [41]. Long short-term memory networks are a type of recurrent neural network that expands the memory capacity. This memory may be thought of as a gated cell, with gated indicating that the cell selects whether or not to store or erase information (i.e., whether or not to open the gates) based on the value it gives to the data. Weights, which are also learnt by the algorithm, are used to allocate importance. This basically implies that it learns what information is important over time and what information is not.

LSTM is well-suited to detect, analyze, and predict time series with unpredictable time delays. The model is trained via back-propagation. In the deep layers of the neural network, LSTMs employ cells with three gates: an input gate $i_t$, an output gate $o_t$, and a forget gate $f_t$. These gates control the flow of data required for the network's output to be predicted. An unrolled LSTM cell is shown in Figure 5.2

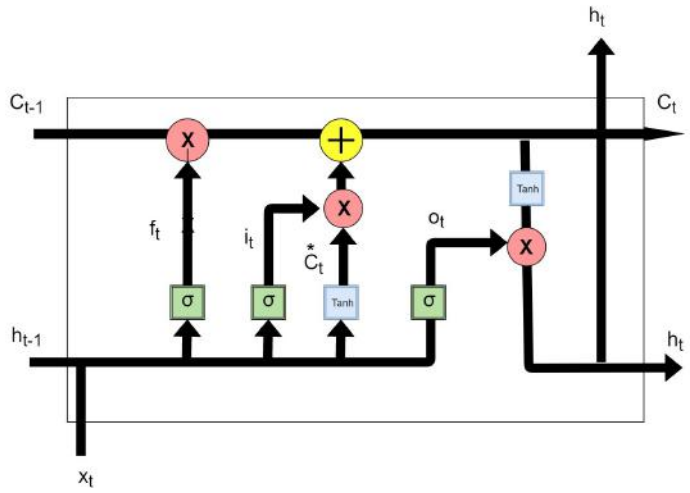



Figure 5.2: LSTM Cell

The gates corresponding to LSTM Cell are as followed.

$$f_t = \sigma * ( W_f * x_t + U_f * h_{t-1} + b_f) \tag{5.1}$$

$$i_t = \sigma * ( W_i * x_t + U_i * h_{t-1} + b_i) \tag{5.2}$$

$$c_t^* = Tanh * ( W_c * x_t + U_c * h_{t-1} + b_c) \tag{5.3}$$

$$o_t = \sigma * ( W_c * x_t + U_o * h_{t-1} + b_o) \tag{5.4}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot c_t^* \tag{5.5}$$

$$h_t = o_t \odot (Tanh * c_t) \tag{5.6}$$

In the above equations, Vectors are represented by lowercase variables. Matrices $W_q$ and $U_q$ contains the weights of the input and recurrent connections. The subscript $\mathbf{i}_t$, $\mathbf{o}_t$, $\mathbf{f}_t$ and $\mathbf{c}_t$ corresponds to input gate, output gate , forget gate and memory cell depending on the activation being calculated. We will use "vector notation" in this part $\mathbf{c}_t \in \mathbb{R}^h$ is not just one cell of one LSTM unit, but contains $\mathbf{h}$ LSTM unit's cells. The initial values are $c_o = 0$ and $h_o = 0$ and the operator $\odot$ denotes the Hadamard product (element-wise product). The subscript t indexes the time step , $\mathbf{x}_t \in \mathbb{R}^d$ is the input vector and $\mathbf{h}_t \in \mathbb{R}^h$ is the hidden output. $W \in \mathbb{R}^{h \times d}$ , $U \in \mathbb{R}^{h \times h}$ and $b \in \mathbb{R}^h$ where the superscripts $\mathbf{d}$ and $\mathbf{h}$ refer to the number of input features and number of hidden units, respectively.

$\mathbf{F}_t \in \mathbb{R}^h$ It is the forget gate that determines whether data should be preserved or destroyed. The sigmoid function conveys information from the previous hidden state as well as current input information. The numbers range from 0 to 1. The closer you go to zero, the more you forget; the closer you get to one, the more you remember.

$\mathbf{I}_t \in \mathbb{R}^h$ is the input gate it is used to update the values coming from sigmoid as memory and from tanh as candidate values($\mathbf{C}_t^* \in \mathbb{R}^h$) for updating the values. To begin, we mix

the prior hidden state with the current input using a sigmoid function. This controls which values will be modified by adjusting the values to be between 0 and 1. You also submit the hidden state and current input into the tanh function to compress values between -1 and 1 to assist manage the network. After that, the sigmoid output is multiplied by the tanh output. The sigmoid output will select which tanh output information should be maintained.

$\mathbf{C}_t \in \mathbb{R}^h$ is the cell state that has sufficient information to compute the cell's status. In a point-by-point manner, the cell state is multiplied by the forget vector. This has the ability to drop values in the cell state if multiplied by values near to 0. Then, on the output of the input gate, we do a point-wise addition, which changes the cell state to new values that the neural network considers relevant. As a result, a new cell state has emerged.

$\mathbf{O}_t \in \mathbb{R}^h$ is the output gate it determines the next concealed state to be. It's crucial to keep in mind that the hidden state incorporates information from previous inputs. The concealed state is also used to make predictions. To begin, we mix the prior hidden state with the current input using a sigmoid function. The tanh function receives the newly adjusted cell state. We multiply the tanh output with the sigmoid output to decide what information the hidden state should contain. The output is the hidden state. The changed cell state and hidden values are then carried over to the next time step.

The core concept in LSTMs is the cell state, the horizontal line running through the top of the LSTM Cell. There are just a few small linear interactions as it passes down the entire chain which makes it really convenient for data to simply flow along it without being altered.

### 5.1. Recursive Least Squares (RLS)

Gabor was the first to use a Volterra series to come up with the notion of a non-linear adaptive filter in 1954 [42] . The LMS algorithm, also known as the Widrow-Hoff rule, was developed by Widrow and Hoff in 1959 as part of their research on a pattern recognition system known as the adaptive linear element [43]. In that LMS algorithm are based on error-correction learning, the LMS algorithm is similar to Rosenblatt's perceptron [44]. During the early years of neural networks, they both appeared about the same time in the late 1950s. Today, the significance of Rosenblatt's perceptron is mostly historical.

The LMS algorithm, on the other hand, has stood the test of time.Channel equalization, system identification, predictive deconvolution, spectrum analysis, signal detection, noise cancellation, and beam forming are just a few of the disciplines where adaptive filters are used.The LMS algorithm is essentially a stochastic gradient algorithm, which implies that the gradient of the error performance surface with respect to the free parameter vector varies randomly from iteration to iteration. Because of this stochasticity, which is compounded by the existence of non-linear feedback, a full convergence study of the LMS algorithm is a complex mathematical effort.

The LMS method achieves simplicity of implementation by employing instantaneous estimations of

- the input signal vector's autocorrelation matrix and
- the input vector's cross-correlation vector with the intended response.

The recursive least-squares (RLS) approach, on the other hand, uses continually updated estimates of these two parameters that go all the way back to the start of the adaptive process. As a result, the RLS algorithm has the following characteristics.

- Convergence rate that is often an order of magnitude quicker than that of the

LMS algorithm.

- Convergence rate that is independent of the eigenvalue dispersion of the input vector's correlation matrix.

### 5.1.1. Modelling Environment Force

#### 5.1.1.1. Model-1.

The standard linear impedance model of the force represented by a mass, spring, and damper system is chosen as the first contact force model. Because it is the most often used linear representation of contact impedance in numerous applications, this model is chosen. The linear impedance model is mathematically represented as:

$$F_{env} = m\delta\ddot{q} + b\delta\dot{q} + k\delta q \tag{5.7}$$

where $\delta q = q_r - q_e$ with $q_r$ and $q_e$ representing the robot end effector position and the environment contact position, respectively. This force model is assumed to be valid only for pushing actions (i.e. $\|q_r\| \geq \|q_e\|$). In (5.7), the parameters $m$, $b$ and $k$ respectively stand for the mass of the material displaced during the penetration of the end effector, the damping constant and the spring constant, respectively.

#### 5.1.1.2. Model-2.

The displaced mass during the contact has a very small magnitude in the great majority of force control applications, and the displacement acceleration (i.e. $\delta\ddot{q}$) is also very small. As a result, contact forces are often dominated by springs, with a minor but significant contribution from the damping effect, especially during transient motion. Furthermore, in force models, the assumption of linear behavior of the spring term might limit the application range to a very narrow range of environment displacement. Considering those details, the second impedance model is selected as a hardening (i.e. non-linear) spring and damper model.

$$F_{env} = b\delta\dot{q} + k\delta q + \nu(\delta q)^3 \tag{5.8}$$

Here, similar to (5.7), $b$ represents the damping constant while $k$ and $\nu$ stand for the linear and non-linear spring constants.

**Γρορ5.1.1.3. Model-3.**

The linear impedance and non-linear spring damper models have been shown to accurately describe the contact force. Both of them, however, have three factors that must be calculated. Given the difficulty of implementing estimate methods in real-time, accuracy can be sacrificed and the number of unknown parameters reduced to two. Referring back to the discussion above, the most likely parameter to drop for the attenuation of the unknown parameter set is the displaced mass ($\delta m \approx 0$). With this assumption in hand, one can note the third impedance model as a simple liner spring damper model as follows

$$F_{env} = b\delta\dot{q} + k\delta q \tag{5.9}$$

### 5.1.2. Estimating Environment Force

For a number of system identification applications, least squares-based techniques have been widely employed. For systems with static parameters, the batch least squares technique is typically favored as long as the input and output data sets are known. The batch least squares estimate approach may be adapted to a recursive algorithm for systems with static parameters and dynamic input and output data sets, generating the recursive least squares (RLS) method. The RLS approach, on the other hand, is not ideal for systems with dynamic unknown parameters because of the memory from previous data inputs. To solve the memory problem, a forgetting factor is added that weights previous input data and provides an approach for estimating time-varying parameters in a linear system.

