The Development of System Identification Approaches for Complex Haptic Devices and Modelling Virtual Effects Using Fuzzy Logic

by

Sze-Man Samantha Tam

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2005

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Haptic applications often employ devices with many degrees of freedom in order to allow the user to have natural movement during human-machine interaction. From the development point of view, the complexity in mechanical dynamics imposes a lot of challenges in modelling the behaviour of the device. Traditional system identification methods for nonlinear systems are often computationally expensive. Moreover, current research on using neural network approaches disconnect the physical device dynamics with the identification process. This thesis proposes a different approach to system identification of complex haptic devices when analytical models are formulated. It organizes the unknowns to be identified based on the governing dynamic equations of the device and reduces the cost of computation. All the experimental work is done with the Freedom 6S, a haptic device with input and feedback in positions and velocities for all 6 degrees of freedom .

Once a symbolic model is developed, a subset of the overall dynamic equations describing selected joint(s) of the haptic robot can be obtained. The advantage of being able to describe the selected joint(s) is that when other non-selected joints are physically fixed or locked up, it mathematically simplifies the subset dynamic equation. Hence, a reduced set of unknowns (e.g. mass, centroid location, inertia, friction, etc) resulting from the simplified subset equation describes the dynamic of the selected joint(s) at a given mechanical orientation of the robot. By studying the subset equations describing the joints, a locking sequence of joints can be determined to minimize the number of unknowns to be determined at a time. All the unknowns of the system can be systematically determined by locking selected joint(s) of the device following this locking sequence. Two system identification methods are proposed: Method of Isolated Joint and Method of Coupling Joints. Simulation results confirm that the latter approach is able to successfully identify the system unknowns of Freedom 6S. Both open-loop experimental tests and close-loop verification comparison between the measured and simulated results are presented.

Once the haptic device is modelled, fuzzy logic is used to address chattering phenomenon common to strong virtual effects. In this work, a virtual wall is used to demonstrate this approach. The fuzzy controller design is discussed and experimental comparison between the performance of using a proportional-derivative gain controller and the designed fuzzy controller is presented. The fuzzy controller is able to outperform the traditional controller, eliminating the need for hardware upgrades for improved haptic performance. Summary of results and conclusions are included along with suggested future work to be done.

Acknowledgements

It was a pleasure to work with my supervisor, Professor Eric Kubica, who has always been supportive in giving me the liberty to explore my research interests. His advice and encouragement are the key factors to making research experience enjoyable.

I would like to thank my mentor, Professor David Wang, for his valuable guidance and support throughout my academic career. His help has made my research possible.

Many thanks to everyone at Handshake VR Inc. On top of being the industrial sponsor for my research, the technical team members provided great mentorship in sharing their expertise in haptics and robotics applications. Their help is very much appreciated.

Special thanks go to Edmon Chan, for his unparallel appetite for helping his fellow colleague in need. Also, I would like to thank NSERC for funding this research project.

My family has always been supportive on my decision of post-graduate studies. I cannot thank my parents enough for their understanding and encouragement.

Last but not least, I like to thank my partner, Joseph Tam, for his continuous love and support in every step throughout this journey.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

With an increasing number of applications on haptic robotics emerging in the fields of medicine, aviation, defense and education [1], it pushes the need for advancing the research on these devices. Whether the research is related to designing control schemes or implementing teleoperation, a good physical dynamic model of the haptic device is essential. This thesis provides some new modelling and system identification methods that would help to achieve that goal.

In addition to the modelling of haptic devices, the thesis also examines the specific haptic effects of virtual wall contact. In the world of virtual reality, simulating the contact of a solid wall is difficult due to the presence of chattering effects. These effects ultimately impair the user's sensation of a hard surface when using a haptic device. This thesis explores the possibility of improving the modelling of a virtual wall to provide a realistic perception for the user using a fuzzy logic approach.

1.1 What is Haptics

The word "haptics", originated from the Greek word "haptikos", means "of or relating to or proceeding from the sense of touch" [2]. Physiologically, the sense of touch is established at the primary sensory cortex of the brain by processing information received from mechanoreceptors populated throughout the skin [3]. Through this information processing, we are able to feel pressure and are able to distinguish different texture and movement.

We heavily depend on the sense of touch to efficiently complete our everyday tasks such as typing keyboards, playing instruments, steering vehicles, writing, etc. Without the sense of touch, it increases the difficulty in completing many tasks tremendously. It is easy to imagine how challenging it is to adjust the ski goggle straps when the hands are frozen in the cold.

With advancement in technology, the concept of haptics can be added to many digital applications to improve efficiency of task-handling. This is accomplished by using haptic devices, which are capable of giving force feedback, to interface with a computer. Through instructions from software, the computer is able to send signals to control the force exerted by the haptic device. The user can then use the device to feel virtual objects from the commanded force.

1.2 Motivation

The two topics discussed in this thesis are motivated by existing applications. The first topic is driven by the challenges involved with doing system identification on robots with a high number of degrees-of-freedom (DOF) coupled with nonlinear friction characteristics. In some cases, haptic devices are simple enough that a basic transfer function can be used based on linear approximation [4]. However, the simplification might not be accurate enough to capture all the dynamic behaviours of the system outside of a certain operating point. The most general approach for modelling haptic devices is done by designers who have knowledge of the physical parameters [5], [6] and these data are used to form the general dynamic equations of the system. Our modelling approach addresses complex systems that cannot be modelled by linear approximation and it is able to model dynamics with unknown physical parameters.

Once a model is derived, the difficulty in simulating a realistic solid wall contact using the virtual wall implementation is investigated. Vibration phenomenon often results as the user tries to interact with the virtual wall. This thesis takes a different approach than the traditional methods in the modelling of virtual wall to improve the existing model without the need for any hardware upgrades by employing fuzzy logic control strategies.

1.3 Problem Description

In the first part of the thesis, the dynamic modelling of a 6 DOF haptic device, the Freedom 6S[1], is studied. The front and back views of the Freedom 6S are displayed in Figures 1.1 and 1.2. It has three direct-drive motors controlling the translational motion of the base links. The roll-pitch-yaw motion is provided by the wrist joints and the torque transmissions are done by tendons and pulleys actuated by three additional motors. It is difficult to disassemble the device and no precise physical parameters (e.g. mass, inertia, centroid location) are available. In addition to these unknowns, the friction effect in each joint has to be accounted for. As a result, the number of unknown system parameters is further increased.

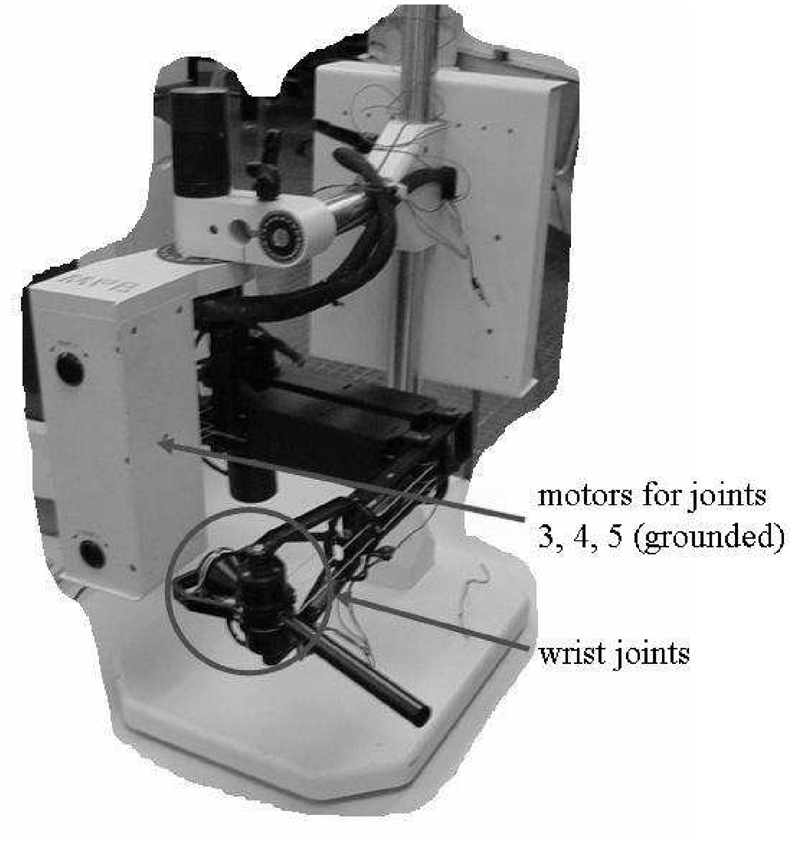


Figure 1.1: Front view of Freedom 6S.

[1] Manufactured by MPB Technologies Inc., Montreal, Quebec. www.mpb-technologies.ca

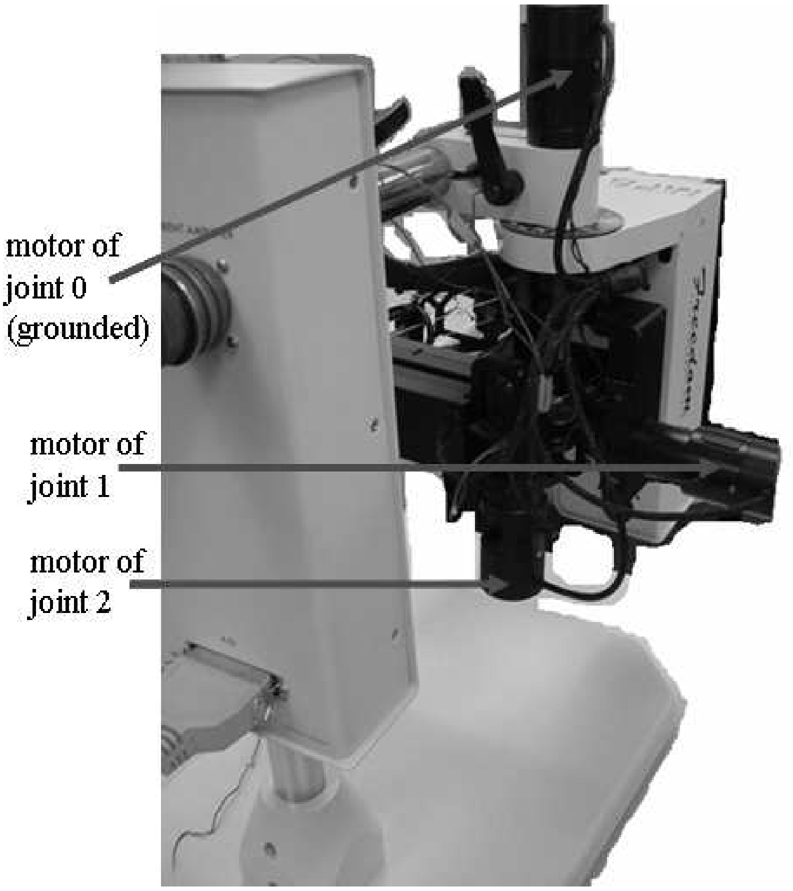


Figure 1.2: Back view of Freedom 6S.

With so many unknown parameters, it is advantageous to organize and to have a systematic methodology to identify parameters in an iterative fashion. For example, it may be possible to reduce the number of parameters by decoupling the wrist joints from the base joints. However, this approach still presents many unknowns to be solved at each step. This thesis proposes two non-conventional system identification techniques to isolate the unknown parameters joint(s) in an iterative fashion. Mathematically, only the dynamic equations of the joints of interest are required for system identification. Physically, joint isolation can be done by locking the non-relevant joints in place. Through this arrangement, only the unknown parameters of the dynamics of specified joint(s) are affecting the system response. By determining the parameters that affect each locked joint's configuration, a sequence of which joints to lock can be established to solve for the unknown parameters iteratively. Hence, the number of parameters required for identification at any iteration is reduced dramatically.

The second part of the thesis investigates the potential of minimizing chattering effects in virtual wall applications using fuzzy logic. The effectiveness of using fuzzy logic in modelling a virtual wall as compared to using of a PD (Proportional and Derivative) controller is evaluated. As previously mentioned, the vibration issues associated with the traditional spring-and-damper model imposes an incorrect perception of a hard surface for the user. Without requiring any change of the existing data acquisition hardware, a fuzzy logic controller can be designed to minimize the vibration. Using the Freedom 6S for experimentation, the measured results of using both the traditional PD method and the fuzzy logic approach are studied and compared.

1.4 Thesis Outline

Chapter 2 provides some background on haptic robotics and the existing modelling techniques and challenges associated with these devices. It also includes some information on the optimization method used in the system identification routine. In addition to the background on modelling, Chapter 2 also reviews some existing work and challenges in working with virtual walls using haptic devices. The background on fuzzy control is reviewed at the end as it is the method proposed for the virtual wall application to be discussed in the latter part of the thesis.

Chapter 3 covers the formulation of a dynamic model based on kinematic relationships. It also demonstrates how to incorporate other observed effects, such as friction, into this dynamic model. In Chapter 4, thorough descriptions of the two proposed system identification techniques, the method of isolated joint and the method of coupling joints, are outlined. Comparisons between the measured and simulation data for an open-loop system are also presented in Chapter 4. As a verification, results from comparing the measured and simulation results in a close-loop system are also included.

Chapter 5 first provides the descriptions of test setups as well as the details of the fuzzy controller design. It is then followed by a comparison of measured data obtained from using a PD controller and the fuzzy controller. The chapter ends with some additional findings on the relationship between the user approach speed and the perception of wall boundary.

Chapter 6 highlights the contributions established by this thesis. Recommendations on future work on applying the thesis results are also included as part of the closing remarks.

Chapter 2

Background and Literature Review

This chapter gives an overview of the haptic device, Freedom 6S, used in the experimental work. It also gives a summary of the existing system identification methods and the background of the optimization routine used for identification in this thesis. Later on in the chapter, work that has been done in simulating solid wall contact with haptic devices, along with some basic background in fuzzy logic will be outlined.

2.1 Research in Haptics

The early research in the field of haptics was done by Kennedy [7] who described a range of haptics (e.g. touch, sensation, etc). The idea of sense of touch was introduced to the world of technology starting in the late 1980's in hope of improving an operator's ability to complete a task more effectively via an haptic interface. Numerous studies established that the addition of haptics effects is useful for many applications involving interaction between human operators and manipulators [8], [9], [10], [11]. For example, tactile sensing is particularly advantageous for robotic surgical procedure involving cutting soft tissues. A surgeon can better plan cutting strategies with the sense of touch because it allows for the detection of features of the soft tissues to be cut [12].

Many researches relating to the designs of haptic devices/interface have also evolved. Some devices, such as the Utah-MIT hand [13], mimics closely to the human hand, fingers and thumb. Others devices are designed to improve the efficiency of force-feedback control methods. The parallel manipulators designed by Merlet [14] have good qualities which facilitate force-feedback control. These include good position accuracy and good passive compliance behaviour. Another example of haptic devices designed to assist in force-feedback operation is the five-bar-linkage force reflecting interface developed by Ching and Wang [15]. The manipulator is designed to be gravity balanced such that the motor power can be devoted completely to force-reflecting operation. There are also cable-driven devices, such as the SPIDAR designed by Hirata and Sato [16], which allows and stops force transmission by braking the cable connected to a finger cap worn by the user. The most dominating commercialized haptic devices are the PHANTOM devices which have multiple DOFs and a wide range of force capabilities [17]. These devices are especially popular with academic researches [18], [19], [20], [21].

Many other researches related to haptics have been published. A lot of work has been done in studying the different control strategies used for force-reflecting operations [22], [23], [24], [25], [26]. One of the first force control methods, proposed by Hogan [27], is the concept of impedance control. Impedance control is used to govern the relationship between velocity and force [28]. This control strategy is commonly used in many haptics researches [29], [30], [31]. Raibert and Craig [32] proposed a hybrid position-force control scheme in which a position control law is designed along force constrained directions and vice versa. Another widely-adopted force control scheme is hybrid impedance control proposed by Anderson and Spong [23]. It combines both impedance and hybrid position/force control into one strategy.

In addition to haptics research pertaining to control methodologies, issues associating with teleoperation are also being researched. Teleoperation is “the remote manual operation of equipment that is usually not within the direct eyesight of the operator, yet the operator requires and is provided with sensory information (sight, sound, accelerations, etc.) for effective manual control”. [2]. One of the known issues is instability caused by time delay. Wang et. al [33] have used proprietary time delay compensation solution to perform real-time remote handshake using haptics effects. Other issues related to teleoperation are synchronization and transparency which have impacts on the efficiency of the operations. In summary, researches ideas related to haptics received much attention and the trend is likely to continue as the sense of touch is being incorporated into more mainstream applications.

2.2 A Haptic Device: The Freedom 6S

Unlike many industrial robots, haptic robots have unique characteristics that make them suitable for force-reflecting operations. Their desirable characteristics include back-driveability, low inertia, minimal friction and little backlash [34]. In addition, haptic devices must be capable of providing a sufficient amount of force for the desired application.