Three RLS variations are investigated as estimate methods for the environment models presented in the preceding part in this study. The intermediate phases of those algorithms, as well as analyses of the minor changes between them, are described in the following sections. The detailed derivations, on the other hand, are beyond the scope of this study and the reader is referred to the reference [45]. All three of those algorithms are derived for a linear system of the form:

$$\Phi\theta = Y \tag{5.10}$$

where $\Phi \in \mathbb{R}^{m\times n}$ is usually called the state transition matrix, $\theta \in \mathbb{R}^{n\times 1}$ is the vector of unknowns to be estimated, and $Y \in \mathbb{R}^{m\times 1}$ stands for the vector of output measurements. The distinction between the estimators tested in this study lies in the selection of the measurement vector and are listed with details in the following sections:

#### 5.1.2.1. Estimator-1: Single Value Recursive Least Squares.

For the sake of simplicity in real-time control loop computations, the first variation of the RLS estimator covers a single unit measurement vector (i.e. $Y \in \mathbb{R}^{1\times 1}$) . The single value RLS method is governed by the following equations:

$$\hat{\theta}_{i+1} = \hat{\theta}_i + K_{i+1}[Y_{i+1} - \Phi_{i+1}^T\hat{\theta}_i] \tag{5.11}$$

$$K_{i+1} = P_{i+1}\Phi_{i+1}\lambda, \tag{5.12}$$

$$P_{i+1} = P_i - \frac{P_i\Phi_{i+1}\Phi_{i+1}^T P_i}{\lambda^{-1} + \Phi_{i+1}^T P_i \Phi_{i+1}} \tag{5.13}$$

where $\hat{\theta}_i \in \mathbb{R}^{n\times 1}$ is the estimated parameters vector at the $i^{\text{th}}$ iteration, $\Phi_i^T \in \mathbb{R}^{1\times n}$ is the regressor vector, $K_i \in \mathbb{R}^{n\times 1}$ is the gain vector, $P_i \in \mathbb{R}^{n\times n}$ is the covariance matrix, and $\lambda$ is the scalar forgetting factor. The single value estimator has the advantage of being simple to implement and having a minimal computing demand. Due to the fact that this version of the estimator only accepts one measurement data every iteration,

the estimation results will be susceptible to noise in the measured data.

#### 5.1.2.2. Estimator-2: Averaged Parameter Recursive Least Squares.

The second RLS version tested in the context of this study is the averaged parameter RLS. This algorithm only differs from the single value RLS in the regressor and measurement vectors, where each element in those vectors is simply the average of the most recent $r$ points. Thus, one can use the same equations from (5.11)-(5.13) by just modifying the regressor and the measurement vectors as:

The averaged parameter RLS is the second RLS variant investigated in this study. The sole difference between this technique and the single value RLS is that each element in the regressor and measurement vectors is just the average of the most recent $r$ points. Thus, by just changing the regressor and measurement vectors, one may utilize the identical equations from(5.11)-(5.13) .

$$\Phi_{i+1} = \frac{\Phi_{i+1} + \Phi_i + \cdots + \Phi_{i+1-r}}{r} \tag{5.14}$$

$$Y_{i+1} = \frac{Y_{i+1} + Y_i + \cdots + Y_{i+1-r}}{r} \tag{5.15}$$

The averaged parameter RLS has the benefit of being significantly more resilient against noise in the observed signals. This approach, however, has the problem of obtaining estimated parameters with a $r/2$ sample delay. However, because the estimator will be utilized for time-delayed teleoperation applications, the extra latency can be accepted in circumstances where estimating precision is more critical.

#### 5.1.2.3. Estimator-3: Sliding Window Recursive Least Squares.

The last version of the estimators is the so-called sliding window RLS algorithm. This algorithm differs in the way that rather than one, it takes $r$ rows of data per each iteration, where $r$ is the window size (i.e, the regressor, $\Phi_i^T$ is now an $r \times n$ matrix

containing the most recent $r$ data points). In other words, the sliding window method employs batch least squares across the system's most recent $r$ data. The memory effect will be restricted in this technique due to the window's sliding characteristic. As a result, an extra forgetting factor is not included in this version of the estimator for ease of implementation. The following are the governing equations for sliding window RLS:

$$\hat{\theta}_{i+1} = \hat{\theta}_i + K_{i+1}[Y_{i+1} - \Phi_{i+1}^T \hat{\theta}_i] \tag{5.16}$$

$$K_{i+1} = P_{i+1}\Phi_{i+1} \tag{5.17}$$

$$P_{i+1} = P_i - P_i\Phi_i^T \left(I + \Phi_i P_i \Phi_i^T\right)^{-1} \Phi_i P_i \tag{5.18}$$

with $Y_i \in \mathbb{R}^{r\times 1}$, $\Phi_i^T \in \mathbb{R}^{r\times n}$, $K_i \in \mathbb{R}^{n\times r}$ and $P_i \in \mathbb{R}^{n\times n}$. The advantage of this method is again the averaging effect of the last $r$ measurements, which enhances the estimation precision with noisy data. The fundamental difference of the sliding window RLS with that of the averaged parameter RLS is that the estimated output is not delayed in the sliding window approach. However, as this method requires several operations with relatively big matrices, it is computationally less efficient than the first two estimation methods.

# 6. EXPERIMENTS AND VALIDATION

## 6.1. Setup Details

Verification on an experimental system consisting of two linear motors is used to illustrate all of the material presented in this thesis. The experimental platform consisted of two Faulhaber LM2070 series direct drive linear actuators and is integrated with Renishaw RGH22 incremental encoders having $0.5\mu m$ resolution. To drive the system in current control mode, electrical connections are made. The usage of printed circuit boards helps to reduce noise. As a result, the total accuracy of the system was roughly $0.5\mu$m. MATLAB-Simulink environment and Automation-Studio were used for implementation of software and real-time processing supported by a B&R X20 CP1586 controller with extendible PLC codes compiled in the C framework. The B&R controller was designed with a 5 KHz sampling frequency. The measurement of the contact force, on the other hand, is made using a reaction force observer (RFOB) algorithm [46]. A picture of the experimental setup is provided in Figure 6.1

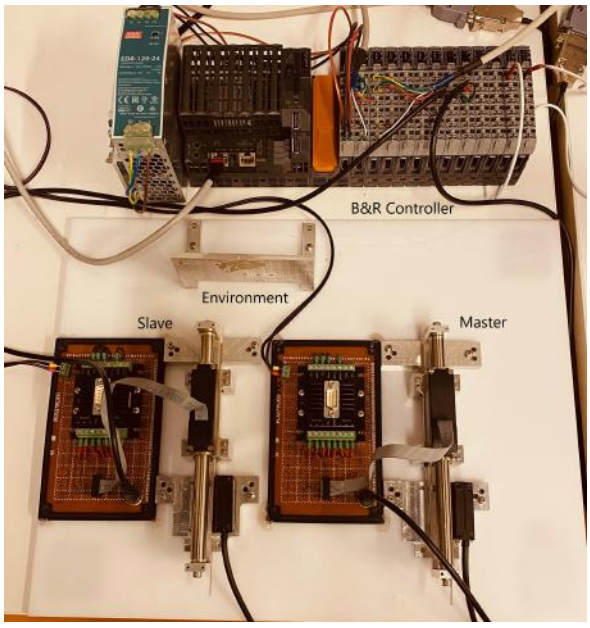
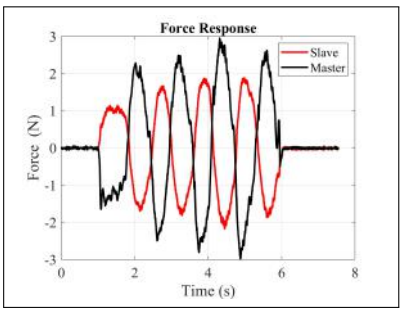



Figure 6.1: Bilateral teleoperation system.

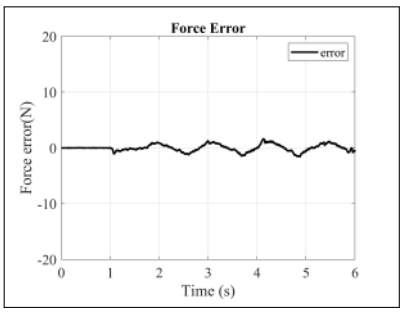
## 6.2. Results

### 6.2.1. Bilateral Control Without Delay

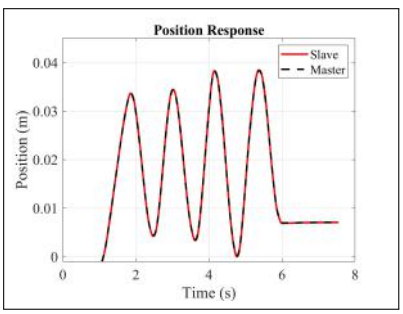
To accurately build a bilateral controller, recognizing the position, velocity and force acted on the actuator is necessary. Position values are recorded by the encoder having $0.5\mu m$ resolution and and velocity is calculated from it. For recording force values without and physical sensors, RFOB was used.