The Freedom 6S, a force feedback hand controller with 6 DOFs, is designed to offer some characteristics of a desirable haptic device. Having 6 DOFs, the first three motors directly drive the base linkages and the last three motors drive the roll-pitch-yaw orientation of the wrist. There is minimal gearing on the motors to facilitate effective force transmission. The first and the last three motors are grounded. This means that they are located at fixed positions and do not move regardless of joint movements. In particular, the last three motors controlling the wrist joints are centralized in a fixed housing (see Figures 1.1 and 1.2).

To minimize the inertia of the system, the four motors are designed to be grounded to remove any reflected motor inertia. The other two motors and the rest of the structure is designed such that they are statically balanced [35]. In doing so, no holding torques are required to maintain the device in a static position. This allows almost all of the torque of the motors to be used for force-reflecting operations with some of the torque required to move the inertia components. This in turns provide a higher force transmission to the end effector.

The wrist joints are tendon-driven and the tendons allow for the motors to be grounded away from the wrist joints. These joints, in general, might pose some problems for modelling since the friction characteristics change dramatically when the tendons are constantly under load. Nonetheless, the Freedom 6S is a device specifically designed for haptic application and it is the robot of interest for the experimental studies used in this thesis.

2.3 Modelling and System Identification of Robots and Haptic Robots

Many different methods of modelling and system identification are employed for robots and haptic robots. Depending on what information is known and what accuracy is required, the appropriate technique is chosen. Existing identification methods used on robots in general are discussed first followed by the methods used to specifically identify haptic robots.

A general approach is to collect data of different frequency responses and perform a least squares algorithm on the data to determine the best-fitted transfer functions. Trautman and Wang [36] employed this method to successfully identify a single flexible link with a shoulder joint. Another technique often used to identify rigid robot dynamics is to take advantage of the linearity in parameters. In Spong and Vidyasagar [28], it is shown that the unknown parameters can be rearranged into coefficient terms from the nonlinear equations of motion:

$$D(q)\ddot{q} + C(q,\dot{q})\ddot{q} + \phi(q) = Y(q,\dot{q},\ddot{q})p = \mathbf{u} \tag{2.1}$$

where D is the inertia matrix, q is the joint position vector, $\ddot{q}$ is the joint acceleration vector, $\boldsymbol{C}$ is the Christoffel matrix, $\dot{q}$ is the joint velocity vector, ϕ is the gravitational effect matrix, Y is the matrix of known functions, p is the unknown coefficient vector.

By grouping the unknown coefficients in the p vector, least square algorithm can be applied directly. The shortcoming of this approach is that it is only feasible for systems with a limited number of unknown coefficients. Otherwise, it can be computationally expensive.

In many cases, linearizing the model may become the necessary option to decrease the computational time required, resulting in compromises in the level of accuracy. Another approach to system identification of dynamic systems is the use of neural networks. For systems which are highly nonlinear and do not allow the unknown parameters to be grouped into a linear coefficient vector, neural networks and fuzzy logic techniques are used as the alternative solutions. Narendra et al. [37] applied neural networks to perform system identification in time and frequency domains for nonlinear dynamic systems. Chu et al. [38] implemented a least squares estimation for both time variant and invariant systems using a Hopfield network.

Ahmed [39] successfully identified nonlinear dynamics systems by applying a rapid neural network. In terms of incorporating fuzzy logic to identify nonlinear systems, Efe et al. [40] employed an adaptive neuro fuzzy inferencing system to identify a 2 DOF direct drive SCARA robot. Gao and Joo [41] used a robust adaptive fuzzy neural controller to identify and control a two-link robot manipulator. Evidently, neural networks and fuzzy logics have achieved much success in modelling nonlinear systems. However, the associated computational cost is still quite high [42], [40] and a good result is not ensured.

More specifically, the system identification techniques used on haptic robots are of interest to the thesis. In some instances, system identification is not necessary. Madill [43], Cauche et al. [5], and Avizzano et al. [6] designed their own force-feedback devices and hence they had access to the physical parameters . In other cases, it is common to use linear approximation on simple haptic devices. Ando et al. [4] used model reference adaptive control on their 6 DOF haptic interface and modelled the plant transfer function using linear approximation. Bluethmann et al. [44] analyzed output response from known inputs using least-squares based estimation algorithm to determine the desired transfer function for the force and position control of an electrohydraulic manipulator. These approaches are limited to linear systems and for systems which are linearizeable.

In hope of modelling a nonlinear haptic robot with a lower computational cost, this thesis proposes a method to systematically identify the unknowns by dividing the dynamic model into subset equations in such a way that each subset can be tested physically. This simplification greatly reduces the computational time of the nonlinear optimization used to determine the unknowns and in the end, the combined results would determine all the unknowns required to describe the overall system.

2.4 Optimization

To perform system identification, the unknown parameters must be determined such that they can best fit the collected data. Nonlinear optimization is used to minimize the difference between the measured and the simulated values during the system identification process described in Chapter 4.

One optimization method is to use a direct search by choosing incremental points (i.e. values of unknown parameters) to be evaluated by the cost function. The combination of points yielding the lowest cost function value would be the optimized parameters. This strategy requires the knowledge of upper and lower bounds in addition to the pre-determined magnitude of the incremental values. However, due to the large number of unknowns to be identified for the Freedom 6S, it is extremely computationally expensive and time-consuming. The gradient search method is a popular optimization method in which it determines the next iterative point by evaluating the gradient of the cost function. Although the gradient method often offers fast convergent rate, the implementation requires the cost function to be continuously differentiable [45]. This requirement also applies to the Bolzano search method in which the derivative of the cost function is needed and it is evaluated at the midpoint of a convex interval [46]. To optimize a discrete cost function with a large number of parameters as described in this thesis, Sequential Quadratic Programming (SQP) is implemented via the Matlab's nonlinear optimization routine. The general principle is outlined below.

In general, nonlinear optimization problems are approached based on finding the solution to the Kuhn-Tucker (KT) equations:

$$\begin{aligned} \bigtriangledown f(x) + \textstyle\sum_{i=1}^{m} \lambda_i \cdot \bigtriangledown G_i(x) &= 0 \\ \lambda_i \cdot G_i(x) &= 0 \quad \text{where } i\text{=1,...,}m \\ \lambda_i &\geq 0 \quad \text{where } i\text{=}m_e\text{+1,...,}m \end{aligned}$$

where *f(x)* is the objective function to be minimized; λ is the Lagrange multiplier which allows for the gradients of the objective function and the active constraints to sum up to 0; *G(x)* is the constraints required for the solution; i is the constraint index; m is the number of constraints; m_e is the index from which the constraints are active. Active constraints set the boundaries in which the solution lies.

To solve the KT equations, SQP methods are used and these methods are applied by first breaking down the original problem into smaller Quadratic Programming (QP) problem during each iteration. A quadratic approximation of the Lagrangian function is used to develop a QP subproblem and it takes the form of:

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i \cdot G_i(x) \tag{2.2}$$

This form allows for the calculation of the Hessian matrix, which is required to set up the QP subproblem, and the determination of a search direction which in turns is used to formulate the next iteration of x. The implementation of SQP is explained in detail in [47]. In this thesis, the optimization is done using the “fmincon” function in Matlab [1].

2.5 Virtual Wall Contact

In haptic applications, modelling a virtual wall is a necessity in the virtual reality environment. A common issue is the presence of nonpassive behaviour from the virtual wall. Passivity can be interpreted as the inability to act as an energy source [48]. By nonpassive, it means that the virtual wall is capable of generating net energy when the user is interacting with it and this can lead to instability. The resulting instability is caused by a combination of factors such as the dynamics of the device and quantization in position sensing. If the resolution of the position sensor is low and/or if the sampling rate is not sufficiently fast, the position data is not accurate and that could lead to incorrect reaction force output by the virtual wall.

Different approaches have been taken to ensure the passivity of a virtual wall is preserved for the user. A method proposed by Madill et al. [49] is to estimate the position and velocity using a nonlinear observer to compensate for the quantization effect from the sensor. Another technique is to model the virtual wall system using the concept of “energy leaks”. Goldfarb and Wang [50] simulated the system as loosing energy by modelling the virtual spring with hysteresis effects. Colgate and Schenkel [51] used viscous friction as a method to dissipate energy generated by the wall by relating viscous friction and sampling rate to the stiffness of the virtual wall. Similarly, Kim and Ryu [52] generated algorithms that restrict the energy generation of the zero-order-hold by using physical damping. The passivity approach fabricates a system which is passive to the user. This thesis takes the approach of modelling the virtual wall as a nonlinear system governed by fuzzy logic based on reasoning through the sense of touch from the user. It is relatively simple to implement and the

[1] Matlab is a scientific computation program developed by The Math Works, Inc. See http://www.mathworks.com for details

background required for the fuzzy controller design is discussed in the next section.

2.6 Fuzzy Logic

In an attempt to minimize vibrations and chattering effects in virtual wall contact applications, a fuzzy logic approach is implemented and the details are described in Chapter 5. This section outlines the existing research on fuzzy controllers and gives a basic background of fuzzy logic required to understand the development of the controller discussed later.

2.6.1 Existing Researches on Fuzzy Controllers

Fuzzy logic is "...the logic underlying approximate, rather than exact, modes of reasoning" [53]. It is an area of active research and it has achieved much success in real world applications. Gao and Joo [41] used a generalized fuzzy neural network to do nonlinear identification and control on a two-link robot manipulator. Their adaptive fuzzy neural controller is of a self-organizing fuzzy neural structure with capabilities to do on-line adaptive learning of uncertainties in nonlinear systems. Zadeh [54] has also implemented a fuzzy logic controller on a nonlinear system. In his work, the fuzzy controller is designed to be optimized with multiple objectives. His work demonstrates the flexibility of fuzzy logic controller for nonlinear systems. Another capability of fuzzy controllers is that they make good universal approximators. Galichet and Foulloy [55] illustrated the methodology to build a fuzzy controller from a given linear controller. Their method is capable of ensuring specified points to belong to a pre-determined control surface by deriving the required rule-base and membership functions. One advantage of having the ability to map linear controllers to fuzzy controllers is that it allows for implementation of control strategies beyond the operating points of the linear controller. Ryu and Park [56] adopt this fact and implemented a hybrid controller resembling a proportional-derivative and proportional-integral controller to control a flexible finger. Stability issues associated with fuzzy controller have been studied by Wang et al. [57]. Their controller design strategy is to employ the concept of "parallel distributed compensation" in which a control rule is derived such that it compensates each rule of a fuzzy system.

The research involving fuzzy controllers remains active and the implementation of fuzzy logic on a diverse range of applications is continually proven to be feasible.

2.6.2 Fuzzy Control Systems

Some definitions commonly used in fuzzy logic are given below considering a fuzzy set A defined over the universe of discourse X with membership functions μ_A.

Definition: ***Support*** The support of a fuzzy set A is the crisp set of all $x \in X$ such that $\mu_A(x) \geq 0$.

Definition: ***Universe of Discourse*** The range of all possible values for an input to a fuzzy system [58].

Definition: ***Membership Function*** A curve that defines how each point in the input space is mapped to a membership value in the interval [0,1]. [59].

Definition: ***Fuzzy Set*** Any set that allows its members to have different grades of membership (membership function) in the interval [0,1]. [58].

Definition: ***Normal Fuzzy Set*** A fuzzy set, say A, that has at least one element $x \in X$ such that $\mu_A(x) = 1$ [60].

Definition: ***Fuzzy Singleton*** A normal fuzzy set with a single support value [60].

The overall structure of the fuzzy controller can be described by three stages and a block diagram summarizes the structure graphically in Figure 2.1:

1. Fuzzyify crisp inputs from the real world.
2. Execute the inference process on the fuzzified inputs based on a fuzzy associative memory.
3. Defuzzify the output to crisp outputs to be used in the real world.

The three major components of a fuzzy logic inference system are described in greater detail below pertaining specifically to the construction of a fuzzy logic controller:

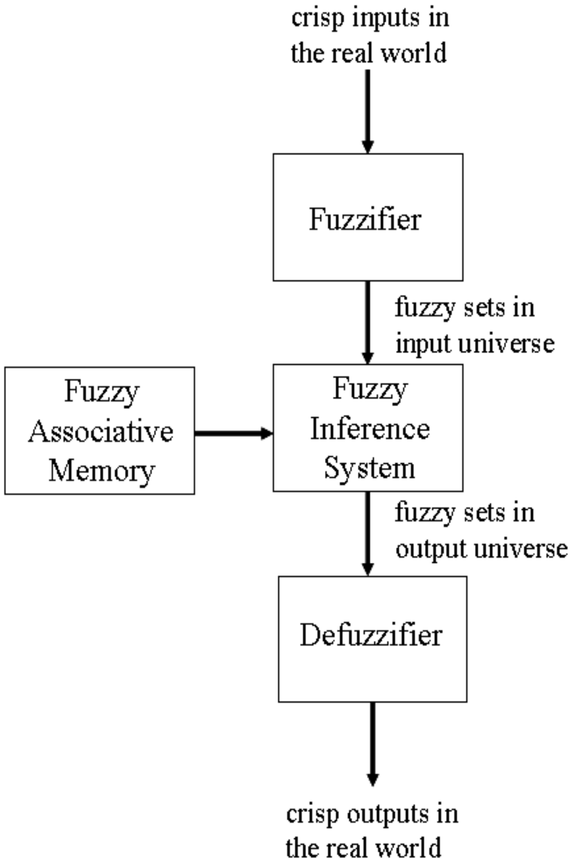

crisp inputs in the real world
Fuzzifier
fuzzy sets in input universe
Fuzzy Associative Memory
Fuzzy Inference System
fuzzy sets in output universe
Defuzzifier
crisp outputs in the real world

Figure 2.1: The process flow of a fuzzy logic inference system.

1. Fuzzify the input from physical domain to fuzzy domain.

   This can be achieved by normalizing the universe of discourse of the input and assigning weighing to the inputs in the form of input membership functions. Figure 2.2 shows an example of input membership functions.

   There are five input membership functions labelled in the form of A_i^l, which means l^{th} fuzzy set A and i^{th} input. $\bar{A}_i^l$ indicates the maximum value of l^{th} fuzzy set of input i. For any given input within the universe of discourse, the membership function values, μ, can be extracted from the relevant fuzzy sets. Section 5.4 describes the criteria used in the construction of the input membership functions, making them specific to this research application.

2. Formulate a mathematical rule-base that dictates the relationships between

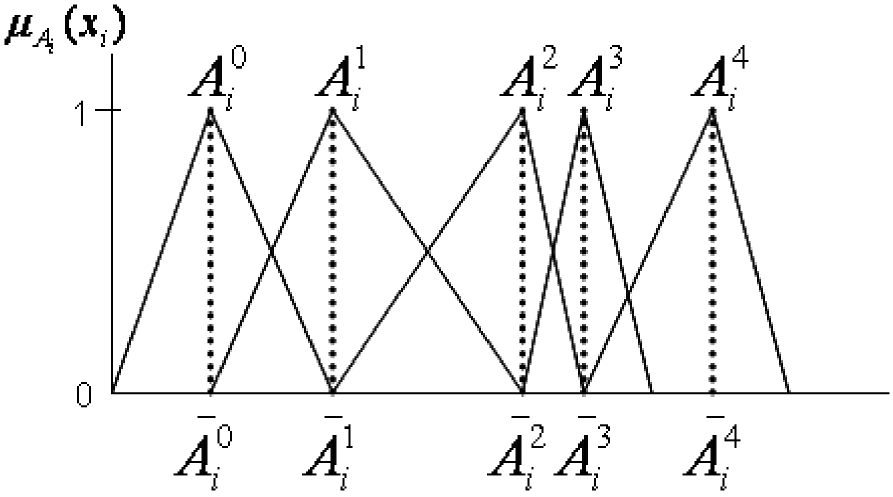


Figure 2.2: An example of input membership functions.

the fuzzy input and output sets.

The rule-base that maps input fuzzy sets to output fuzzy sets is called the fuzzy associative memory (FAM). If the input vector in the physical domain is $\mathbf{x} = (x_1, ..., x_n)^T$, then the l^{th} rule of a FAM can be expressed as the general form:

$$\text{If } x_1 \text{ is } A_1^l \text{ and ... and } x_n \text{ is } A_n^l \text{ Then } y \text{ is } B^l$$

where A_n^l refers to the l^{th} fuzzy set of input n, y is the output vector $\mathbf{y} = (y_1, ..., y_n)^T$ in the physical domain, B^l is the l^{th} output fuzzy set.

The rule verbally describes the rule that if the input(s) belong(s) to the specified fuzzy input set(s), then the physical output is in a specified fuzzy output set. To translate the verbal rule to mathematical form, output membership functions are required to relate to the fuzzy inputs. The criteria used for the construction of the output membership functions are described in Section 5.4.

3. Defuzzify the fuzzy output to an output in the physical domain.

   Let the l^{th} output fuzzy set be B^l and the output membership functions are fuzzy singletons such that

$$\mu_{B^l}(\hat{\mathbf{x}}) = 1 \quad \text{for sample } \hat{\mathbf{x}}$$
$$\mu_{B^l}(\mathbf{x}) = 0 \quad \text{for all the other } \mathbf{x}$$

where μ_{B^l} is the output membership function value of l^{th} fuzzy set B.