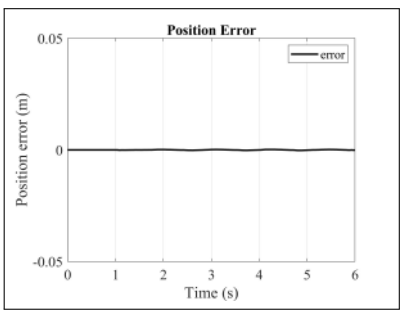

Exp1: Force Response



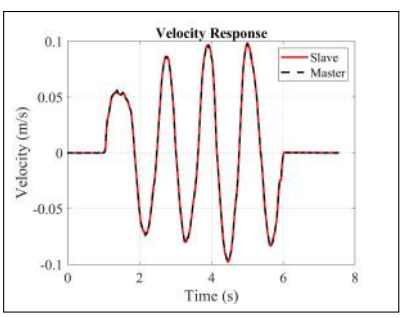

Exp1: Force Response Error



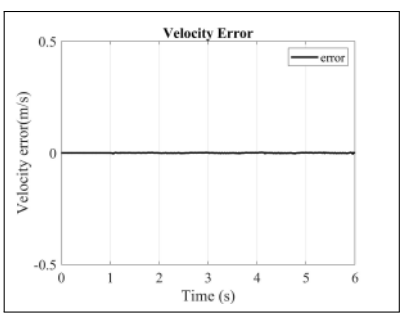

Exp1: Position Response




Exp1: Position Response Error

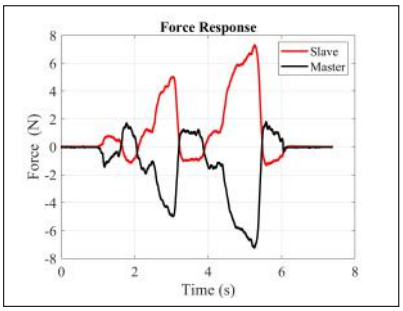



Exp1: Velocity Response

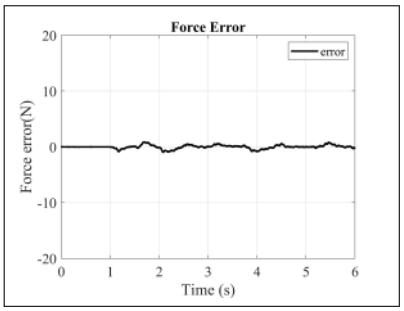



Exp1: Velocity Response Error

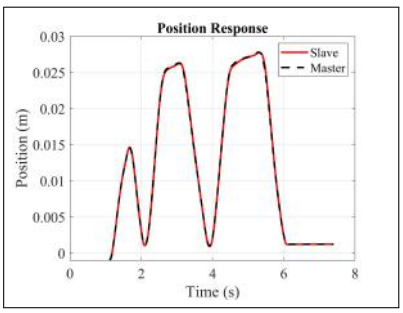
Figure 6.2: Experiment Results 1: Force Response with error (first row), Position Response with error (second row) and Velocity Response with error(third row)



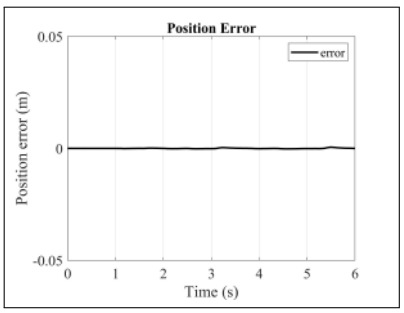

Exp2: Force Response



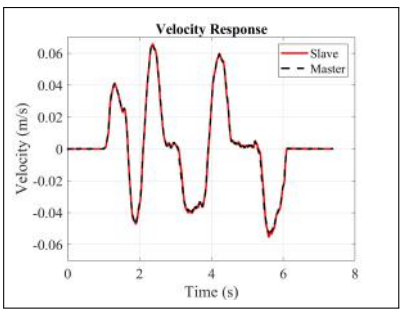

Exp2: Force Response Error



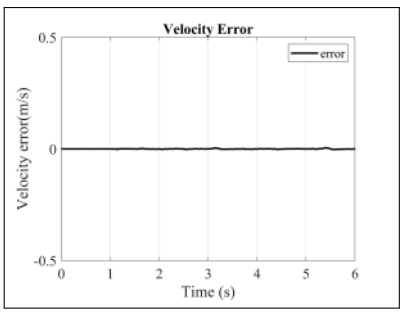

Exp2: Position Response




Exp2: Position Response Error




Exp2: Velocity Response

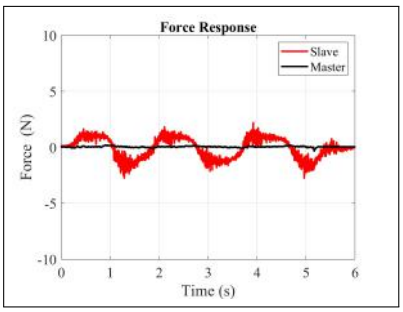



Exp2: Velocity Response Error

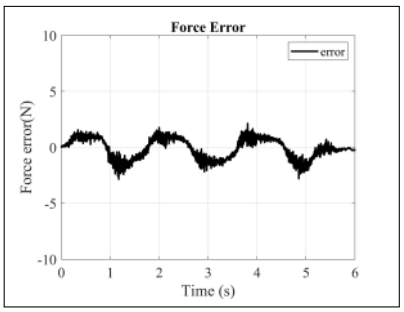
Figure 6.3: Experiment Results 2: Force Response with error (first row), Position Response with error (second row) and Velocity Response with error(third row)

### 6.2.2. Motion Control System With Delay

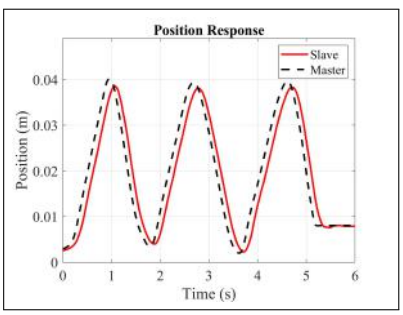
By ensuring the stability and transparency, precise and accurate bilateral controller with delay of $300ms$ was established having a accuracy of $0.5\mu$m. An arbitrary reference was given to the master for tracking delayed position and velocity. The slave was in free motion until 3 second and then was interacted with the soft environment as illustrated in figure 6.2 and in figure 6.5 the slave interacted with the hard environment after 6 seconds



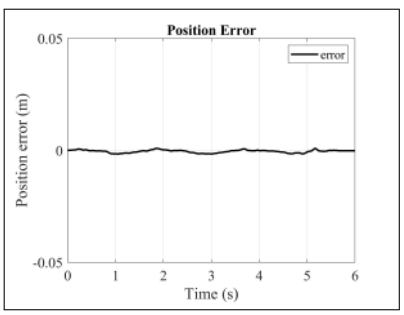

Exp2: Force Response



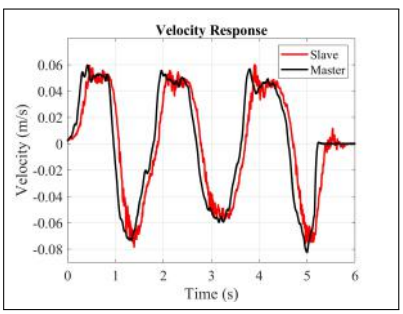

Exp2: Force Response Error



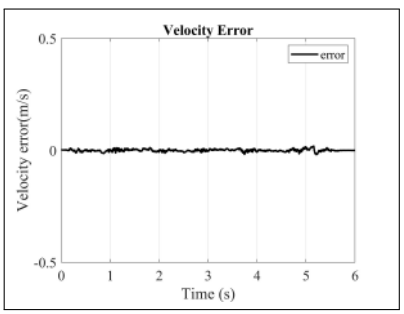

Exp2: Position Response




Exp2: Position Response Error

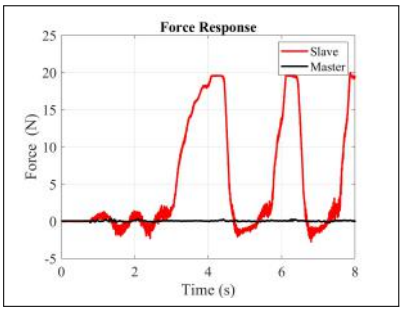



Exp2: Velocity Response

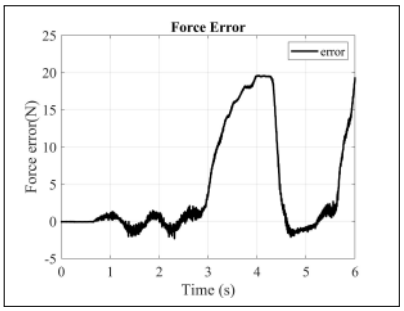



Exp2: Velocity Response Error

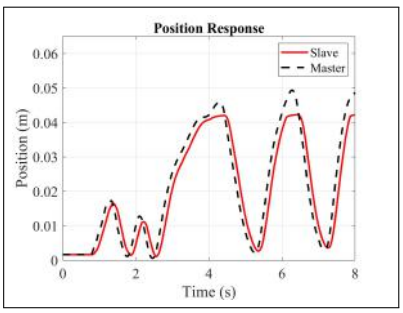
Figure 6.4: Experiment Results 2: Force Response with error (first row), Position Response with error (second row) and Velocity Response with error(third row)



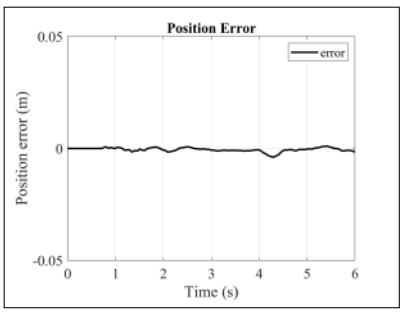

Exp2: Force Response



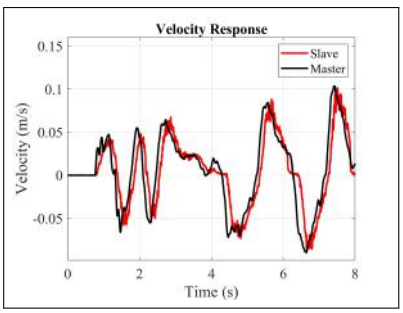

Exp2: Force Response Error



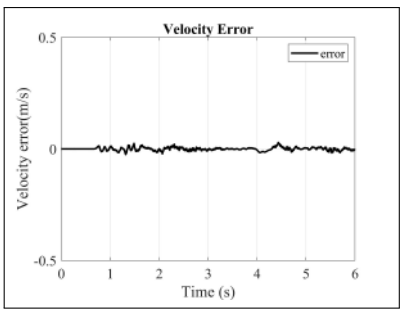

Exp2: Position Response




Exp2: Position Response Error




Exp2: Velocity Response




Exp2: Velocity Response Error

Figure 6.5: Experiment Results 2: Force Response with error (first row), Position Response with error (second row) and Velocity Response with error(third row)

### 6.2.3. LSTM Results and Discussion

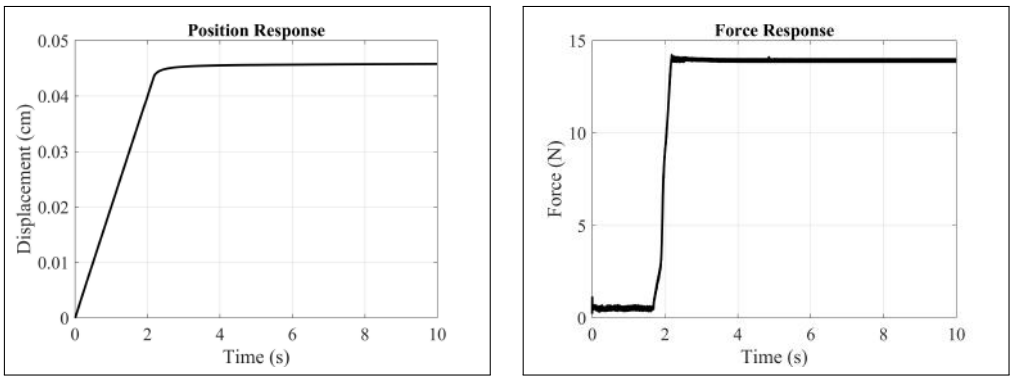
#### 6.2.3.1. Data collection.

Experiments were recorded in order to acquire the real data needed for the simulation purpose. The data was collected with five different environments and different velocity references a shown in the table (6.1)

| Amp. | Freq. | Env 1 | Env 2 | Env 3 | Env 4 | Env 5 |
|---|---|---|---|---|---|---|
| 0.035 | 0.10 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 0.20 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 0.30 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 0.40 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 0.50 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 0.75 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 1.00 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 1.50 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 2.00 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.035 | 3.00 | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.010 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.020 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.030 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.050 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.075 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.100 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.250 | - | Sponge soft | Sponge Hard | PLA Plastic | Soft Plastic | Mild Plastic |
| 0.500 | - | Sponge soft | Hard soft | PLA Plastic | Soft Plastic | Mild Plastic |

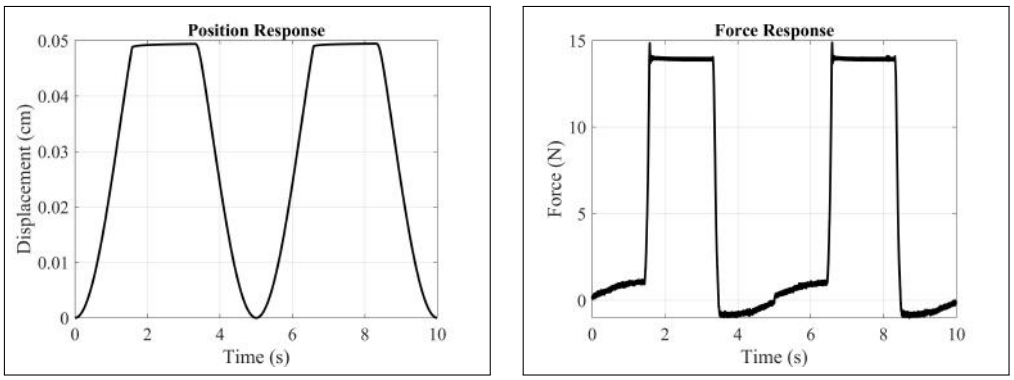
Table 6.1: Real Data Collected For Simulation.

Recognizing the location of the environment in relation to the actuator is required to appropriately estimate the environment force. As a result, the force response of the RFOB is thresholded to identify whether there is contact with the environment, as indicated by [47]. The environment contact position is defined as the location of the motor when the force response first exceeds the threshold value. Numerous experiments were carried out to record the real data for simulation. A step force input reference was used in several experiments with single penetration and multiple penetration. The purpose of such a force reference is to monitor the performance of the chosen force model and Learning Algorithm when the force response changes suddenly.



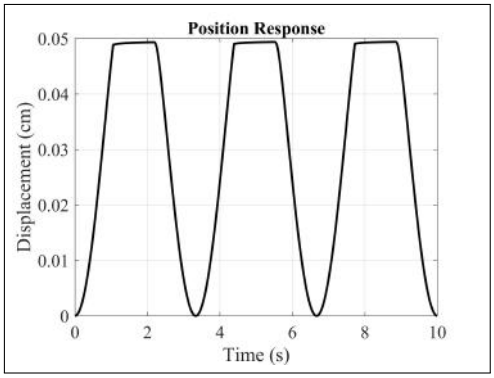

Exp-1-LSTM: Position Exp-1-LSTM: Force

Figure 6.6: Experiments Results1-LSTM: Force Responses and Position Responses

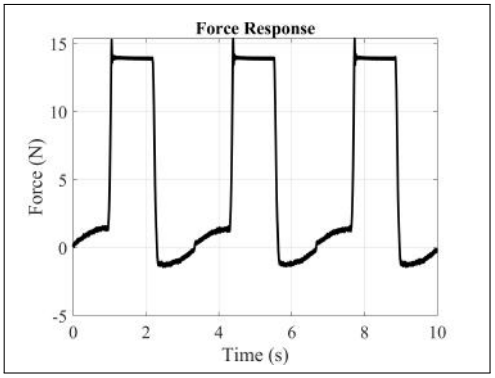



Exp-2-LSTM: Position Exp-2-LSTM: Force

Figure 6.7: Experiments Results2-LSTM: Force Responses and Position Responses




Exp-3: Position




Exp-3: Force

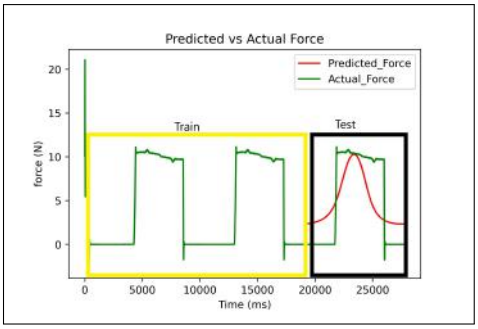
Figure 6.8: Experiments Results3: Force Responses and Position Responses

#### 6.2.3.2. Simulations.

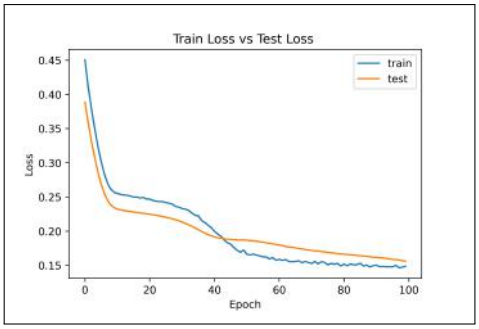
For the implementation of any neural networks, the most important thing is the data sets. These data sets were obtained in the previously where five different environments with different velocity references were recorded. The data recorded was split in two parts Train and Test. 80% of the data sets were used for training where as 20% was used for testing purpose. Jupyter notebook was used for creating and running the LSTM algorithm.

To use the data for LSTM algorithm it was necessary to convert the data to 3 dimension, so the data was preprocessed for this purpose. Once the data was preprocessed then it was passed through LSTM algorithm. Numerous Layer Structures with different set of trainable parameters were used of which some are illustrated in table 6.2. Due to the complexity of teleoperation system, the predicted response coming from the LSTM was not promising for the real-time situation. Algorithm was tried on different epochs but for the generalisation it was standardized to 100.

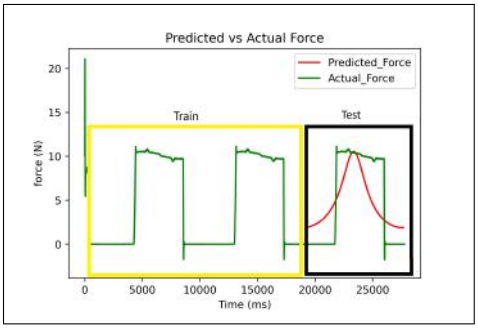
Looking at the force reconstruction results of the algorithm, one can observe LSTM algorithms show very slow convergence to the required results, this is basically due to the trainable parameters in the algorithm. Meanwhile, higher number of trainable parameters show much better transient response and rapidly converge to the actual values of the result.



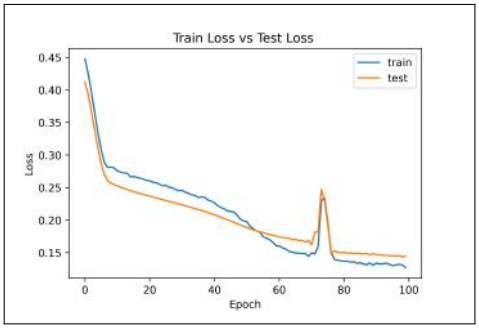

Model-1: Force



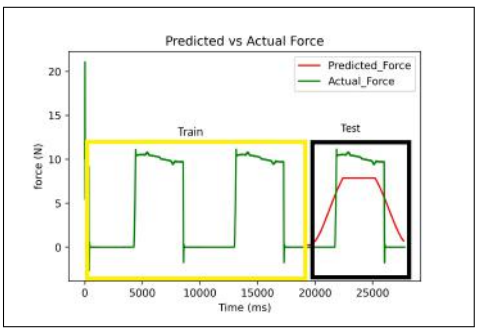

Model-1: Loss



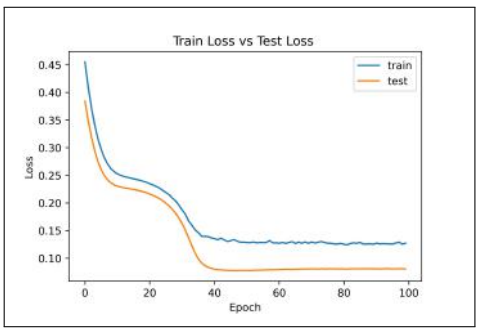

Model-2: Force



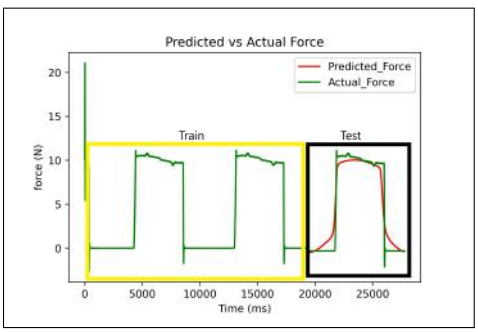

Model-2: Loss



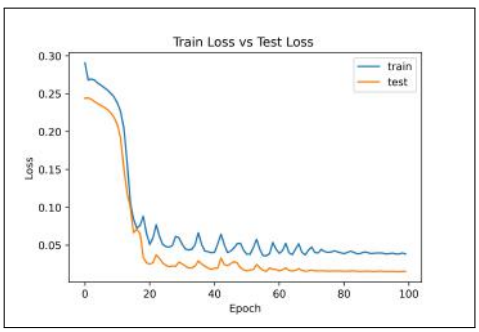

Model-3: Force




Model-3: Loss




Model-4: Force




Model-4: Loss

Figure 6.9: Simulation Results

| Model | Number of Trainable Parameters |
|---|---|
| 1 | 49 |
| 2 | 65 |
| 3 | 117 |
| 4 | 1401 |
| 5 | 5165 |
| 6 | 19517 |