Then, simplification can be made to the weighted average of rule outputs and the defuzzified output in the physical domain can be calculated by:

$$y_0(\mathbf{x}) = \frac{\sum_{l=1}^{M} \bar{B}^l(\prod_{i=1}^{n} \mu_{A_i^l}(x_i))}{\sum_{l=1}^{M}(\prod_{i=1}^{n} \mu_{A_i^l}(x_i))} \tag{2.3}$$

Further simplification can be made to Equation 2.3 when a specific type of input membership functions are used. Again, details are included later in Sections 5.4 to 5.5.

Chapter 3

Dynamic Modelling of the Freedom 6S

The dynamic modelling of the Freedom 6S begins with incorporating the manipulator dynamics using traditional techniques found in [28]. Prior to dynamic descriptions, kinematics analysis is performed and the details are described in the next section.

3.1 Forward Kinematics

The kinematics of Freedom 6S is derived using the Denavit-Hartenberg (DH) representation [28]. In this convention, a product of transformation matrices is used to describe frame transformations. Each transformation, A, is represented as a multiplication of four fundamental transformations:

$$A = Rot_{z,\theta} Trans_{z,d} Trans_{x,a} Rot_{x,\alpha} \tag{3.1}$$

The first fundamental transformation is rotation about the z-axis by an angle of θ, followed by a translation along the z-axis by a distance of d. The third transformation is a translation along the x-axis by a distance of a and followed by a rotation about the x-axis by an angle of α. Evaluating the expression in Equation 3.1 from left to right, each basic transformation uses the local axes as reference. The closed chain schematic is shown on Figure 3.1. Since the DH representation is used for serial chain, two chains are required to described the closed kinematic chain of the Freedom 6S.

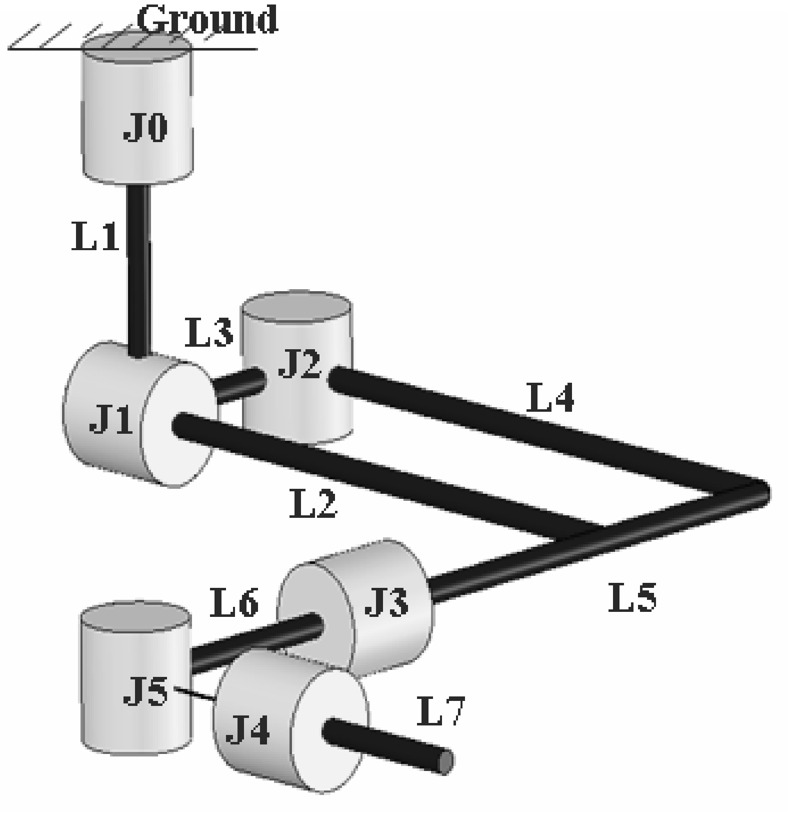


Figure 3.1: A schematic showing the closed chain kinematics of the Freedom 6S.

The first serial chain contains frames of reference describing links 2, 5, 6, and 7. The graphical representation is given in Figure 3.2. The frames are referred to as "f_x", where "x" denotes the frame number. These notations are used throughout the thesis for the ease of frame referencing. For example, Figure 3.2 shows the vectors for x_0 and z_0. These vectors define frame 0 (f_0). The link parameters corresponding to these frames are shown in Table 3.1.

The second chain describes frames of reference attached to links 3 and 4. The frame assignments are shown in Figure 3.3 and the associating link parameters are described in Table 3.2. The remaining link, L1, is described by a frame coinciding with f_0 as shown in both Figure 3.2 and Figure 3.3.

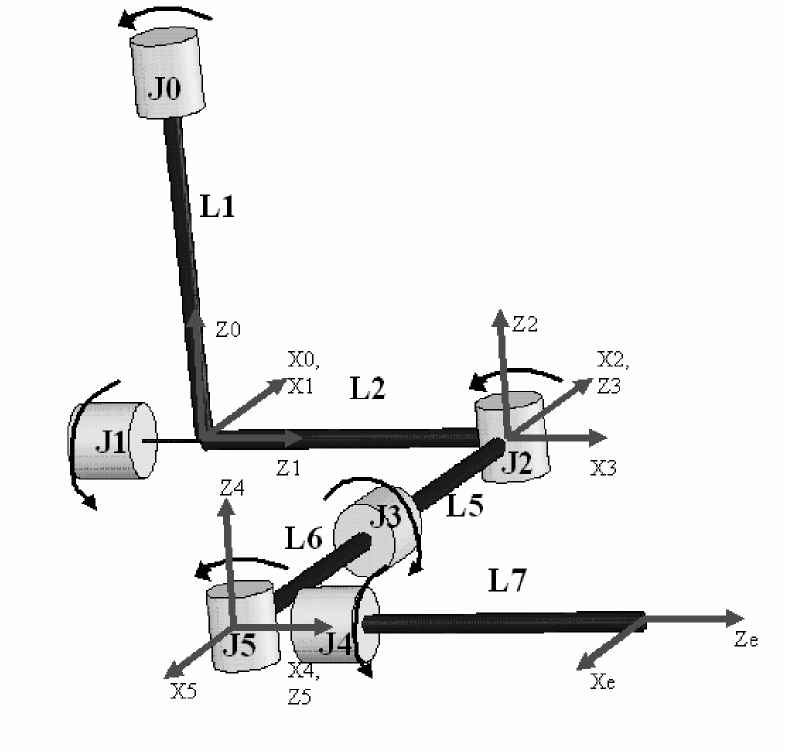


Figure 3.2: D-H frame assignments for masses of links 2, 5, 6 and 7

To verify that the kinematics are accurate, two tests are performed. The first test is to verify that the Cartesian endpoint found from Figure 3.2 transformation from f_0 to f_e agrees with the endpoint given from the software development kit (SDK) of Freedom 6S. This ensures that the frame transformation describing links 1, 2, 5, 6 and 7 is correct. When comparing the calculated endpoint from the frame assignment developed with that of the SDK of Freedom 6S, it is important to note that the frames of reference used by the two sources are different. Appendix A shows the relationship between the endpoint coordinates obtained by the DH representation and from the SDK as a reference.

Once the endpoint position is verified, a second test can be used to verify the consistency of the kinematics done in Table 3.2. For links 1, 3 and 4, the consistency of the frame transformations is confirmed by matching the origin of f_e in Figure 3.3 with a vector with length of link 2 extending from f_3 of Figure 3.2 in the z-direction.

Frame Transformation	Rot(z)	Trans(z)	Trans(x)	Rot(x)
f0 to f1	q0	0	0	$\pi/2$
f1 to f2 (used to describe c.o.m of L2)	q1	L2	0	-$\pi/2$
f2 to f3 (used to describe c.o.m of L5)	q2-$\pi/2$	0	0	-$\pi/2$
f3 to f4 (used to describe c.o.m of L6)	q3	-L5-L6	0	$\pi/2$
f4 to f5	q5-$\pi/2$	0	0	-$\pi/2$
f5 to fe (used to describe c.o.m of L7)	q4	L7	0	0

Table 3.1: DH parameters corresponding to Figure 3.2

Frame Transformation	Rot(z)	Trans(z)	Trans(x)	Rot(x)
f0 to f1	q0	0	0	$\pi/2$
f1 to f2	q1	0	0	-$\pi/2$
f2 to f3 (used to describe c.o.m of L3)	q2	0	L3	0
f3 to fe (used to describe c.o.m of L4)	-q2-$\pi/2$	0	L4	0

Table 3.2: DH parameters corresponding to Figure 3.3

3.2 Lagrangian Dynamics

Using the forward kinematics, a 6 DOF analytical dynamic model is developed for the Freedom 6S using the Lagrangian approach. The components in the equations of motion (i.e. the inertia matrix, Christoffel matrix, the gravity effect matrix) derived from Euler-Lagrange equations utilize the geometric relationships derived from the kinematic analysis. Information on rotational matrices and displacement vector describing the location of the center of mass of each link are required to calculate the inertia matrix, Christoffel matrix, and the gravity effect matrix.

The inertial matrix, $\boldsymbol{D}$, is a function of $\boldsymbol{q}$ only. It is derived from isolating the inertia term when the total kinetic energy of the links is considered. The inertia matrix is calculated as follows:

$$D = \sum_{i=1}^{n} m_i J_{v_{ci}}(q)^T J_{v_{ci}}(q) + J_{\omega_i}(q)^T R_i(q) I_i R_i(q)^T J_{\omega_i}(q) \tag{3.2}$$

where n is the number of links of the robot, m_i is the mass of link i, $J_{v_{ci}}$ is the 3x6

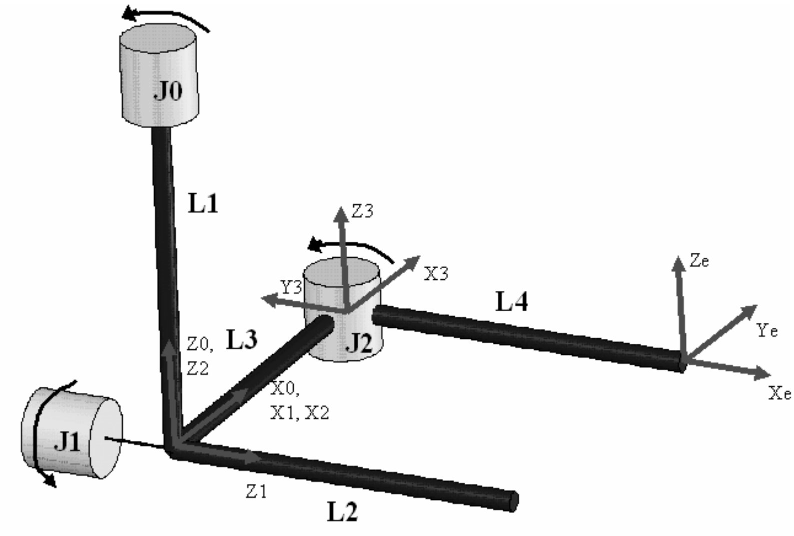


Figure 3.3: D-H frame assignments for masses of links 1, 3 and 4

linear portion of the Jacobian matrix based on the center of mass of link i, J_{ω_i} is the 3x6 angular portion of the Jacobian matrix based on the center of mass of link i, R_i is the 3x3 rotational matrix describing link i based on an inertial reference frame, I_i is the 3x3 inertia matrix. The Jacobian matrix and the rotational matrix can be computed/extracted from the kinematic analysis done previously.

The calculation of the Christoffel matrix, $\boldsymbol{C}$, is based on simplifications made during the derivation from Euler-Lagrange. The k,j-th element of C is calculated by:

$$C_{kj} = \sum_{i=1}^{n} \frac{1}{2}(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k})\dot{q}_i \tag{3.3}$$

where d_{kj} is the k,j-th element of the $\boldsymbol{D}$ matrix, q_i is the joint position of link i, $\dot{q}_i$ is the joint velocity of joint i.

The gravitational effect of the links on the dynamic of the system is described by ϕ. It is calculated by differentiating the potential energy with respect to each joint position. The individual element in the 6x1 vector of ϕ can be calculated by:

$$\phi_k = \frac{\partial V(q)}{\partial q_k} \tag{3.4}$$

where V is the sum of all the gravitational potential energy of the links as a function of $\boldsymbol{q}$, and q_k is the position of joint k.

The calculation is accomplished by determining dynamic relationships using symbolic programming in Maple[1]. A schematic of the linkages and joints are shown previously in Figure 3.1. Simplifications are done symbolically at each intermediate calculation to reduce the memory required to store the result. Some simplifications made include reducing the number of unknowns in a 3x3 inertia matrix from 9 parameters to 6 parameters by noting that an inertia matrix is symmetric. The complete model presented in Equation 3.5 includes inertia torques (product of a 6x6 inertia matrix, $\boldsymbol{D}$, and a 6x1 joint acceleration vector, $\ddot{q}$), centrifugal and Coriolis torques (product of a 6x6 Christoffel matrix, $\boldsymbol{C}$, and a 6x1 joint velocity vector, $\dot{q}$), and a 6x1 gravitation effect vector (ϕ). The input to the system is a 6x1 torque vector (τ). The individual joint position, $\boldsymbol{q}$, is defined with a subscript starting from 0 (i.e. $\boldsymbol{q}$=$\{q_0, q_1, q_2, q_3, q_4, q_5\}$).

$$\tau = D * \ddot{q} + C * \dot{q} + \phi \tag{3.5}$$

Equation 3.5 is the equation of motion derived from Euler-Lagrange equations. Appendix B shows the detailed derivation from the Euler-Lagrange equations to the form appears in Equation 3.5.

Despite the fact that motors 1 and 2, as indicated by joints 1 and 2 in Figure 3.1 respectively, are mounted onto the linkages themselves, their dynamic effects (i.e. gyroscopic effect) caused by the rotor rotation is negligible in comparison to the inertia forces from the actual masses of the motors. This is because the two motors are directly-driven and are not operated at high speeds. Hence, the effect of the motors can be incorporated to Equation 3.5 by including the masses of the motors onto the appropriate links. In this case, the motor and the link to which the motor is attached would be grouped as one rigid body. Specifically, the mass of motor 1 is included with that of link 2 and the mass of motor 2 is included with that of link 3. The calculation of all the terms in Equation 3.5 is done symbolically in Maple and the script is included in Appendix C.

[1] A mathematical application software with symbolic computation developed by Maplesoft. www.maplesoft.com

3.3 Friction Modelling

In addition to the Lagrangian dynamics, other effects must be incorporated to produce a more accurate model. Friction is a dominating factor in the modelling of Freedom 6S and it must be included to properly describe the behaviour of the device.

There are many different types of friction modelling and they can be divided into two general categories: static models and dynamic models. Static models are simple functions of displacement or velocity and they are easy to implement. However, they are not able to capture many characteristics of friction, such as effects from deflection between two contact surfaces. The static friction model is suitable for simple applications. It is easy to compute and it offers a fast simulation result. On the other hand, dynamic models offer a more detailed description of friction at the cost of computation complexity. For Freedom 6S, both models are investigated and they are discussed in details below.

3.3.1 Static Friction Models

Static models are usually based on functions of velocity. The Coulomb friction model, for example, represents Coulomb friction as a value with constant magnitude but with a direction opposite to the direction of the velocity. Another type of friction, static friction, describes the force that must be overcame before motion can result. This friction is of high magnitude and it only has a value when the velocity of the object is zero. Once the object is in motion, kinetic friction results and it is generally modelled as a constant value when velocity is greater or less than zero. In addition to kinetic friction, viscous friction also occurs when velocity is not zero and it increases linearly with speed. Another friction effect used in modelling is called the Stribeck effect. It occurs during low velocities at which the friction force decreases continuously with increasing velocities [61].

A combined static friction model with all the static effects (i.e. Coulomb, static, kinetic, viscous, and Stribeck) is displayed in Figure 3.4. This particular model is proposed by Turner [62] and it has no discontinuity which is a more realistic behaviour of friction. This static friction model is combined with the Lagrangian model as an initial trial.

The static friction, which is the force required for an object to have motion, is described as the first peak of Figure 3.4. As this is a velocity-dependent curve, the static friction modelled on this curve is an approximation only. The kinetic friction parameter, f_k, describes the force needed to maintain the velocity of the object once it has overcome the static friction. Viscous friction is modelled when velocity is greater than V_4 and it increases with the velocity by a slope of s_2.