Table 6.2: Simulations Performed On Different Models

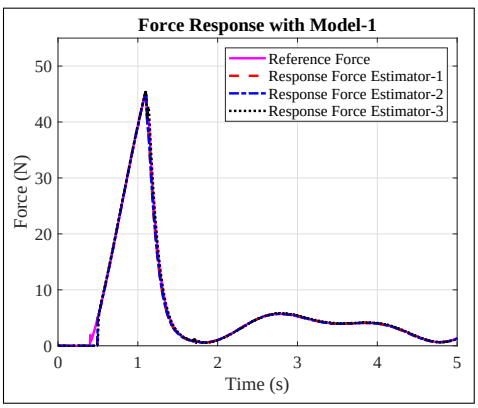
On the other hand, inspecting the loss between train and test one can observe that the convergence in the model with smaller parameters is slow compared to the convergence with higher number of parameters.

In summary, it was concluded that for trainable parameters around 100 hundred which is suitable for real-time implementation, the algorithm was not able to predict the force response shown in figure 6.9. Whereas for trainable parameter values above 1000 the algorithm was able to predict nicely which is not suitable for real-time implementation.

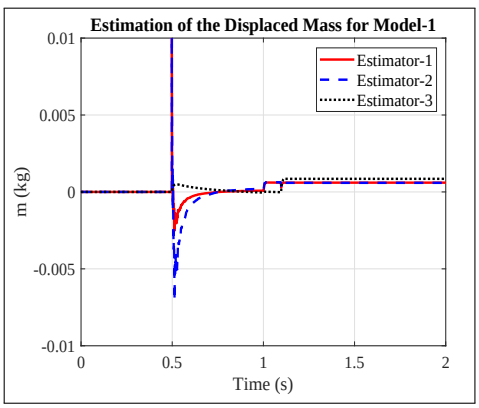
### 6.2.4. RLS Results

#### 6.2.4.1. Simulations.

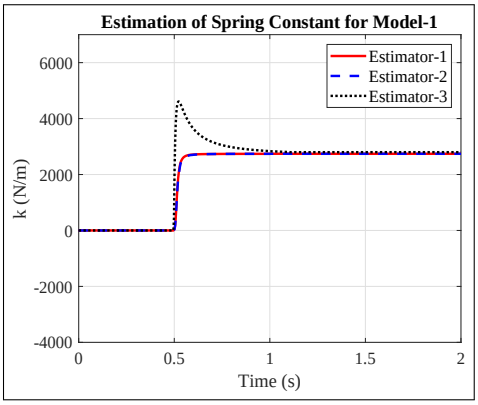
The environment force models in (5.7) – (5.9) are tested through a series of simulations with the estimator algorithms presented in the previous section. The simulations are made in the Matlab Simulink environment with the recorded position, velocity, acceleration and force data from the real experimented data recorded in the previous section. In the simulations, the same sampling rate used for the acquisition of the experimental data is used. The results of the simulations for the estimation algorithms discussed above are illustrated in figure 6.10 , 6.11 & 6.12.



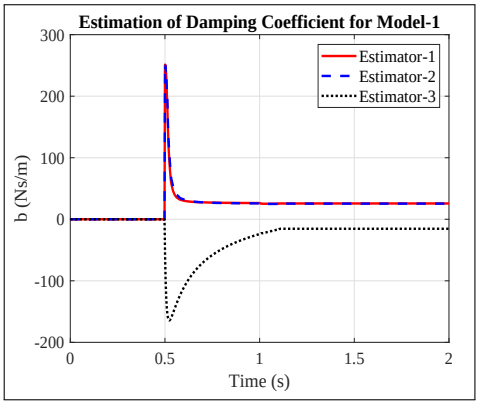

Model-1: Force




Model-1: Mass

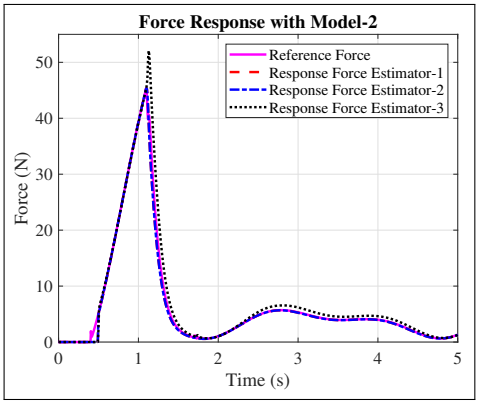



Model-1: Stiffness

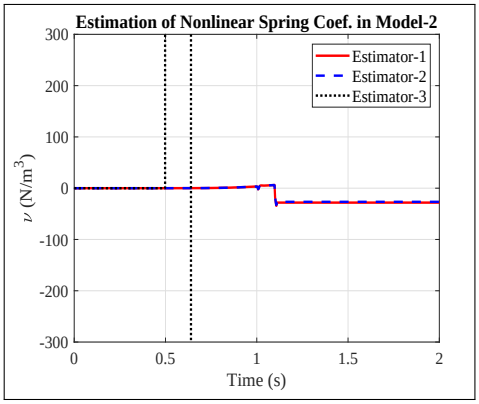



Model-1: Damping

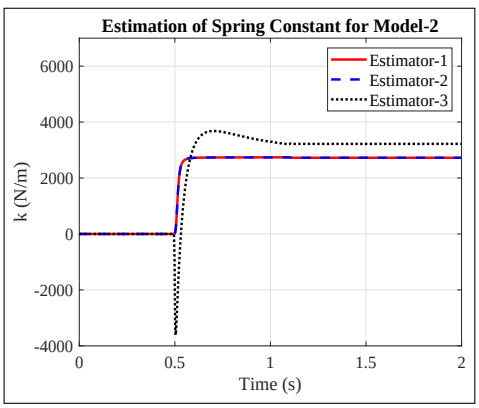
Figure 6.10: Simulation Results 1: Predicted Force Responses and the Estimated Parameters



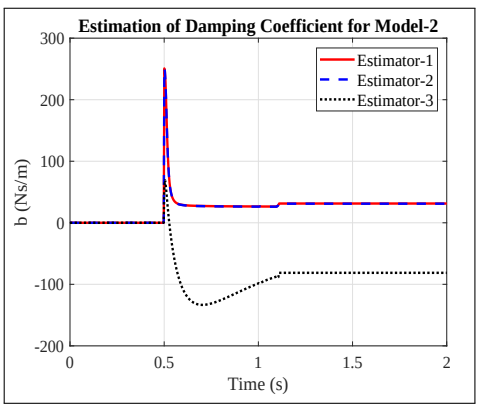

Model-2: Force




Model-2: Non-linear Spring Constant

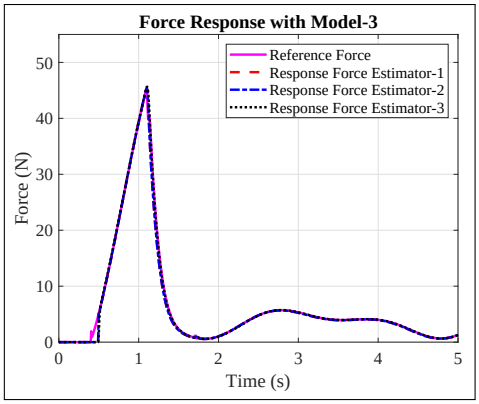



Model-2: Stiffness

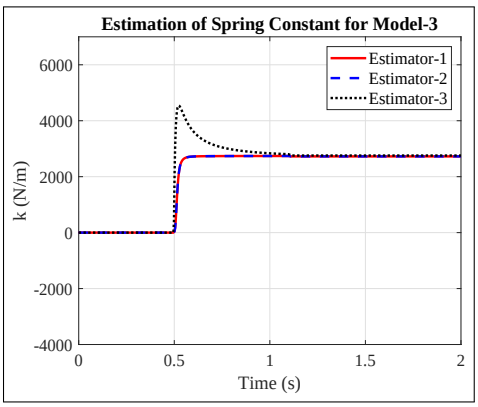



Model-2: Damping

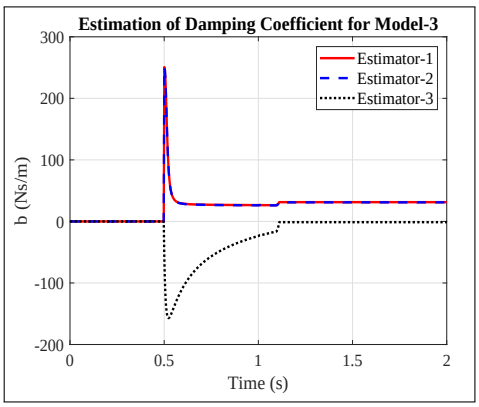
Figure 6.11: Simulation Results 2: Predicted Force Responses and the Estimated Parameters




Model-3: Force




Model-3: Stiffness




Model-3: Damping

Figure 6.12: Simulation Results 3: Predicted Force Responses and the Estimated Parameters

When examining the simulation results for force reconstruction, it is clear that the single value RLS and averaged value RLS estimation algorithms provide relatively comparable results, while both outperform the sliding window RLS algorithm. This is basically due to the low speed convergence of the model parameters in sliding window RLS. Meanwhile, the single value RLS and the averaged value RLS have a considerably superior transient response and quickly converge to the parameter values.

On the other hand, inspecting the responses between different force models one can observe that the convergence to the actual value of $\delta m$ in the model including the mass term is relatively slow. However, the linear impedance models with and without $\delta m$ term do not show salient differences in the reconstructed force. In other words, inclusion of the mass term does not contribute much on the force reconstruction results. Another important observation is the estimated non-linear spring coefficient shown in the results. This parameter does not show proper convergence to a positive value, which eventually provokes the reliability of the hardening spring force model.

In summary, simulations highlight that the single value RLS and averaged value RLS algorithms give good results with both of the linear impedance models. Taking into consideration the computational advantage of simplified impedance model (i.e. without the mass term) and the additional delay introduced in averaged RLS model, the single value RLS algorithm and the simple linear spring damper force model are selected for the experimental validation on a real-time platform.

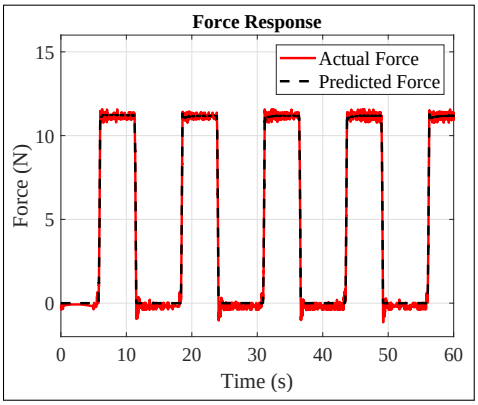
To accurately estimate the environment force, recognizing the position of the environment with respect to the actuator is necessary. Hence, in order to determine whether there is contact with an environment, the force response of the RFOB is thresholded as also suggested by [47]. This is required to prevent running the estimation algorithm mistakenly considering the free motion friction force as the contact force. The position of the motor at the time when the force response first exceeds the threshold value is taken as the environment contact position and the estimation algorithm is allowed to run and update only when the force is above that threshold value.

#### 6.2.4.2. Experiments.

In order to best illustrate the performance of the estimation algorithm three different experiments are made. In the first experiment, a relatively stiff rubber is preferred as the environment and the linear motor is controlled with saturating force reference in a repetitive motion. In other words, the saturation force value (i.e. 11.64 N) is used as the input reference of the force controlled system and this reference is given to the system repeatedly. The reason behind such a force reference is to observe the performance of the selected force model and the estimator when the force response changes suddenly, like a step function. The force tracking results and the estimated parameters from this experiment are illustrated in Fig. 6.16 (A) and (B), respectively.

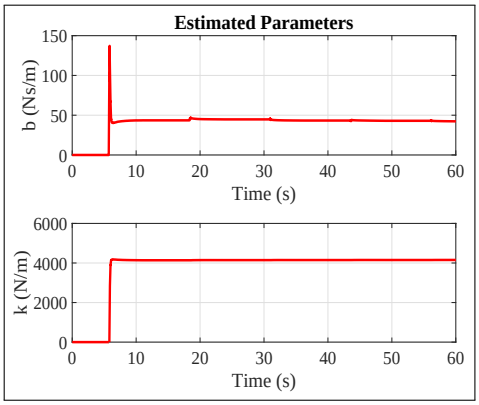
In the second experiment a relatively soft sponge is used as the environment and the linear motor is controlled with sinusoidal position reference such that the end effector of the actuator can have both free motion and contact motion periodically. The aim in this experiment is to see the behavior of the estimator for the cases where transition from free to contact motion occurs in a controlled manner. The results from this experiment are shown in Fig. 6.17 (C) and (D) for the same set of variables shown in the previous experiment results.

Finally, in the last experiment a harder rubber material is used as the environment and a randomly generated position reference is applied on the position controlled actuator. This last experiment is made in order to mimic a random motion response of the selected model and the estimator. Likewise, the results of this experiment are depicted in Fig. 6.18 (E) and (F). As also obvious from the experiments, the selected force model and the single value RLS algorithm give promising results for the estimation and reconstruction of the environment forces. The successful reconstruction of the forces in Fig. 6.18 also highlights the potential application possibility of this model and estimator for the local force feedback formulations of bilateral teleoperation systems.




Exp.1 Force

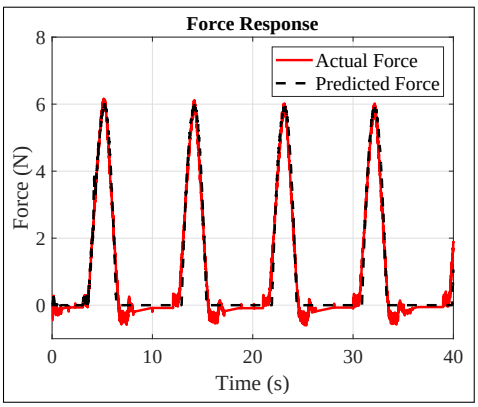
(A)




Exp.1 Parameters

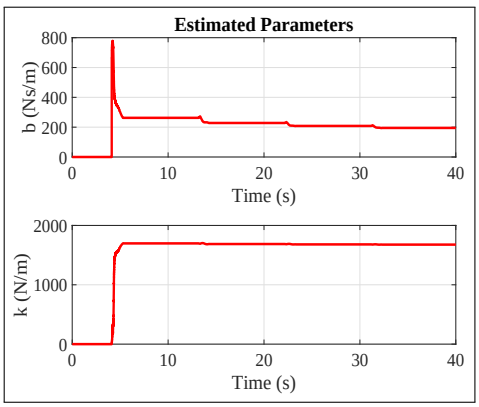
(B)

Figure 6.13: Experimental Results 1: Force Reference vs. Response and the Estimated Parameters




Exp.2 Force

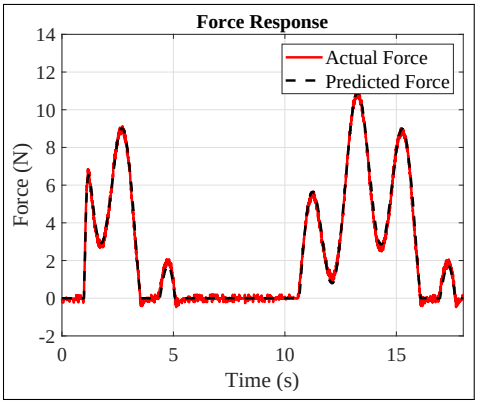
(C)




Exp.2 Parameters

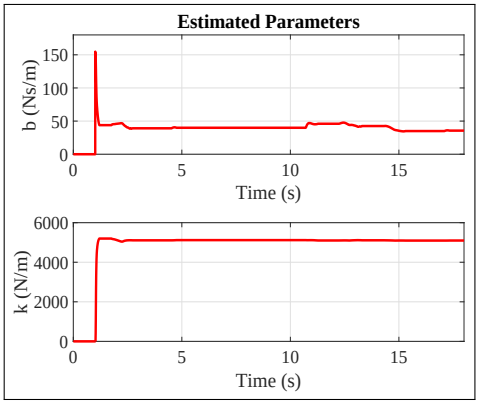
(D)

Figure 6.14: Experimental Results 2: Force Reference vs. Response and the Estimated Parameters




Exp.3 Force

(E)




Exp.3 Parameters

(F)

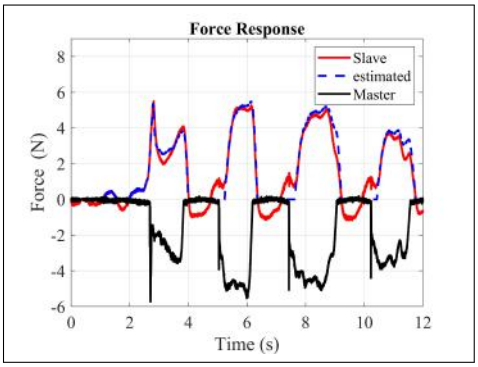
Figure 6.15: Experimental Results 3: Force Reference vs. Response and the Estimated Parameters

### 6.2.5. Motion Control System with Delay and Local Force Feedback Results

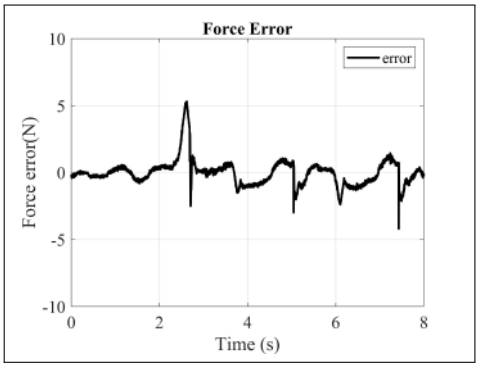
In order to best illustrate the performance of the estimation algorithm three different experiments are made. In the first experiment, a relatively soft sponge is preferred as the environment and the linear motor is controlled with arbitrary human force reference in a repetitive motion. In other words, this reference is given to the system repeatedly. The reason behind such a force reference is to observe the performance of the selected force model and the estimator when the force response changes suddenly. Hence the reconstruction of the local force feedback from the estimated parameters can be seen in figure

In the second experiment a relatively medium sponge is used as the environment and the linear motor is controlled with arbitrary human reference such that the end effector of the actuator can have both free motion and contact motion periodically. The aim in this experiment is to see the behaviour of the estimator for the cases where transition from free to contact motion occurs in a controlled manner. Then the response of reconstruction of the local force feedback from the estimated parameters is observed.