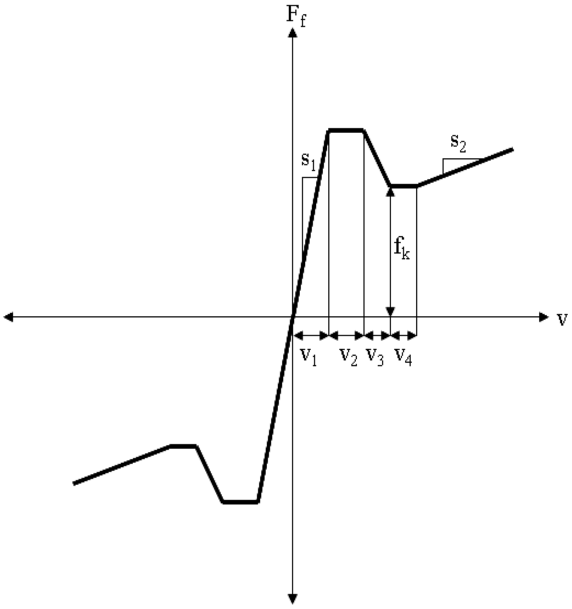


Figure 3.4: A Continuous static and kinetic friction model.

The five parameters listed on Figure 3.4 are required to characterize the friction model. As mentioned previously, f_k is the kinetic friction. The parameter, s_1, describes the steep slope that continues until the static friction is reached at a velocity of v_1. The other slope parameter, s_2, is the viscous damping coefficient.

Unfortunately, this model is inadequate in capturing all the nonlinear frictional behaviour of Freedom 6S. In particular, it has problems with modelling the behaviour of the joints when the velocities are low. When cosine input torque commands are given to the wrist joints, the observed joint responses take the shape of plateaus at the peaks and valleys of the position curves. However, the simulations of the joint positions often have rounded peaks and valleys on the response curves and these features do not describe the existence of friction during low velocities. Hence, a dynamic friction model is investigated.

3.3.2 Dynamic Friction Models

A dynamic friction model is considered since the simple friction model cannot accurately describe the friction observed from the Freedom 6S. Dynamic friction models treat the friction force as a state of the system. Dahl [63] modelled stiction as having a spring-like behaviour. The model describes a delayed effect in Coulomb friction when the velocity changes sign. The Dahl model, however, does not account for Stribeck effect. Another dynamic friction model is proposed by Armstrong-Helouvry [64]. Their model is consisted of seven parameters and it keeps the different friction effects as separate models by having one model to describe stiction and another model to describe sliding fricton.

One well-known dynamic friction model is the LuGre friction model. In particular, the model represents two contacting surfaces as contact between bristles. These bristles on the surfaces are deformed during contact. This deformation is referred to as the "deflection of the contact surfaces". The LuGre model embodies a combination of friction effects such as the deflection of the contact surfaces, Stribeck effect, stiction, Coulomb and viscous frictions. It has been proven to be capable of capturing friction characteristics observed in many applications [65], [66]. It is chosen to be used with the modelling of Freedom 6S because of its ability to include all the friction effects discussed and its easy implementation. The model treats the deflection of contact surfaces as an internal state to dynamically describe the behaviour of friction [67]. The governing equations are presented in Equations 3.6 to 3.8.

$$\begin{aligned} F_{friction} &= \sigma_0 \times z + \sigma_1 \times \frac{dz}{dt} + \sigma_2 \times v && (3.6) \\ \frac{dz}{dt} &= v - \sigma_0 \times \frac{|v|}{g(v)} \times z && (3.7) \\ g(v) &= F_c + (F_s - F_c) \times e^{-(v/v_s)^2} && (3.8) \end{aligned}$$

where σ_0 is the stiffness of the contact surface, z is the average deflection of the contact surface, σ_1 is the damping coefficient of the contact surface, σ_2 is the viscous damping, v is the velocity, F_c is the Coulomb friction, F_s is stiction, v_s is Stribeck velocity.

Equation 3.6 is the combined friction effect based on the stiffness and damping effects of the contact surface and viscous friction. Equation 3.7 describes the rate of change of deflection of the contact surface. It changes proportionally to velocity and deflection but inversely proportional to Coulomb friction and stiction. Equation 3.8 is used to describe the stick and slip behaviour of friction and it embodies the Stribeck effect. The parameters characterizing these equations are included the list of unknowns to be identified. Table 3.3 gives a summary of the notation used to represent these parameters along with their descriptions.

To have a complete model, other effects affecting the manipulator behaviour must be included for system identification. For the Freedom 6S, these effects include plant friction, (amplifier) gain and cable drag and they are discussed in the next section.

3.4 Other Effects in the Model

The additional effects included in the Freedom 6S model are described below.

3.4.1 Plant Gain: Amplifier and Gearing

For simplicity, the six amplifier gains are combined with the gains from gearing. The amplifier gains for the six inputs to the motors require identification since the actual gains of the amplifier might have changed from the time when it was manufactured. In terms of gearing, the first three motors are direct drive motors and thus they have 1:1 input-to-output mechanical ratios. In contrast, the wrist joints are driven by tendons routing via a set of pulleys. The mechanical ratio is included with the amplifier gain to form an overall plant gain for each joint.

3.4.2 Cable Drag

There are cables/wires coming out of the direct drive motors of the first three joints. They must be taken into account when modelling since the cables exert some pulling forces on to the casings of the motors, which in turn affects the position of the links. As for the wrist joints, the wires of the sensors are mounted at the joints and they would have similar effects on the links controlling the wrist. Hence, a cable drag effect is incorporated to all joints in the model. It can be observed that if no persistent external force is applied, a joint would return to its equilibrium position.

Hence, a spring term would mimic the torsional elasticity observed at each joint. The implementation is achieved by introducing spring torques:

$$\tau_s = K_s \times (\mathbf{q} - q_{eq}) \tag{3.9}$$

where τ_s is the torque vector resulting from the springs, K_s is the torsional spring stiffness vector, $\mathbf{q}$ is the position vector of the joints, q_{eq} is the equilibrium positions vector of the joints.

3.5 Chapter Summary: Overall Model

Table 3.3 shows the additional parameters for each joint required for identification. These unknowns are the characteristics that describe the dynamic behaviour of Freedom 6S in addition to the inertial dynamics.

Notation	Description
Kp_i	Plant gain (gearing and amplifier gain)
q_{eq_i}	Equilibrium positions of joints
Ks_i	Spring constant (modeling cable drag)
σ_{0_i}	Stiffness of bristles (friction parameter)
σ_{1_i}	Damping coefficient of bristles (friction parameter)
σ_{2_i}	Damping coefficient of links (friction parameter)
v_{s_i}	Stribeck velocity (friction parameter)
F_{s_i}	Stiction friction force (friction parameter)
F_{c_i}	Coulomb fricton force (friction parameter)

Table 3.3: Additional model parameters for a joint.

Incorporating the additional parameters outlined in Sections 3.2-3.4, the overall model describing the manipulator is shown in Figure 3.5.

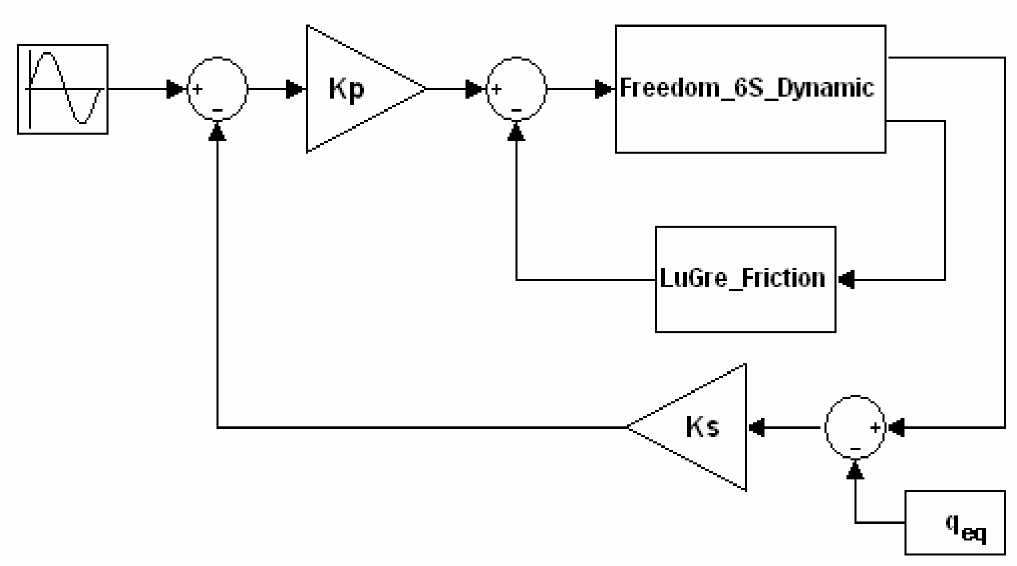


Figure 3.5: Block diagram of the overall model of the Freedom 6S.

Chapter 4

System Identification of the Freedom 6S

Once the form of the model is developed, the unknown parameters have to be identified. The nonlinear model determined from Chapter 3 has 118 unknown parameters, making it highly complex. Systematic approaches are warranted in order to organize the unknowns into groups from which identification can be done efficiently. This chapter outlines two system identification approaches to model the Freedom6S.

4.1 System Identification by the Method of Isolated Joint

The essence of the first system identification method proposed in this thesis is to group the unknown parameters from the model by joint. The identification process is iterative in nature. One important concept of method of isolated joint is that the identification process only allows one free joint to move at a time while all the other joints are locked. The joints of the manipulator need to allow individual locking to facilitate data collection used to determine the unknowns associated with each joint separately. By physically restricting the movement of joints, the dynamic equations can be simplified and hence the required number of unknowns to be identified is reduced for a given fixed joints configuration. By fixing the joints in a special order, a smaller number of unknowns can be identified for a given fixed joint test. The following section outlines the steps required to identify the system by using the method of isolated joint.

4.1.1 The Locking Sequence

Once the model equations from Chapter 3 are developed, a “locking sequence” can be established. On the Freedom 6S device, the joints are constrained mechanically to ensure locking occurs. The positions of the locked joints need not to be at their home positions as long as no relative movement is allowed on that joint. If a torque input is given to the free joint and no joint displacements are recorded from the sensors of the locked joints, the locked joints are considered well-restrained. This condition is enforced during each test run recorded for system identification.

The concept of the sequence at which the joints are locked can be illustrated by a simple example. Suppose there is a serial robot with 3 DOFs. When joints 1 and 2 are locked, their respective joint velocities and accelerations are zero. This reduces the original equations of motion describing a 3 DOF system to a 1 DOF equation of motion describing only joint 3 (see Figure 4.1).

$$\begin{bmatrix}\tau_1\\ \tau_2\\ \boxed{\tau_3}\end{bmatrix}=\begin{bmatrix}M_{11} & M_{12} & M_{13}\\ M_{21} & M_{22} & M_{23}\\ M_{31} & M_{32} & \boxed{M_{33}}\end{bmatrix}\cdot\begin{bmatrix}\ddot{q}_1\\ \ddot{q}_2\\ \boxed{\ddot{q}_3}\end{bmatrix}+\begin{bmatrix}C_{11} & C_{12} & C_{13}\\ C_{21} & C_{22} & C_{23}\\ C_{31} & C_{32} & \boxed{C_{33}}\end{bmatrix}\cdot\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\\ \boxed{\dot{q}_3}\end{bmatrix}+\begin{bmatrix}\phi_1\\ \phi_2\\ \boxed{\phi_3}\end{bmatrix}$$

Figure 4.1: The reduced equation by locking joints 1 and 2 is highlighted by the borders.

By studying the 1 DOF equations of each joint separately, the unknowns parameters from each equation can be determined. A table showing the masses of the links of the example robot with respect to the 1 DOF equations are shown in Table 4.1.

From Table 4.1, the 1 DOF equation describing joint 1 contains masses from all three links while the 1 DOF equation describing joint 3 only contains one unknown (i.e. the mass of link 3). Hence, the locking sequence should be done by freeing

Parameter	Joint 1	Joint 2	Joint 3
Mass of Link 1	$\surd$	X	X
Mass of Link 2	$\surd$	$\surd$	X
Mass of Link 3	$\surd$	$\surd$	$\surd$
Number of Parameters	3	2	1

Table 4.1: The concept of locking sequence.

only joint 3 (i.e. locking joints 1 and 2) first to determine the mass of link 3. Then, the next test should be followed by freeing only joint 2 (i.e. locking joints 1 and 3). At this point, the mass of link 2 is the only unknown since the mass of link 3 is determined in the previous test. The last test is then to free only joint 1 (i.e. locking joints 2 and 3) to determine the remaining parameter of mass of link 1.

Figure 4.2 has a list of unknown parameters to be determined by the associated locked joint configuration for Freedom 6S. The cells in which the number "1" is listed means the corresponding parameters are to be determined from the respective configuration test. The nomenclature used to describe the parameters is as follows: "I" prefix refers to the inertia (e.g. $I1_{12}$ means the inertia of link 1 and it is the inertia element (1,2) in the 3x3 inertia matrix); "m" prefix refers to the mass of the link (e.g. m_1 means the mass of link 1); "x", "y", "z" prefixes refer to the centroid location of the link (e.g. x_1 means the x position of the centroid of link 1). Starting with joint 4, each configuration uses parameters determined by the previous configuration in order to reduce the number of unknown parameters for the configuration at the next step. By listing the parameters associated with each 1 DOF dynamic equation, it shows that an additional test of keeping joints 0 and 1 free at the same time is necessary to determine all the unknown parameters of the system. Thus, there are seven configurations to be tested. Table 4.2 summarizes the locking sequence of Freedom 6S.

4.1.2 Additional Parameters

In addition to the Lagrangian dynamics, friction and other effects must be incorporated to produce a more accurate model as mentioned in Sections 3.3-3.4 of Chapter 3. These unknowns (i.e. Kp_i, $epos_i$, Ks_i, σ_{0_i}, σ_{1_i}, σ_{2_i}, v_{s_i}, F_{s_i}, F_{c_i}) are incorporated

Parameters	**Configuration**	**Parameters**	**Configuration**	**Parameters**	**Configuration**
	1 2 3 4 5 6 7		**1 2 3 4 5 6 7**		**1 2 3 4 5 6 7**
$I1_{11}$	0 0 0 0 0 0 0	$I3_{11}$	0 0 0 0 1 0 0	$I5_{11}$	0 0 0 0 1 0 0
$I1_{12}$	0 0 0 0 0 0 0	$I3_{12}$	0 0 0 0 1 0 0	$I5_{12}$	0 0 0 0 0 1 0
$I1_{13}$	0 0 0 0 0 0 0	$I3_{13}$	0 0 0 0 0 1 0	$I5_{13}$	0 0 0 0 1 0 0
$I1_{22}$	0 0 0 0 0 0 0	$I3_{22}$	0 0 0 0 1 0 0	$I5_{22}$	0 0 0 1 0 0 0
$I1_{23}$	0 0 0 0 0 0 0	$I3_{23}$	0 0 0 0 0 1 0	$I5_{23}$	0 0 0 0 0 1 0
$I1_{33}$	0 0 0 0 0 1 0	$I3_{33}$	0 0 0 1 0 0 0	$I5_{33}$	0 0 0 0 1 0 0
m_1	0 0 0 0 0 1 0	m_3	0 0 0 1 0 0 0	m_5	0 0 0 1 0 0 0
x_1	0 0 0 0 0 1 0	x_3	0 0 0 1 0 0 0	x_5	0 0 0 1 0 0 0
y_1	0 0 0 0 0 1 0	y_3	0 0 0 1 0 0 0	y_5	0 0 0 0 1 0 0
z_1	0 0 0 0 0 0 0	z_3	0 0 0 0 1 0 0	z_5	0 0 0 1 0 0 0
$I2_{11}$	0 0 0 0 0 1 0	$I4_{11}$	0 0 0 0 1 0 0	$I6_{11}$	0 0 0 1 0 0 0
$I2_{12}$	0 0 0 0 0 0 1	$I4_{12}$	0 0 0 0 0 0 1	$I6_{12}$	0 0 0 0 1 0 0
$I2_{13}$	0 0 0 0 0 1 0	$I4_{13}$	0 0 0 0 0 0 1	$I6_{13}$	0 0 0 1 0 0 0
$I2_{22}$	0 0 0 0 1 0 0	$I4_{22}$	0 0 0 0 0 1 0	$I6_{22}$	0 0 1 0 0 0 0
$I2_{23}$	0 0 0 0 0 0 1	$I4_{23}$	0 0 0 0 0 1 0	$I6_{23}$	0 0 0 0 1 0 0
$I2_{33}$	0 0 0 0 0 1 0	$I4_{33}$	0 0 0 0 0 1 0	$I6_{33}$	0 0 0 1 0 0 0
m_2	0 0 0 0 1 0 0	m_4	0 0 0 1 0 0 0	m_6	0 0 1 0 0 0 0
x_2	0 0 0 0 1 0 0	x_4	0 0 0 0 0 1 0	x_6	0 0 1 0 0 0 0
y_2	0 0 0 0 0 1 0	y_4	0 0 0 0 1 0 0	y_6	0 0 0 1 0 0 0
z_2	0 0 0 0 1 0 0	z_4	0 0 0 0 1 0 0	z_6	0 0 1 0 0 0 0
				$I7_{11}$	0 1 0 0 0 0 0
				$I7_{12}$	0 1 0 0 0 0 0
				$I7_{13}$	0 0 1 0 0 0 0
				$I7_{22}$	0 1 0 0 0 0 0
				$I7_{23}$	0 0 1 0 0 0 0
				$I7_{33}$	1 0 0 0 0 0 0
				m_7	1 0 0 0 0 0 0
				x_7	1 0 0 0 0 0 0
				y_7	1 0 0 0 0 0 0
				z_7	0 1 0 0 0 0 0

Summary	**Configuration**						
	1	**2**	**3**	**4**	**5**	**6**	**7**
# of param	4	4	6	13	17	16	4

Figure 4.2: Parameters of links 1 to 7 listed by order of locking configuration.

with the dynamic equations as described by Figure 3.5 and they must be included with each joint during system identification.