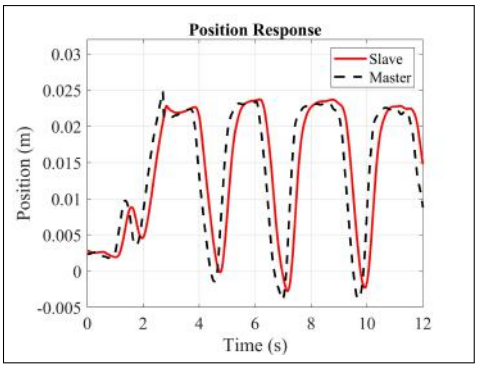
Finally, in the last experiment a harder sponge material is used as the environment and a randomly generated position reference is applied on the position controlled actuator. This last experiment is made in order to mimic a random motion response of the selected model and the estimator.



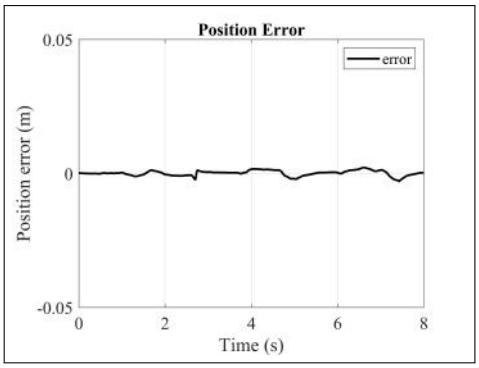

Exp1: Force Response



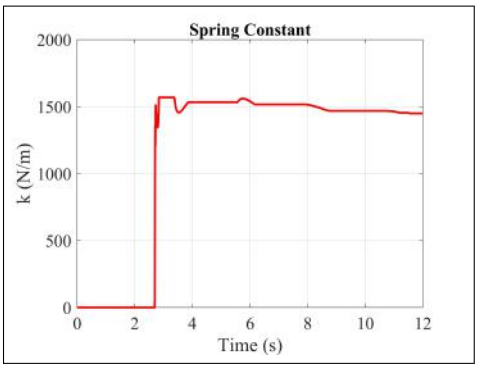

Exp1: Force Response Error



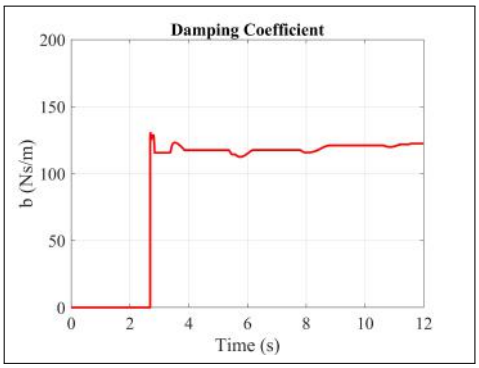

Exp1: Position Response




Exp1: Position Response Error

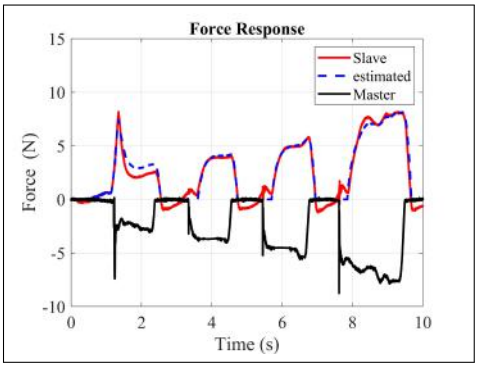



Exp1: Stiffness

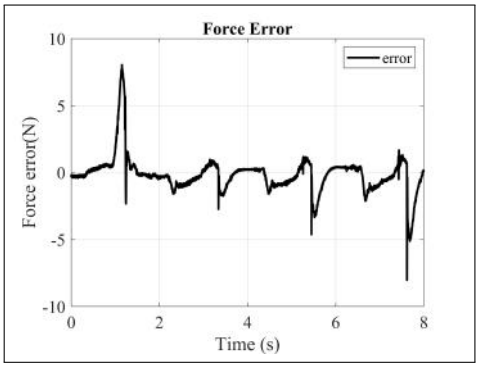



Exp1: Damping

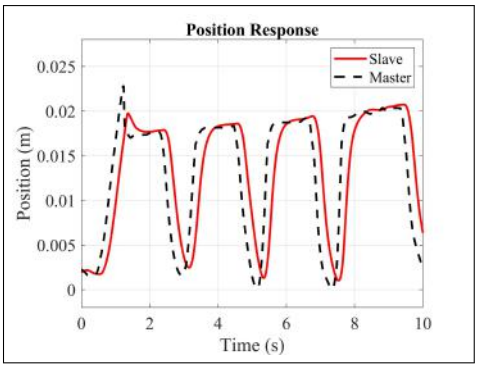
Figure 6.16: Experimented Results1: Force Reference vs Responses vs Local Force Feedback Response with error (first row), Position Response with error(second row) and Estimated Parameters (third row)



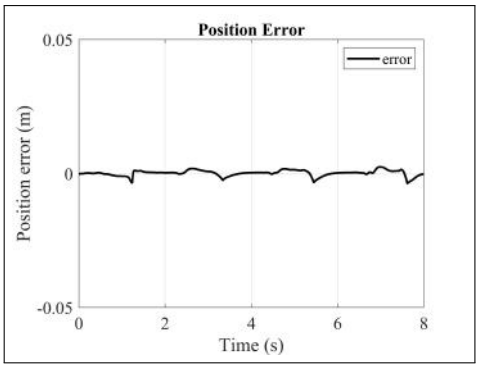

Exp2: Force Response



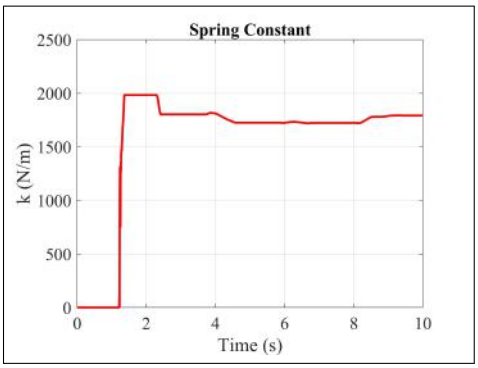

Exp2: Force Response Error



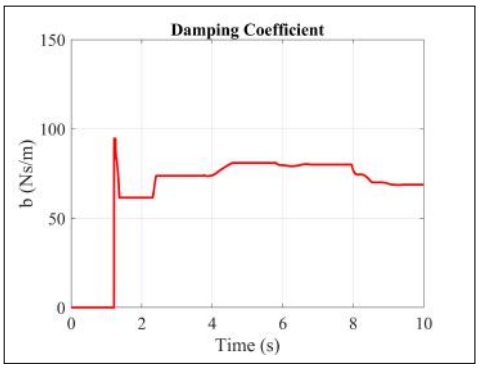

Exp2: Position Response




Exp2: Position Response Error

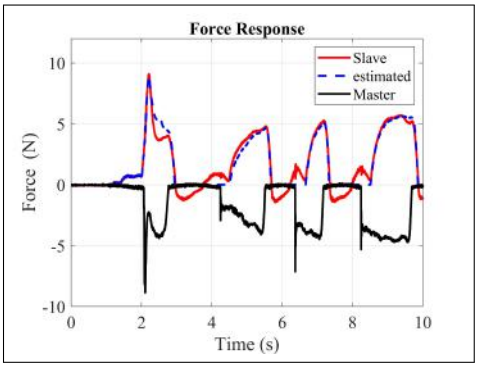



Exp2: Stiffness

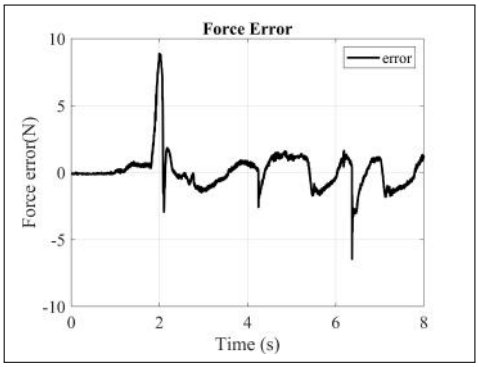



Exp2: Damping

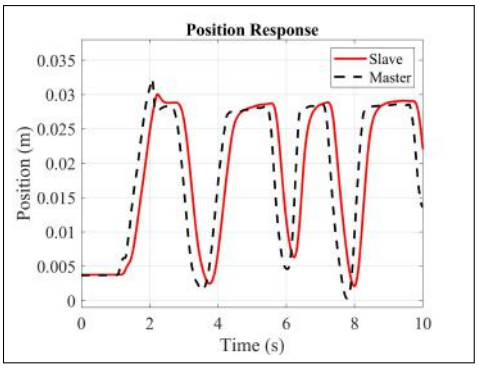
Figure 6.17: Experimented Results2: Force Reference vs Responses vs Local Force Feedback Response with error (first row), Position Response with error(second row) and Estimated Parameters (third row)



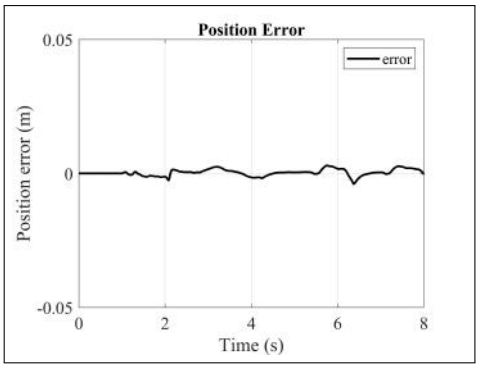

Exp3: Force Response



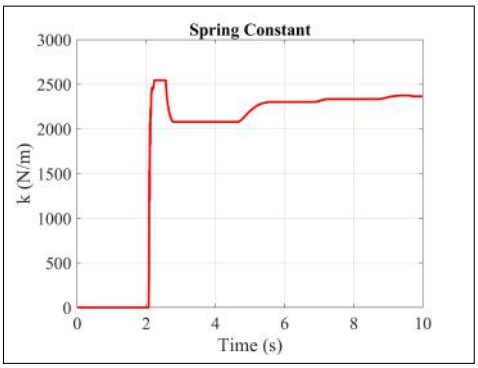

Exp3: Force Response Error



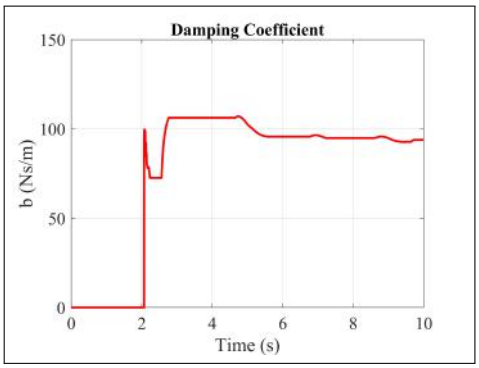

Exp3: Position Response




Exp3: Position Response Error




Exp3: Stiffness




Exp3: Damping

Figure 6.18: Experimented Results3: Force Reference vs Responses vs Local Force Feedback Response with error (first row), Position Response with error(second row) and Estimated Parameters (third row)

# 7. CONCLUSION AND FUTURE WORK

In this thesis a master-slave teleoperation system, bilateral control with and without delay is analysed and experimentally tested. A stable and transparent bilateral control can be achieved using the DOB, RFOB, CDOB (for bilateral control with delay) and acceleration control architecture. Two different methods were used for the estimation of remote environment parameters. The LSTM approach was not reliable because of the real-time implementation in the CPU.

In the second approach, the recursive least squares algorithm was used in the estimation of environment parameters to be used in bilateral teleoperation with time delay. The tested environment models include the classical linear impedance model, non-linear (i.e. hardening) and linear spring-damper models while the tested algorithms include the single value RLS, averaged parameter RLS and the sliding window RLS. The models and the algorithms were tested in the simulation environment making use of real experiment data. Among the force models investigated, the linear spring damper model is preferred due to its sufficiently accurate force estimation as well as its simplicity and low computational cost. The performance in the prediction of the contact forces is confirmed via the experimental results.

Once the environment parameters were estimated precisely they were fed back to the system for the reconstruction of a virtual environment on the master side, hence resulting in the Local Force Feedback controller of the bilateral teleoperation system.

For future work, improvements can be made for better estimation to get better performance especially for the hard environment. The same approach can be used to test tele-grasping applications. Furthermore, it can be used to test tele-surgical applications.

# REFERENCES


1. Passenberg, C., A. Peer and M. Buss, "A survey of environment-, operator-, and task-adapted controllers for teleoperation systems", *Mechatronics*, Vol. 20, 10 2010.

2. Lawrence, D., "Stability and transparency in bilateral teleoperation", IEEE.

3. Saglam, C. O., E. A. Baran, A. O. Nergiz and A. Sabanovic, "Model following control with discrete time SMC for time-delayed bilateral control systems", IEEE, 4 2011.

4. Goertz, R. C., "Master-Slave Manipulator", , 3 1949.

5. Marin, R., "History of Teleoperators Current Needs and Activities Control Remoto de Dispositivos Some Necessary Definitions Time Delay in Master-Slave Teleoperation History of Telerobotics", .

6. Ferrell, W. R., "Remote manipulation with transmission delay", *IEEE Transactions on Human Factors in Electronics*, Vol. HFE-6, 9 1965.

7. Ferrell, W. R. and T. B. Sheridan, "Supervisory control of remote manipulation", *IEEE Spectrum*, Vol. 4, 10 1967.

8. Miyazaki, F., S. Matsubayashi, T. Yoshimi and S. Arimoto, "A new control methodology toward advanced teleoperation of master-slave robot systems", Institute of Electrical and Electronics Engineers.

9. Hannaford, B. and P. Fiorini, "A DETAILED MODEL OF BI-LATERAL TELEOPERATION", .

10. Raju, G., G. Verghese and T. Sheridan, "Design issues in 2-port network models of bilateral remote manipulation", IEEE Comput. Soc. Press.

11. Anderson, R. and M. Spong, "Bilateral control of teleoperators with time delay", *IEEE Transactions on Automatic Control*, Vol. 34, 5 1989.

12. Niemeyer, G. and J.-J. Slotine, "Stable adaptive teleoperation", *IEEE Journal of Oceanic Engineering*, Vol. 16, 1 1991.

13. Niemeyer, G., "Using wave variables in time delayed force reflecting teleoperation", , 1996, `http://hdl.handle.net/1721.1/10622`.

14. Niemeyer, G. and J.-J. Slotine, "Towards force-reflecting teleoperation over the Internet", IEEE.

15. Kawashima, K., K. Tadano, G. Sankaranarayanan and B. Hannaford, "Bilateral teleoperation with time delay using modified wave variables", IEEE, 9 2008.

16. Yokokohji, Y., T. Imaida and T. Yoshikawa, "Bilateral teleoperation under time-varying communication delay", IEEE.

17. Munir, S. and W. Book, "Internet-based teleoperation using wave variables with prediction", *IEEE/ASME Transactions on Mechatronics*, Vol. 7, 6 2002.

18. Niemeyer, G. and J.-J. E. Slotine, "Telemanipulation with Time Delays", *The International Journal of Robotics Research*, Vol. 23, 9 2004.

19. Chopra, N., M. Spong, R. Ortega and N. Barabanov, "On tracking performance in bilateral teleoperation", *IEEE Transactions on Robotics*, Vol. 22, 8 2006.

20. Hashtrudi-Zaad, K. and S. Salcudean, "Transparency in time-delayed systems and the effect of local force feedback for transparent teleoperation", *IEEE Transactions on Robotics and Automation*, Vol. 18, 2002.

21. Massimiliano, V., "Performance Improvement of Smith Predictor through Automatic Computation of Dead Time", .

22. Hashtrudi-Zaad, K. and S. Salcudean, "Adaptive transparent impedance reflecting teleoperation", IEEE.

23. Colgate, J., "Robust impedance shaping telemanipulation", *IEEE Transactions on Robotics and Automation*, Vol. 9, 1993.

24. Slama, T., D. Aubry, R. Oboe and F. Kratz, "Robust bilateral generalized predictive control for teleoperation systems", IEEE, 6 2007.

25. Leung, G., B. Francis and J. Apkarian, "Bilateral controller for teleoperators with time delay via μ-synthesis", *IEEE Transactions on Robotics and Automation*, Vol. 11, 1995.

26. Natori, K., "Time delay compensation for motion control systems", , 2008.

27. Natori, K., R. Oboe and K. Ohnishi, "Analysis and Design of Time Delayed Control Systems with Communication Disturbance Observer", IEEE, 6 2007.

28. Natori, K., T. Tsuji and K. Ohnishi, "Time delay compensation by communication disturbance observer in bilateral teleoperation systems", IEEE.

29. Sabanovic, A., K. Ohnishi, D. Yashiro and N. Sabanovic, "Motion control systems with network delay", IEEE, 11 2009.

30. Rodriguez-Seda, E. J., D. Lee and M. W. Spong, "An experimental comparison study for bilateral internet-based teleoperation", IEEE, 10 2006.

31. Sankaranarayanan, G. and B. Hannaford, "Experimental comparison of internet haptic collaboration with time-delay compensation techniques", IEEE, 5 2008.

32. Liang, J., J. Mahler, M. Laskey, P. Li and K. Goldberg, "Using dVRK teleoperation to facilitate deep learning of automation tasks for an industrial robot", IEEE, 8 2017.

33. Kebria, P. M., A. Khosravi, S. Nahavandi, Z. Najdovski and S. J. Hilton, "Neural Network Adaptive Control of Teleoperation Systems with Uncertainties and Time-Varying Delay", IEEE, 8 2018.

34. Ochi, H., W. Wan, Y. Yang, N. Yamanobe, J. Pan and K. Harada, "Deep Learning Scooping Motion using Bilateral Teleoperations", , 10 2018.

35. Ogata, K., *System Dynamics*, Prentice Hall, 2003.

36. Ohnishi, K., M. Shibata and T. Murakami, "Motion control for advanced mechatronics", *IEEE/ASME Transactions on Mechatronics*, Vol. 1, 3 1996.

37. Murakami, T., F. Yu and K. Ohnishi, "Torque sensorless control in multidegree-of-freedom manipulator", *IEEE Transactions on Industrial Electronics*, Vol. 40, 4 1993.

38. Murakami, T., F. Yu and K. Ohnishi, "Torque sensorless control in multidegree-of-freedom manipulator", *IEEE Transactions on Industrial Electronics*, Vol. 40, 4 1993.

39. Matsumoto, Y., S. Katsura and K. Ohnishi, "An analysis and design of bilateral control based on disturbance observer", IEEE.

40. Šabanović, A., K. Ohnishi, D. Yashiro, M. Acer and N. [U+FFFD] Behlilović, "CONTROL SYSTEMS WITH NETWORK DELAY", .

41. Hochreiter, S. and J. Schmidhuber, "Long Short-term Memory", *Neural computation*, Vol. 9, pp. 1735–1780, 10 1997.

42. Gabor, D., "Communication Theory and Cybernetics", *Transactions of the IRE Professional Group on Circuit Theory*, Vol. CT-1, 12 1954.

43. Widrow, B. and M. E. Hoff, ""Adaptive Switching Circuits"", *"1960 IRE*

*WESCON Convention Record"*, pp. 96–104, 1960.

44. Rosenblatt, F., "The perceptron: A probabilistic model for information storage and organization in the brain.", *Psychological Review*, Vol. 65, 1958.

45. Li, J., X. Wei and H. Dai, "Design and implementation of RLS identification algorithm based on FPGA", IEEE, 8 2009.

46. Umeda, K., T. Sakuma, K. Tsuda, S. Sakaino and T. Tsuji, "Reaction Force Estimation of Electro-hydrostatic Actuator Using Reaction Force Observer", *IEEJ Journal of Industry Applications*, Vol. 7, 5 2018.

47. Mitra, P. and G. Niemeyer, "Model-mediated Telemanipulation", *The International Journal of Robotics Research*, Vol. 27, 2 2008.