4.1.3 Optimizing Locked Joints Parameters

After implementing the friction model and incorporating all the parameters into the model shown in Figure 3.5, a nonlinear optimization is done in an effort to find parameters that best fit the measured data. Note that optimization is done on the open-loop system rather than the closed-loop system, allowing friction effects to be more observable and giving a higher sensitivity to any changes in parameter.

It is found that the amplifier gain of the system is highly nonlinear. Initially, chirp signals as listed in Table 4.3, with various frequency ranges are used to determine the specific ranges suitable for each joint. The input signals begin at 0.5-1 Hz and are increased to the 2 to 3 Hz on the upper range as this is near the maximum operating frequency of the joints.

At high frequencies (i.e. t≈80s), the amplifier gain exhibits nonlinear behaviour. When the frequency is beyond a threshold value, the gain changes with increasing frequency. As a result, torque inputs within a certain frequency range, as listed in Table 4.4, are explored to avoid the nonlinear gain regions. The ranges of frequency used for experiments start from the lowest frequencies of the chirp signals to the highest frequencies at which the amplifier gains are still linear. For tests on each joint, five input cosine torques with the same amplitude and with an uniformly increase in frequencies are used for joint excitation. However, not all the measured data from these inputs are used for optimization because of non-uniform system behaviour. Since these ignored sets of measured data have much deviation from the rest of the data, convergence cannot be achieved by the optimization routine. Hence, it is decided that they can be ignored. Table 4.4 lists the set of inputs used for each joint. At different frequencies, different types of friction effects would dominate and these measured data is used as a basis for the optimization to find the suitable friction parameters.

The optimization utilizes a sequential quadratic programming method[1] (see Sec-

[1] The command used in Matlab optimization toolbox is "fmincon".

Configuration	Joint0	Joint1	Joint2	Joint3	Joint4	Joint5
1	L	L	L	L	U	L
2	L	L	L	L	L	U
3	L	L	L	U	L	L
4	L	L	U	L	L	L
5	L	U	L	L	L	L
6	U	L	L	L	L	L
7	U	U	L	L	L	L

Table 4.2: The locking sequence of joints: L = Locked, U = Unlocked

Free Joint	Amplitude (A) (N*m)	Frequency ($\omega(t)$) (rad/s)
J0	0.02000	1+(1/60)*t
J1	0.01500	1+(1/60)*t
J2	0.02000	1+(2/60)*t
J3	0.008000	1+(1.5/60)*t
J4	0.005000	1+(2.5/60)*t
J5	0.008000	1+(1.5/60)*t

Table 4.3: Initial input chirp signals of the form A*cos($\omega(t)$*t).

Free Joint	Amplitude (A) (N*m)	Frequency (ω) (rad/s)
J0	0.02000	1.000; 1.325; 1.650; (1.975); (2.300);
J1	0.01500	1.000; 1.208; 1.417; 1.625; (1.833);
J2	0.02000	1.000; 1.500; 2.000; (2.499); (2.300);
J3	0.008000	0.5000; 0.8750; 1.250; 1.625; 2.000;
J4	0.005000	0.5000; 1.125; 1.750; 2.375; (3.000);
J5	0.008000	0.5000; 1.000; 1.500; 2.000; 2.500;
J0, J1	0.01600, 0.01500	1.500, 1.000; 1.604, 1.104; 1.708, 1.2082; 1.812, 1.312; 1.917, 1.417;

Table 4.4: Input torque for locked joint tests with the frequencies used by each test separated by a ";". The inputs with frequencies in "()" are not used for optimization due to non-uniform system behaviour. The last group of input torques are sent to J0 and J1 simultaneously to collect coupling joint test result.

tion 2.4 for concepts). The inputs required are the initial guesses of the parameters with their respective lower and upper bounds. In order to make good initial guesses, a SolidWorks[2] model of Freedom 6S is made by using the approximated physical measurement of the device. The physical measurements are done with the assembled device and some assumptions are made in order to model the parts in SolidWorks. Motors moving joints 1 and 2 are modelled as cylinders with their housings. The weight of the cables and wires wrapped around the links are not included in the SolidWorks model. Values of inertia, mass and centroid location from this Solid-

[2] A solid modeling package. www.solidworks.com.

Works model are used as initial guesses for parameters listed in Figure 4.2. The initial guesses from SolidWorks are presented in Appendix D.

Since the links of the robot are made out of aluminum alloy, all the solid modelling assumes the material use of 2014 aluminum alloy. Due to this uncertainty and measurement inaccuracies, the range between the lower and upper bounds are estimated to be +/-50 percent and +/-30 percent for mass and centroid parameters respectively. Once these two ranges are decided, the range between upper and lower bounds for inertia parameters can be calculated by error propagation (Appendix E). As part of the initial conditions, the initial deformation value (i.e. z in Equation 3.6) in the LuGre friction model is assumed to be 0. This is a reasonable assumption considering no movement is present initially and hence, any deformation due to the interaction of links on a joint is negligible. In addition to the initial conditions, an inequality constraint is added to restrict the optimized value of the stiction force to be larger than that of the Coulomb friction.

$$Fs_i \geq Fc_i \tag{4.1}$$

where Fs_i is the stiction force of joint i, Fc_i is the Coulomb friction force of joint i.

The cost function used for evaluation is the sum squared error between the measured data and the simulation data. A time weighing factor is incorporated into the cost function to minimize the transient effect. During the initial data collection of each run, the position reading might not be accurate due to initialization procedure of the sensors during which spikes produced by the amplifier are a common phenomenon. Hence, the first half of each data set is weighted half as heavily than the remaining portion of the data set. The cost function used for optimization is

$$J = \Sigma_{i=1}^{n}(\lambda_i \times (y_i - \hat{y}_i))^2 \tag{4.2}$$

where J is the cost function, $i = \mathit{1..n}$ is the time step, λ_i is the time weighing factor at time i, y_i is the measured data at time i, and $\hat{y}_i$ is the simulated data at time i.

In order to minimize the processing time of the optimization function used in Matlab, the symbolic dynamic equations are exported to Microsoft Visual C++ (VC++) from Maple to create a dynamically link library (dll). This dll uses the parameters to be optimized as inputs and output the joint positions and velocities of the model at each time step. In addition, a nonadaptive 4th-order Runge-Kutta solver is also scripted in VC++ to facilitate the calculation of joint positions and velocities of the model.

The optimization is done for each locking sequence test as indicated in Table 4.2. Within each iteration, a quadratic programming subproblem is solved and the direction of the search is calculated at the end of each step. Convergence is achieved when the search direction changes less than 2e-6 and the maximum constraint violation is less than 1e-6[3]. Further information on this nonlinear search method can be found in [47].

4.1.4 Parameter Verification

After identification of all the parameters, they are used in the full model (i.e. substituting values into the full DOF model) for verification. The results from using the method of isolated joint are presented in Section 4.2 and the test setup used for all experimental tests is described in Appendix F.

4.2 Results from using the Method of Isolated Joint

Using the method of isolated joint, optimization is done on data sets for each joint. For each test, a set of cosine input torques is applied in sequence to the joint of interest (i.e. the unlock joint) as shown in Figure 3.5. The resulting data sets are used to evaluate the cost function for optimizing the parameters of each joint. Table 4.4 lists the input torques given to the device for generating measured data used for optimization. A fixed time step of 0.5ms and the ODE4 Runge-Kutta solver are used for the simulation of joints 0 - 5. This solver solves differential equations with a non-adaptive Runge-Kutta method of order 4.

4.2.1 Individual Joint Test Results

The results obtained from simulation using the optimized parameters and the measured data for individual joints are presented in Figures 4.3 to 4.10. Not all five

[3] These are the default values used by Matlab.

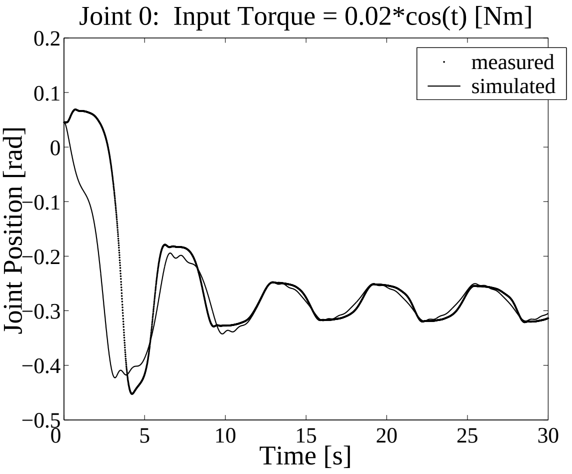


Figure 4.3: Measured and simulated joint positions of joint 0 with all other joints fixed.

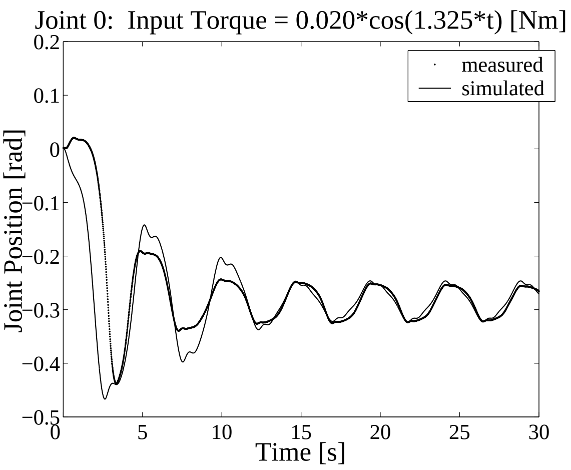


Figure 4.4: Measured and simulated joint positions of Joint 0 with all other joints fixed.

data sets for each joint are used for optimization because some collected data has variations that affected convergence. These are the “()” frequencies in Table 4.4. The results of using different frequency torque inputs are presented for joint 0, showing a sample result set of different input frequencies applied to the same joint. For joints 1-5, only the results of torque input at the middle frequencies (J1: ω=1.417; J2: ω=1.500; J3: ω=0.8750; J4: ω=1.750; J5: ω=1.000) are presented because the rest of the results show a similar trend. The simulation time is chosen such that a relevant number of cycles can be displayed.

In general, the simulation closely matches the measured data after the initial transient. The transient results between the observation and the simulation are different because the time weighing factors are implemented to weigh more heavily on the steady state data. The simulated joint positions of the base joints (Figures 4.3 to 4.7) show results that are in agreement with the measured data. Since these joints are driven directly by motors, less friction effects are expected. The results for joints 1-2 have consistent amplitudes and frequencies with the measured data. As for joint 0, a similar consistency is shown at the steady state response. It is worth noting that the measured data of the base joints exhibits the characteristics of a non-minimum phase system in which the initial inverse response occurs in the

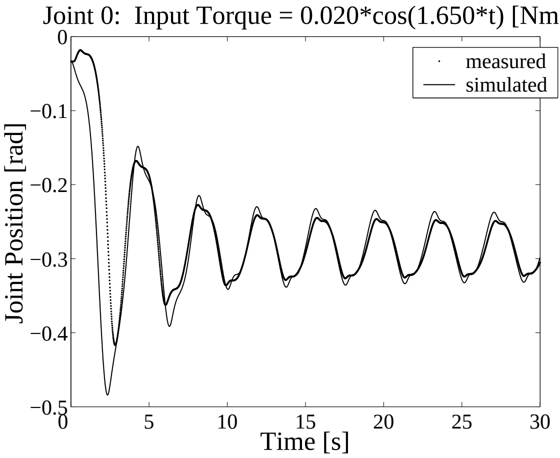


Figure 4.5: Measured and simulated joint positions of joint 0 with all other joints fixed.

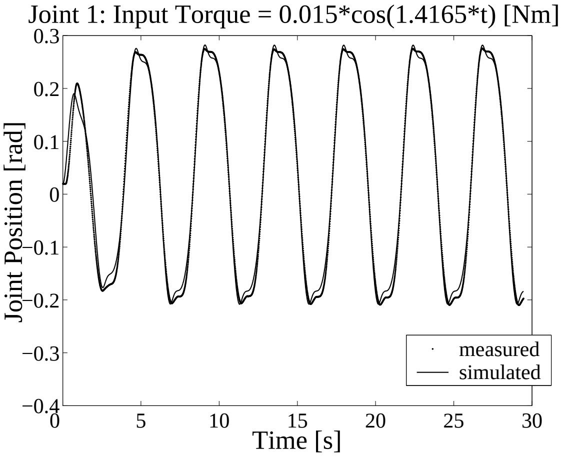


Figure 4.6: Measured and simulated joint positions of joint 1 with all other joints fixed.

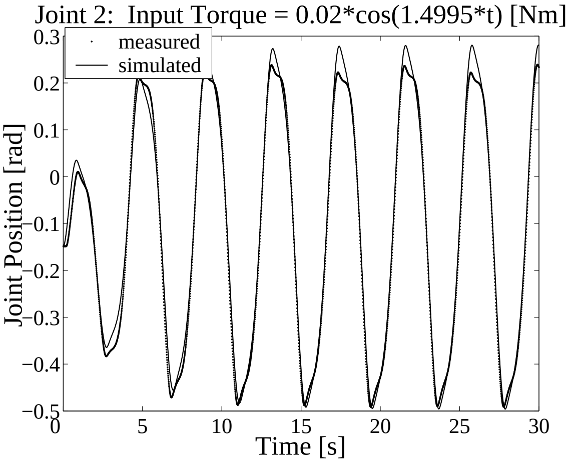


Figure 4.7: Measured and simulated joint positions of joint 2 with all other joints fixed.

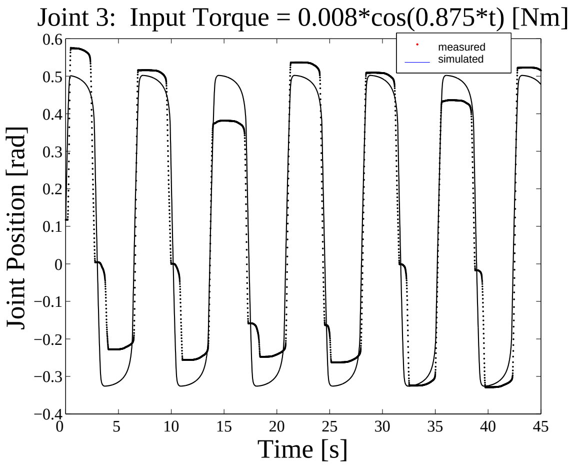


Figure 4.8: Measured and simulated joint positions of joint 3 with all other joints fixed.

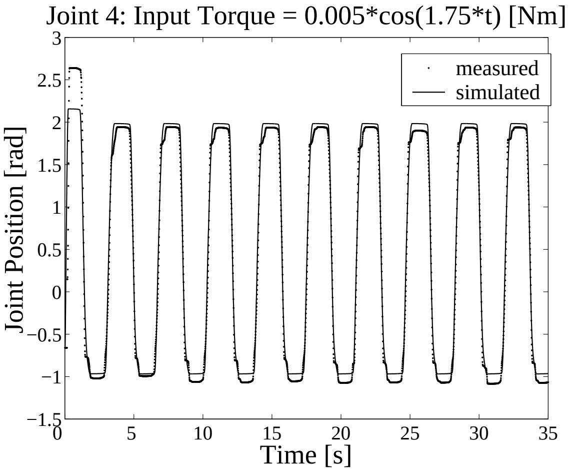


Figure 4.9: Measured and simulated joint positions of joint 4 with all other joints fixed.

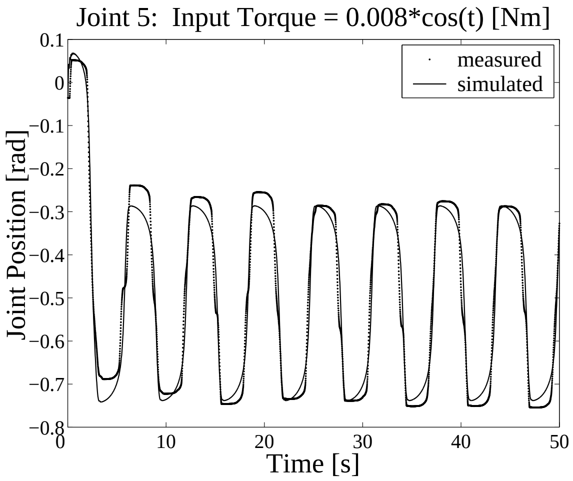


Figure 4.10: Measured and simulated joint positions of joint 5 with all other joints fixed.

opposite direction to the input. It can be speculated that this characteristic has a potential to cause instability in the system [68].

Figures 4.8 to 4.10 show the simulated joint positions of the wrist joints. The LuGre friction model is able to produce stiction effects that agree with the experiment because the plateau-shaped peaks and valleys of the joint positions are successfully simulated. The optimized parameters produce simulations that are able to match the frequency of the measured data completely, with acceptable variations in amplitudes. The measured data exhibits some randomness in amplitude (Figure 4.8) that cannot be accounted for in the non-stochastic dynamic model. This may be caused the intrinsic elasticity of the tendons used to transmit torques in the wrist joints, making their responses less predictable.

4.2.2 Joints 0 and 1 Coupling Test Result

From Table 4.2, the last test required to completely identify the unknown parameters is to perform a coupling test by sending excitation torques to move joints 0 and 1 simultaneously. Five measured data files (see last entry in Table 4.4 for the different input torques used to collect the data) are used in the optimization simultaneously to obtain the last group of parameters. Figures 4.11 to 4.12 shows the simulated results with the measured data from the first data file. Since similar pattern exists for comparison between simulated results and different input torques with other frequencies, the other data files comparisons are not displayed to avoid redundancy. The overall conclusion from the joints 0 and 1 coupling test result is that although the simulation result of joint 1 agrees with the respective measured result, the simulated result of joint 0 deviates much from its measured result. It is speculated that the inaccuracies in the parameters from other isolated joint tests in Section 4.2 have attributed to the mismatch observed in joint 0. Since the dynamic equations of joints 0 and 1 are dependent on previously determined parameters of joints 2 to 5, any inaccuracies in these pre-determined parameters would accumulate as the identification process continues to joints 0 and 1. As a result, the parameters describing the base joints do not converge properly in the presence of incorrect parameters describing the joints “downstream” (i.e. joints that are further away from the base joints). This is addressed by using another system identification approach, the method of coupling joints, as outlined in Section 4.3.

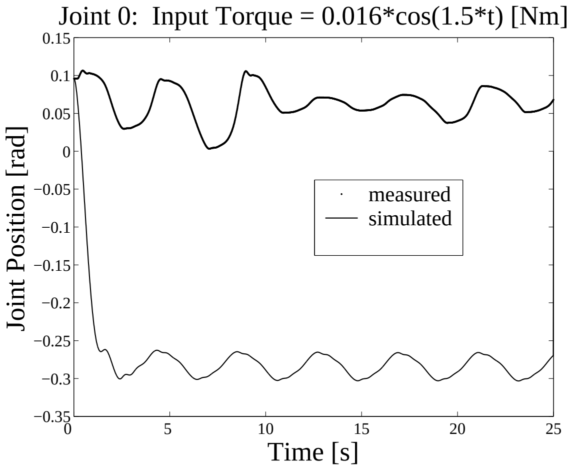


Figure 4.11: Measured and simulated joint positions of joint 0 with joints 2 to 5 fixed.

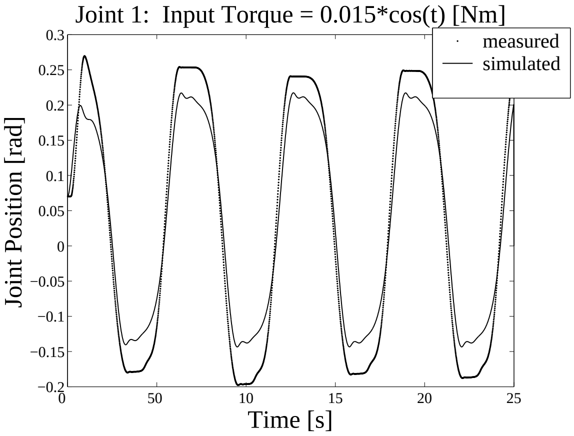


Figure 4.12: Measured and simulated joint positions of joint 1 with joints 2 to 5 fixed.

4.2.3 Model Verification

Despite the mismatch of joint 0 between the simulation and the measured result, a verification using the optimized parameters in the 6 DOF model is completed. A set of input torques, as listed in Table 4.5, are sent to all the joints simultaneously and the open-loop response is measured. Figures 4.13 to 4.18 display the measured and simulated results of each joint.

Free Joint	Amplitude (A) (N*m)	Frequency (ω) (rad/s)
J0	0.016	1.5
J1	0.015	1.0
J2	0.015	1.0
J3	0.0080	0.50
J4	0.0050	0.50
J5	0.0080	0.50

Table 4.5: Input torques for verification of model with parameters found using method of isolated joint.

There are unacceptable differences between the measured and simulated results for joints 0 and 5. The mismatch in joint 0 is likely a result of accumulation of

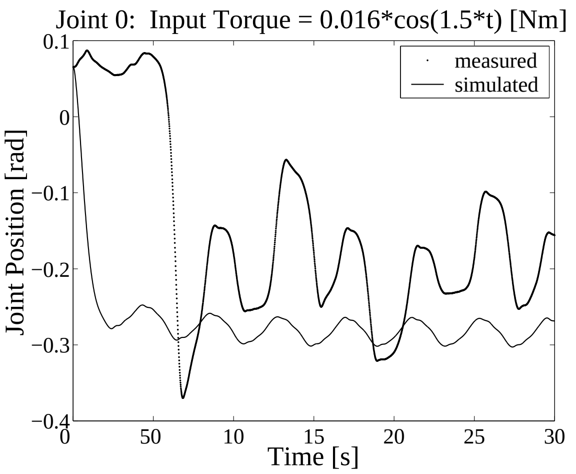


Figure 4.13: Positions of joint 0 with all joints free.

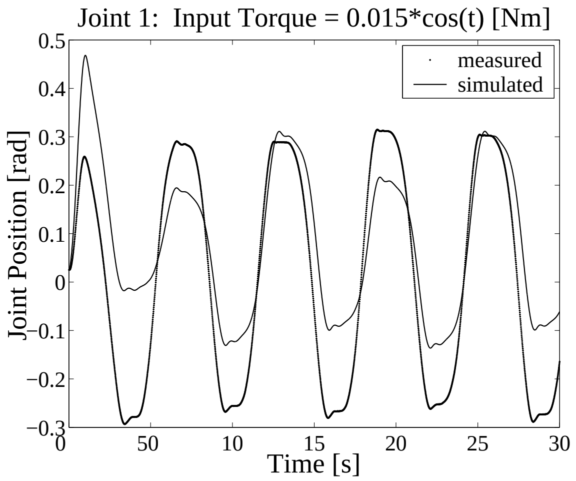


Figure 4.14: Positions of joint 1 with all joints free.

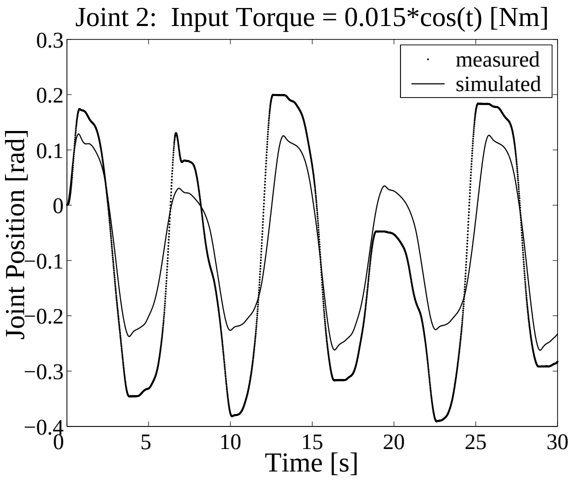


Figure 4.15: Positions of joint 2 with all joints free.

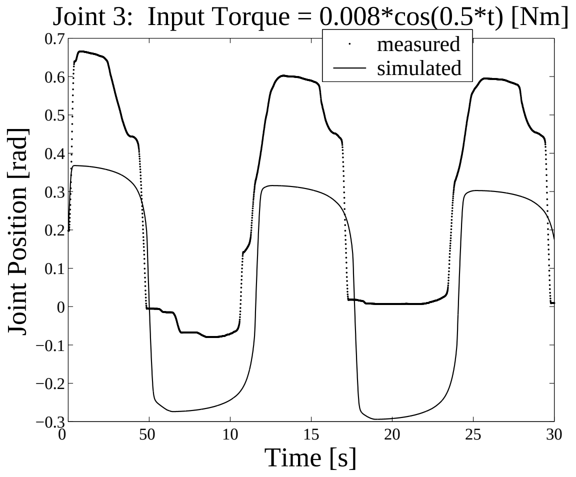


Figure 4.16: Positions of joint 3 with all joints free.

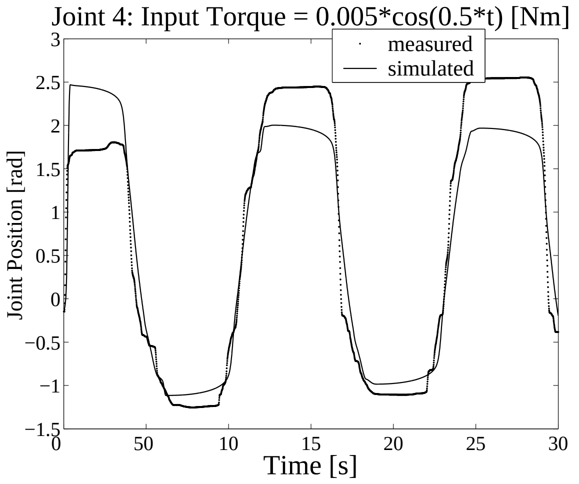


Figure 4.17: Positions of joint 4 with all joints free.

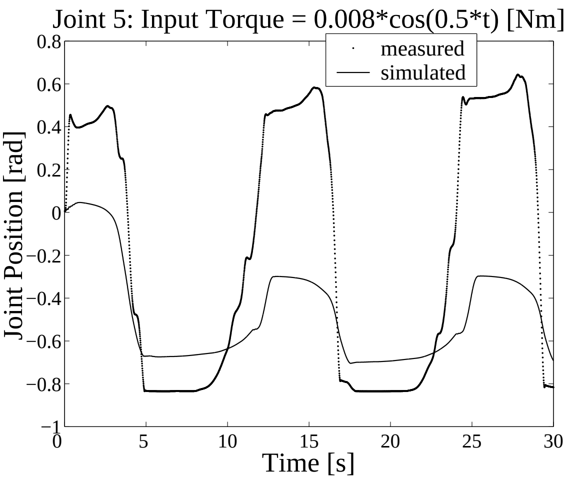


Figure 4.18: Positions of joint 5 with all joints free.

inaccuracies in the optimized parameters, particular for the parameters associating with the wrist joints. The dynamic model of the base joints are very sensitive to the parameters of the wrist joints because they change the reflected load drastically. As for joint 5, the difference in amplitudes between the measured and simulated results is a consequence of coupling effects of the tendons that are not accounted for during the isolated joint identification of joint 5.

4.3 System Identification by the Method of Coupling Joint

The second system identification method is to test the joints together in combinations. This is a good alternative to the first method which has a disadvantage of masking coupling affects. Specifically, coupling between joints through the tendon routing is not previously considered since the kinematics does not reflect such a scenario. When the isolated joint test is used, the coupling effects between joints are not apparent and subsequently the optimization might not be able to converge to values that give the best fit values of the coupling terms of the overall system. For Freedom 6S, the isolated joint test does not give a successful overall model of the 6 DOF device, and hence the method of coupling joint is used in an attempt to determining better model parameters. Since the form of the model is known from Chapter 3, this section outlines the steps to be executed in order to identify all unknowns in the model.

4.3.1 Locking Sequence

The locking sequence is used to identify the same set of parameters listed in Figure 4.2. However, instead of locking only one joint a time, the coupling joint method requires a locking sequence as listed in Table 4.6. The first test is to free only joints 4 and 5 while keeping all the other joints locked. The parameters determined from this first test are used in the second test which involves with leaving joints 3 to 5 free. The tests are continued in similar fashion by freeing one additional joint towards joint 0. By working from the wrist joints towards the base joints, the number of unknowns is minimized and the unlocking of more than one joint at any given time ensures that coupling effects are emphasized. The bracketed numbers from the table are the weighting factors used during the optimization routine and they will be explained shortly in Section 4.3.3.

Configuration	Joint0	Joint1	Joint2	Joint3	Joint4	Joint5
1	L	L	L	L	U (1)	U (1)
2	L	L	L	U (50)	U (1)	U (1)
3	L	L	U (50)	U (1)	U (1)	U (1)
4	L	U (50)	U (1)	U (1)	U (1)	U (1)
5	U (50)	U (1)	U (1)	U (1)	U (1)	U (1)

Table 4.6: The locking sequence of joints: L = Locked, U = Unlocked. The numbers in "()" are weighting factors used for optimization.

4.3.2 Additional Parameters

Identical to Section 4.1.2, friction and other effects have to be included with the Lagrangian unknowns to describe the entire systems.

4.3.3 Optimizing Locked Joints Parameters

Similar to Section 4.1.3, the same nonlinear optimization and initial conditions are used to determine parameters that are in best agreement with the measured data. Again it is done on the open-loop system to make friction more observable. To avoid transient effects, the cost function is the sum squared error between the measured data and the simulation data beyond the transient range. In addition, since the coupling joint tests have more than one unlocked joint at a time, the cost function consists of the sum squared error of all the free joints. To ensure that the optimization has the objective of finding parameters that are the closest match to the current joint of interest, heavier weighting would be placed on that one joint. The cost function is:

$$J = \Sigma_{j=m_1}^{m_2} \Sigma_{i=n_0}^{n} (\lambda_i \times (y_i - \hat{y}_i))^2) \tag{4.3}$$

where J is the cost function, $j = m_1,..,m_2$ is the unlocked joint index with m_1 being the first unlocked joint and m_2 being the last unlocked joint, $i = n_0,..,n$ is the time step index with n_0 being the first time step after the transient response (usually it is the 1001^{th} data point—approximately after 10 s), λ_i is the weighting factor on the joint of interest, y_i is the measured data, and $\hat{y}_i$ is the simulated data.

For configuration 1 in Table 4.6, j=4,5 and both λ_4 and λ_5 are equal to 1. This is because the objective of optimizing configuration 1 is to find parameters that best describe both joints 4 and 5. For configuration 2 in Table 4.6, j=3,4,5 and λ_4 and λ_5 both remain at the value 1. Joint 3 is the joint of interest for configuration 2 and so λ_3 is set to have a value of 50. This value is chosen based on trial and error and it ensures that the optimization has more emphasis on finding parameters that would best fit joint 3. The same weighting method is used for the rest of the configurations and it is listed in Table 4.6. The bracketed numbers indicate the weighting factors used on the specified joints for optimization. Note that there is no weighting factors on the locked joints because optimization is only done for free joints for each configuration.

To have rich signals for exciting the system, cosine chirp signals with varying amplitudes within the operating bandwidth are used as input signals. The signals with the respective joints are listed in Table 4.7. The ranges of the amplitudes and frequencies are chosen in ways such that the input torques would be large enough to maximize the coverage of the workspace but small enough so that they would not cause the linkages to hit hard stops when the joints are moved simultaneously. This is implemented by adding phase shifts on the input signals of joints 0 and 4.

Free Joint	Amplitude (A) (N*m)	Frequency (ω) (rad/s)	Phase Shift (ϕ)(rad)
J0	0.02	1+(2/60)*t	$\pi/2$
J1	0.008+(0.01/60)*t	1+(1/60)*t	0
J2	0.01+(0.01/60)*t	1+(0.5/60)*t	0
J3	0.008+(0.002/60)*t	0.5+(0.5/60)*t	0
J4	0.0045-(0.002/60)*t	0.5+(2.5/60)*t	$\pi/2$
J5	0.0055-(0.003/60)*t	0.5+(2/60)*t	0

Table 4.7: Input torque for coupled joint tests. They take the form of Torque = A*cos(ω*t+ϕ).

4.3.4 The Use of Beating Effect in Input Coupling Identification

Another advantage of doing system identification with coupling joints is that it can be used to identify any existing cable coupling between joints. For the Freedom 6S, the wrist joints are all tendon-driven and potential cable couplings between the joints must be investigated. Specifically, this refers to the routing for joints 4 and 5 since one tendon is used to transmit motion to both joints (see Freedom 6S User Manual [69] for routing details).

The first test according to the coupling joints locking sequence is to test joints 4 and 5 together. The input signals are chosen such that they have a slight offset in frequencies (i.e. joint 4 has a frequency increment of (2.5/60)*t while joint 5 has a frequency increment of (2/60)*t as listed in Table 4.7). If a beating phenomenon results from this offset in frequencies, it suggests that the tendons are providing additional couplings between the joints that are not previously included from the Lagrangian derivation.

When the coupling test between joints 4 and 5 is carried out, it can be observed that a beating effect is present. A beat is produced when two signals of slightly different frequencies are superimposed on top of each other [70]. The combined effect is a signal that oscillates with a beat frequency of:

$$\omega_{beat} = \omega_1 - \omega_2 \tag{4.4}$$

where ω_{beat} is the beat frequency, ω_1 and ω_2 are two frequencies that are slightly different from each other.

A response of joint 5 shown in Figure 4.19 resembles the beating phenomenon. To verify this hypothesis, two additional experiments are performed.

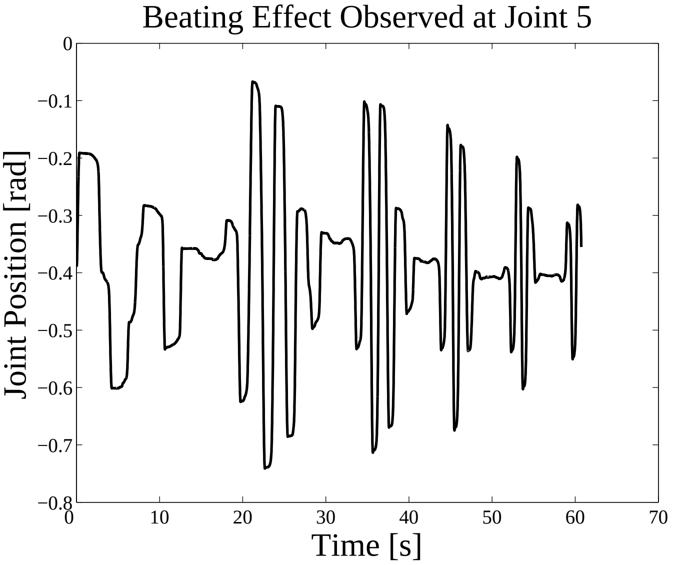


Figure 4.19: Beating effect observed at Joint 5.

4.3.4.1 Experiment: Input coupling in joints 4 and 5

For both experiments, all the joints remained locked except for joints 4 and 5. In the first experiment, a torque signal is sent to joint 4. Plots of joint position versus time for joints 4 and 5 are presented in Figure 4.20. Similar to the first experiment, the second experiment has an input torque to joint 5. The respective plot is presented in Figure 4.21.

From these results, it has shown that the tendons transmit input torques between joints 4 and 5 through mechanical coupling of the tendons. Also, by comparing

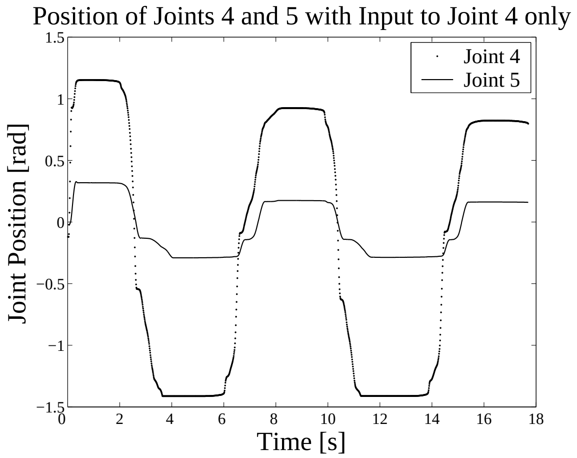


Figure 4.20: Effect of torque input to joint 4 on joint 5.

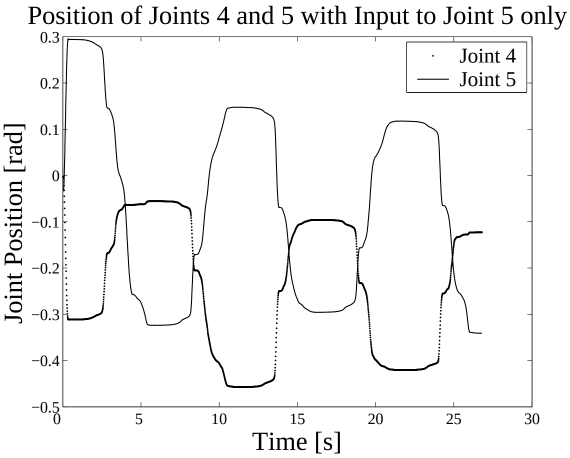


Figure 4.21: Effect of torque input to joint 5 on joint 4.

Figures 4.20 and 4.21, it appears that the input torque to joint 5 has a greater effect on the position of joint 4. In contrast, input signal to joint 4 has a much smaller effect on joint 5. In addition, an input torque to joint 5 would result in joint 4 moving 180 degrees out of phase as compared to joint 5. This has to do with the fact that the same tendon is used to drive both joints and the routing of tendons around the pulleys, as described by the MPB user manual [69], causes the phase shifts between the joints.

The result of this experiment demonstrates that there exists significant input coupling between joints 4 and 5. This is a mechanical coupling via tendons. This effect is included in the model by introducing two additional parameters to be optimized: *Gearing4To5* and *Gearing5To4*. They each represent the proportion of input torque of one joint to be fed to the other joint. With the phase shift in mind, the input to the model can be modified as follows. If the physical input torques to joints 4 and 5 are:

$$\tau_{q_4} = A_4 \times cos(\omega_4 \times t)$$

$$\tau_{q_5} = A_5 \times cos(\omega_5 \times t)$$

where τ_{q4} is the input torque to joint 4, A_4 is the amplitude of joint 4 in Newton-meter, ω_4 is the angular frequency of joint 4 in radians per second, t is the time in second. Similar notations are used for joint 5.

Then, the modified inputs to the model, denoted by τ'_{q4} and τ'_{q5}, to compensate for the physically coupling between joints 4 and 5 are:

$$\tau'_{q4} = \tau_{q4} - Gearing5To4 * \tau_{q5} \tag{4.5}$$

$$\tau'_{q5} = \tau_{q5} + Gearing4To5 * \tau_{q4} \tag{4.6}$$

The subtraction of the second term in Equation 4.5 accounts for the fact that a torque input to joint 5 would move joint 4 180 degrees out of phase. All experimental inputs to the model are modified using above descriptions.

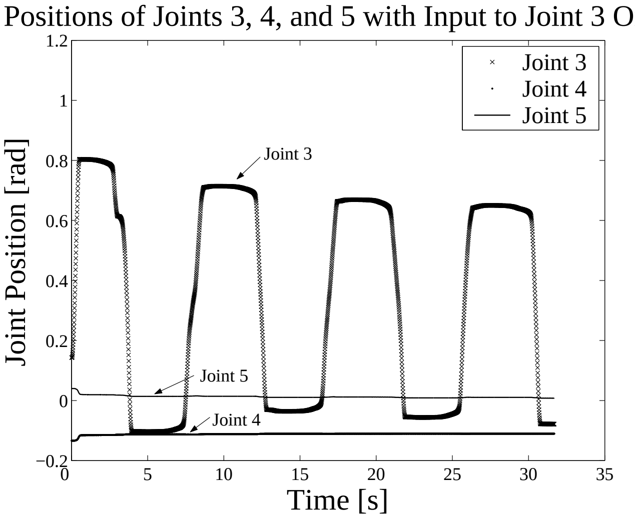


Figure 4.22: Effect of torque input to joint 3 on joints 4 and 5.

A similar input coupling experiment was done between joints 3, 4 and 5. A torque signal is sent to joint 3 only and no joint displacement from joints 4 and 5 are resulted from this input (Figure 4.22). Since joint 3 is driven by a separate tendon from joints 4 and 5, no input coupling is present.

4.3.5 Positional Bias on Friction

Depending on the mechanical system, considerations should also be given to positional-dependence on frictional characteristics. Two amplitudes are used to determine whether if a positional bias exists for a joint. One amplitude is the distance between the equilibrium position of the joint to the peak of the recorded joint position and the other is the distance between the equilibrium position to the valley of the recorded joint position. A position bias occurs when the two amplitudes do not have the same magnitude. The positional bias becomes obvious when experiments are done on the base joints. Hence, a positional bias term, which is a factor weighting the influence of friction depending on whether the joint position is greater or less than 0 degrees, is implemented in the model for the base joints. This factor is an additional parameter for each joint required to be solved by the optimization routine outlined in Section 4.3.3. Positional bias on friction is implemented in the model after the LuGre friction has been calculated. If the joint position at that time step is less than the equilibrium position of that joint, then the new friction value is calculated by multiplying the LuGre friction value by a"friction bias factor" (see Appendix D for the numerical values). As a result, the LuGre friction value would describe the friction value when joint position is larger than the joint equilibrium position and a scaled friction value is implemented when the joint position is smaller than the joint equilibrium position.

4.4 Results from using the Method of Coupling Joints

Using the method of coupling joints, optimization is done on data sets for each joint as described in Section 4.3. Again, nonlinear optimization method (see Section 2.4) is used to determine the most suitable parameter values that can minimize the sum squared error between the measured and the simulated results. For each test, a set of cosine chirp input torques is applied in sequence to the joints of interest (i.e. the unlock joints) as listed in Table 4.6. The resulting data sets are used to evaluate the cost function for optimizing the parameters of each joint. Table 4.7 lists the input torques given to the device for generating measured data used for optimization. A

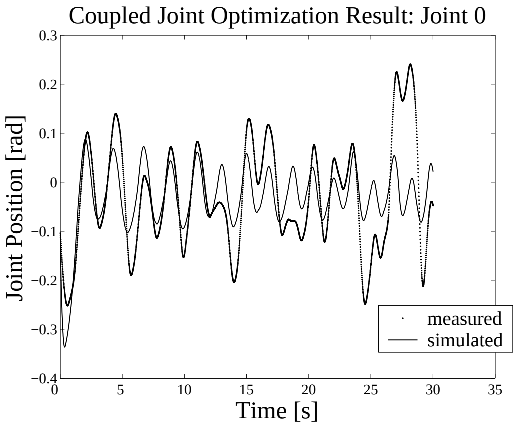


Figure 4.23: Positions of joint 0 using the coupling joint method.

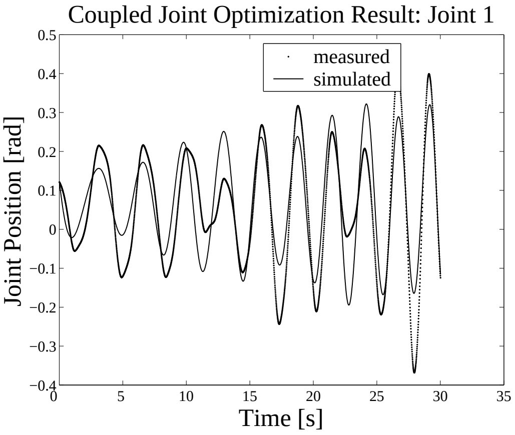


Figure 4.24: Positions of joint 1 using the coupling joint method.

fixed time step of 0.1667 ms (=0.01/60 s) and the ODE4 Runge-Kutta solver are used for the simulation of joints 0 to 5. Recall from Section 4.2, a fixed time step of 0.5 ms was used for simulations on isolated joint. A smaller time step of is required for the coupling joints simulations because the dynamic responses from the coupled joints are more sensitive. Hence, it requires a higher resolution to accurately capture the response. By trial and error, the time step of 1/60-th of the sampling time of the ADC (0.01s) is found to be a sufficient resolution for the model to simulate properly.

The results obtained from simulation using the optimized parameters and the measured data for each joint are presented in Figures 4.23 to 4.28. According to the locking sequence of coupling joints test (Table 4.6), only the result(s) of the joint(s) being optimized are presented. From Table 4.6, configuration 1 allows for optimization done on joints 4 and 5. Hence, the results presented below for joints 4 and 5 come from the optimization done on configuration 1. Similarly, configuration 2 allows for optimization done on joint 3. Hence, the result presented below for joint 3 comes from the optimization done on configuration 2. The simulation time is chosen such that a relevant number of cycles can be displayed.

In general, the simulation provides acceptable matches to the measured data. The simulated joint positions of the base joints (Figures 4.23 to 4.25) show results that are in agreement with the trends of the measured data. The differences can

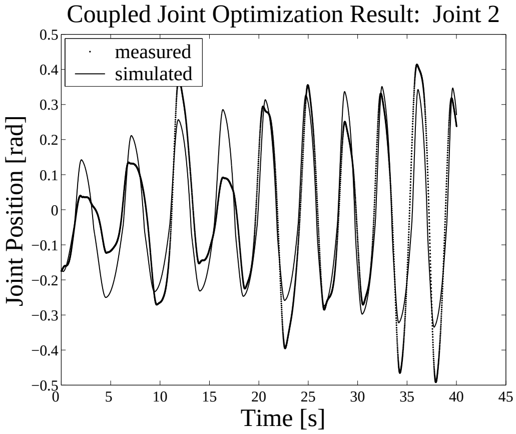


Figure 4.25: Positions of joint 2 using the coupling joint method.

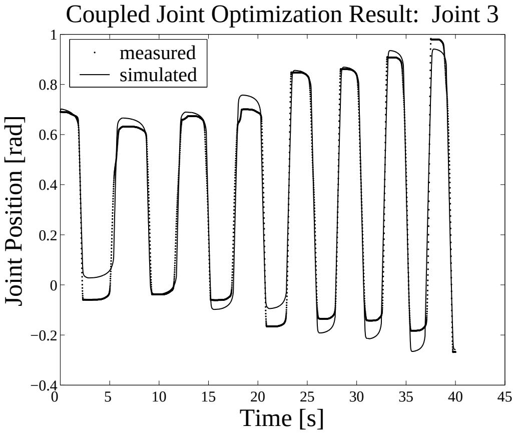


Figure 4.26: Positions of joint 3 using the coupling joint method.

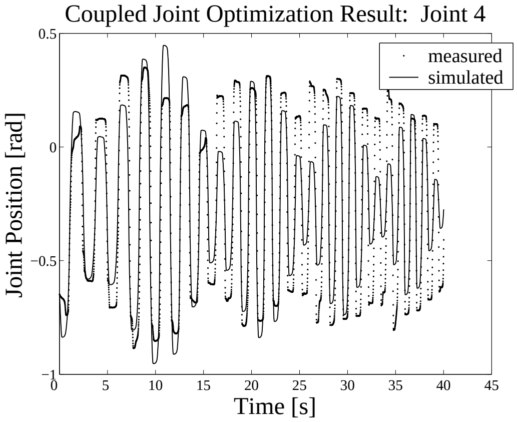


Figure 4.27: Positions of joint 4 using the coupling joint method.

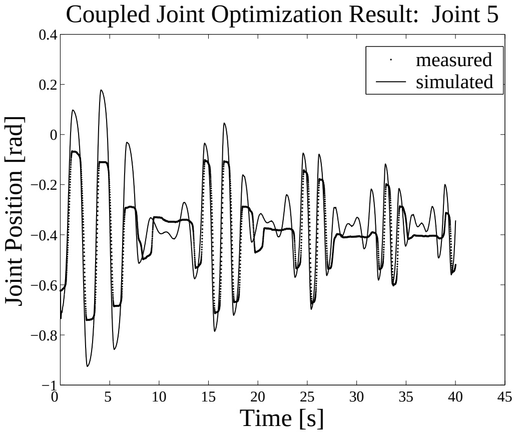


Figure 4.28: Positions of joint 5 using the coupling joint method.

be attributed to the accumulated errors from optimizations done on the previous joints. This is most apparent in the result of joint 0 since the optimization of joint 0 is based on all the optimized parameters of the previous joints. Despite the lack of agreement in terms of amplitude, the simulated result shows similar trends to that of the measured result. The simulation results of joints 1 and 2 also show acceptable similarity in trends and amplitudes compared to the measured data sets. Since all the measurements are done as an open-loop system, it is speculated that the model is able to describe the system even better in a closed-loop system.

Figures 4.26 to 4.28 show the simulated joint positions of the wrist joints. Again, the LuGre friction model is able to produce stiction effects that agree with experiment. Since the number of peaks over a given time are the same from the figures, the optimized parameters produce simulation results that are able to match the frequency of the measured data adequately with acceptable variations in amplitudes. By introducing coupled input torques between joints 4 and 5 (described in Section 4.3.4), beating phenomenons are correctly simulated to match those observed in measured data.

As all the identifications are done with an open-loop system, a closed-loop system should provide a better match between the actual system and the model. The closed-loop experiments described in the next section is to further check the validity of the model obtained from the method of coupling joints.

4.5 Closed-Loop Verification

The open-loop optimization result obtained from using the method of coupling joints are further verified with a closed-loop test. Two sets of proportional-derivative (PD) gains are chosen such that they reduce the error between the command and the measured signals. During the tuning of the PD gains, it becomes apparent that it is necessary to filter the error signal derivative ($\frac{de}{dt}$) because it fluctuates greatly near the beginning of the command signal. This large magnitude of the error derivative gives rise to a large input torque that would cause instability in the device. Furthermore, the roll joint (joint 4) is designed such that its joint position is calculated from two sensors and a large torque often causes this joint to over rotate. When over rotation occurs on joint 4, it causes the sensors to send error messages.

Due to these problems, a second order low-pass filter is implemented to smooth out the initial spikes observed in the error derivative to avoid instability and stalling of sensors.

4.5.1 Filter Implementation

The second order low-pass filter is chosen such that it has the lowest cut-off frequency without causing a significant lag in response. This frequency is determined by implementing the filter at a selected frequency with an arbitrary set of PD gains along with a step input. If the response appears to have too much lag, a higher frequency filter will be used. On the other hand, if the chosen frequency causes instability of the system, a lower cutoff frequency is used. From trial and error, 15 Hz is chosen to be the cutoff frequency for the filter for all joints. The filter has the difference equation form of:

$$y(k) \quad = \quad \frac{1}{a1} \times (-a2 \times y(k-1) - a3 \times y(k-2) + b1 \times u(k-1) + b2 \times u(k-2)) \tag{4.7}$$

where $y(k)$ is the filtered $\frac{de}{dt}$ at the current time k, $y(k-1)$ is the filtered $\frac{de}{dt}$ at the previous time step $k-1$, $y(k-2)$ is the filtered $\frac{de}{dt}$ at two previous time steps $k-2$, $u(k-1)$ is the pre-filtered $\frac{de}{dt}$ at the previous time step $k-1$, $u(k-1)$ is the pre-filtered $\frac{de}{dt}$ at two previous time steps $k-2$, and the constants for a 15 Hz cutoff frequency are $a1$ = 1, $a2$ = -0.8073, $a3$ = 0.2638, $b1$ = 0.2789, $b2$ = 0.1775.

4.5.2 PD Gains

After the filter is implemented, tuning can be done to find an appropriate sets of PD gains. Two sets of PD gains are desired to demonstrate the effect of increasing gains on the resulting trends of the measured and simulated data. By trail and error, the smaller set of gains are chosen to give a good step response (i.e. a response with minimum overshoot and oscillations) while the higher set is approximately twice as high as the smaller sets. The PD gains used for each joint are presented in Table 4.8.

Joint	P (small)	D (small)	P (big)	D (big)
J0	0.8	0.04	1.2	0.08
J1	0.3	0.04	0.5	0.08
J2	0.3	0.015	0.5	0.03
J3	0.04	0.0015	0.08	0.003
J4	0.006	0.0000025	0.01	0.000005
J5	0.04	0.0015	0.08	0.003

Table 4.8: PD gains used for closed-loop verification.

4.5.3 Closed-Loop Verification Results

The inputs used for closed-loop verification are still chirp signals with the same frequencies as those used in Section 4.3.3 for open-loop experiments. However, the closed-loop input signals (i.e. joint position command) have different amplitudes than the ones used for open-loop (i.e. motor torque command). The closed-loop input signals are listed in Table 4.9.

Joint	Amplitude (A(t)) (rad)	Frequency ($\omega(t)$) (rad/s)	Phase Shift (ϕ)(rad)
J0	0.2-(0.2/60)*t	1+(2/60)*t	$\pi/2$
J1	0.1+(0.3/60)*t	1+(1/60)*t	0
J2	0.1+(0.1/60)*t	1+(0.5/60)*t	0
J3	0.4+(0.3/60)*t	0.5+(0.5/60)*t	0
J4	0.5-(0.3/60)*t	0.5+(2.5/60)*t	$\pi/2$
J5	0.4-(0.2/60)*t	0.5+(2/60)*t	0

Table 4.9: Input position command for closed-loop verification test. They take the form of Torque = A(t)*cos($\omega(t)$*t+ϕ).

Two sets of measured data are made with the small and high sets of PD gains. The input position commands are sent to all the joints simultaneously and they are compared with the model simulation. The results are grouped by the PD gains used.

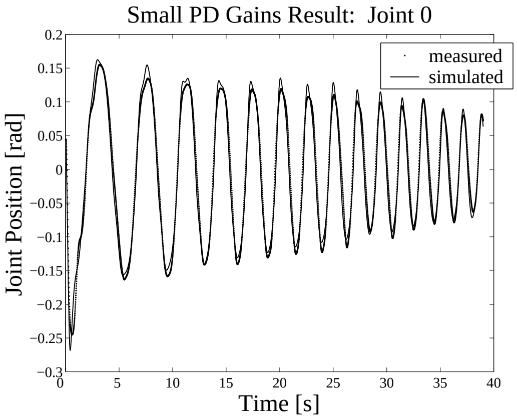


Figure 4.29: Positions of joint 0 using small PD gains.

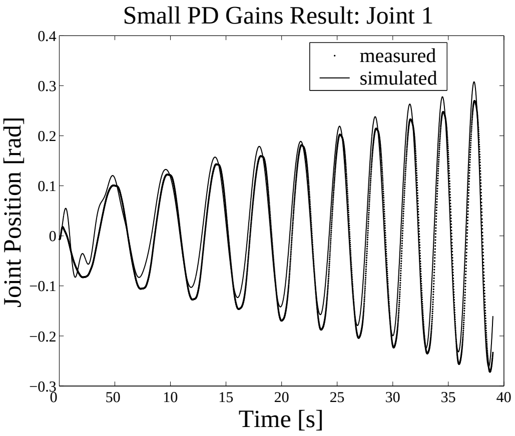


Figure 4.30: Positions of joint 1 using small PD gains.

4.5.3.1 Small PD Gains Verification

Figures 4.29 to 4.34 show the results for the small PD gains.

The closed-loop model matches very closely for joints 0 to 3 with small PD gains. There seems to be some discrepancies in amplitude between the model and measured data for joint 4 but it is suspected to be related to frictional effects of the wrist joint. Previously determined friction parameters could have changed as the tendons came loose and needed to be rerouted to proceed with more testing. As for joint 5, the difference between the model and the measured data is unacceptable. From observing the measured data of joint 5, it seems that it is different from the past measurement in that it exhibits no beating phenomenon when joints 4 and 5 are operated simultaneously. This discrepancy in measured data is attributed to the fact that the wrist tendons had been rerouted a couple of times since the last measurements are made for the coupling joint experiment. By rerouting the wrist joint, the friction characteristics are different since all the tendons are wound around the pulleys differently than before. It is suspected that when the tendons are wound up properly without much overlapping, the behaviour of joint 5 is independent of the input of joint 4. This is confirmed by observing joint 5 when joint 4 is moved physically by hand. Hence, the previous model must be modified to describe the correct tendon routing for joint 5. Joint 5 is only influenced by its own input signal and it is not affected by that of joint 4. The torque input to joint 5 described by

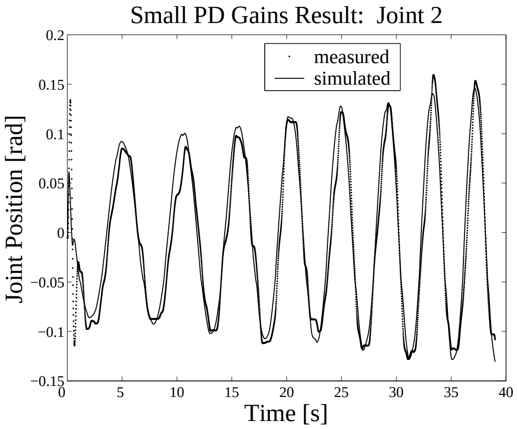


Figure 4.31: Positions of joint 2 using small PD gains.

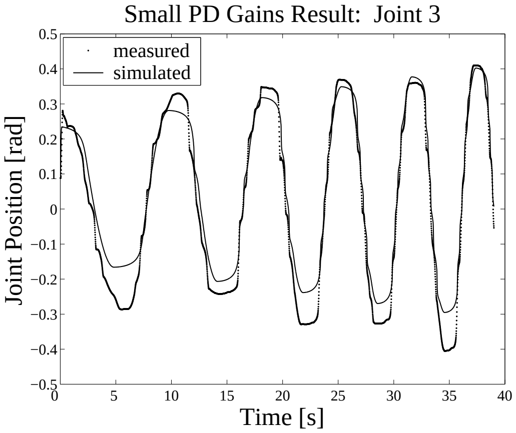


Figure 4.32: Positions of joint 3 using small PD gains.

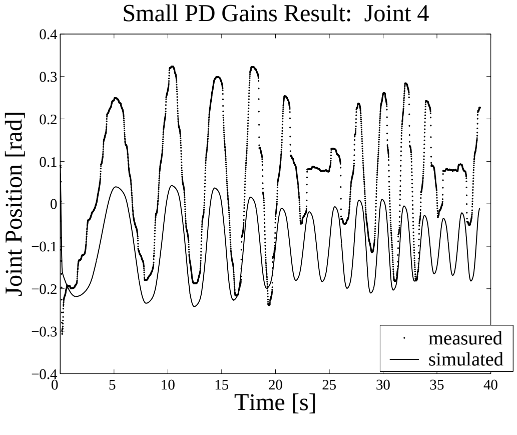


Figure 4.33: Positions of joint 4 using small PD gains.

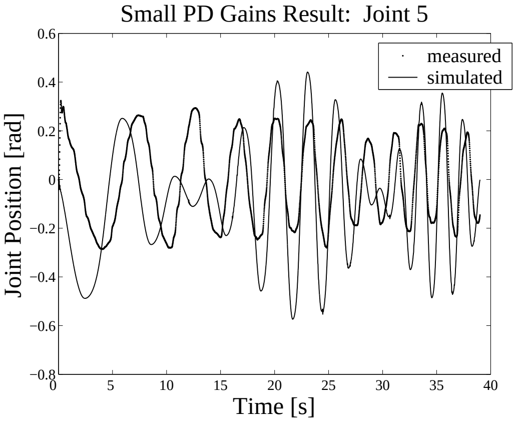


Figure 4.34: Positions of joint 5 using small PD gains.

Equation 4.6 is modified to:

$$InputTorqueToJoint5 = A5 \times cos(w5 \times t)$$

With this modification, a new simulation is run and a great improvement has resulted from this modification on model of joint 5. Since the results from joints 0 to 4 remain relatively similar to the previous figures, only the result of joint 5 is presented in Figure 4.35. It is confirmed, with this new result, that the movement of joint 5 is independent of the input of joint 4. Again, the difference in magnitude is attributed to the fact that the previously determined friction parameters are not exact since the tendons have been rerouted.

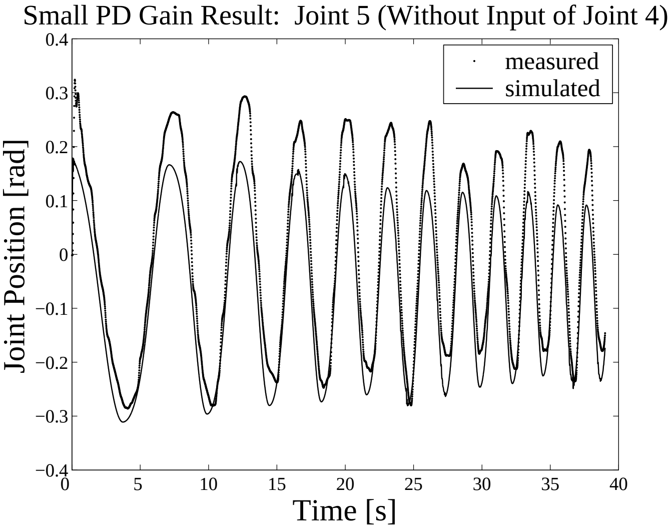


Figure 4.35: Measured and simulated joint positions of joint 5 using small PD gains with input from joint 4.

4.5.3.2 High PD Gains Verification

By doubling the small PD gains and modifying the model of joint 5, the results of using the high PD gains are presented in Figures 4.36 to 4.41.

With a higher set of PD gains, the model shows excellent agreement for measured data of joints 0 to 3 and 5. For joint 4, the model is able to provide a better match

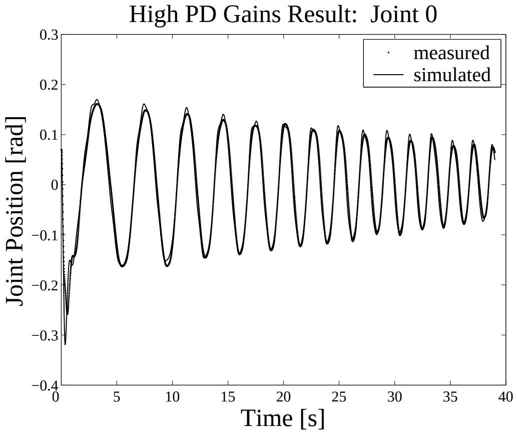


Figure 4.36: Positions of joint 0 using high PD gains.

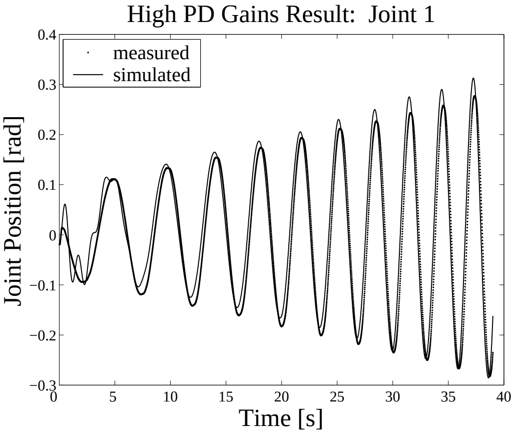


Figure 4.37: Positions of joint 1 using high PD gains.

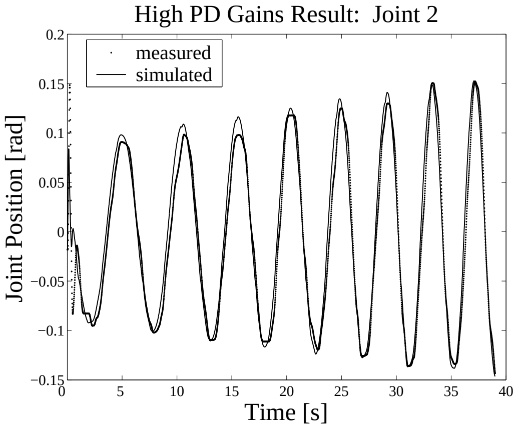


Figure 4.38: Positions of joint 2 using high PD gains.

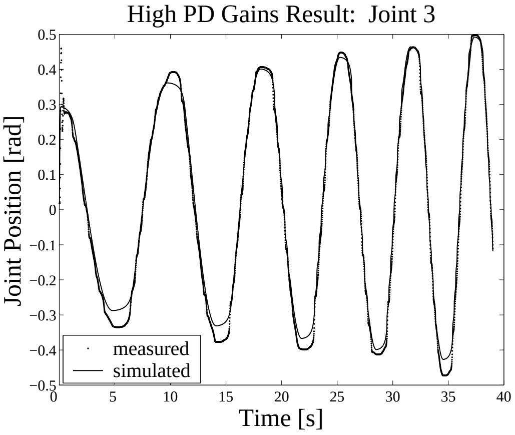


Figure 4.39: Positions of joint 3 using high PD gains.

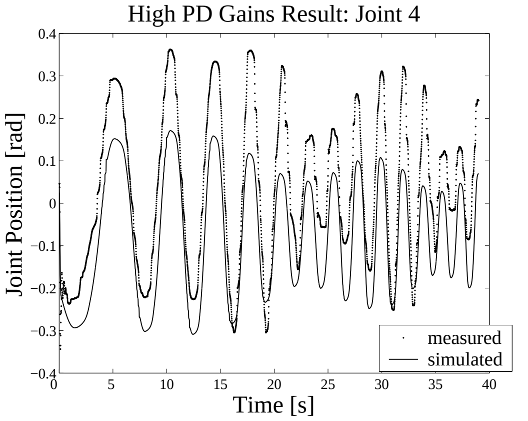


Figure 4.40: Positions of joint 4 using high PD gains.

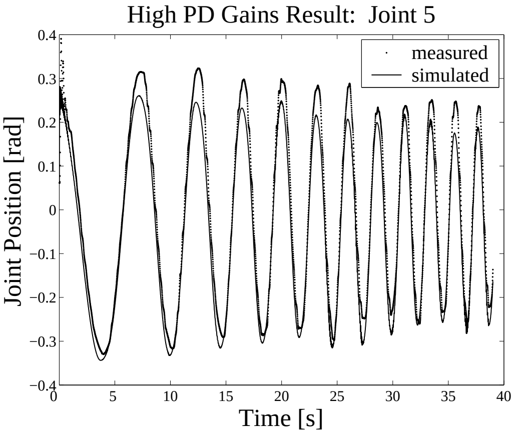


Figure 4.41: Positions of joint 5 using high PD gains.

for the large PD gains measured data than the ones with small PD gains. This does not mean that the friction model is poor as the controller is not designed to compensate for frictional effects. In general the large set of PD gains provide input torque that is large enough to overcome higher amount of friction and hence, the measured results are in a superior agreement with the model.

4.6 Chapter Summary

This chapter completes the modelling of Freedom 6S and the importance of the proposed methodologies and experimental results are summarized below.

4.6.1 System Identification Methods

In this chapter, two methods of system identification are presented and both have the purpose of reducing the number of parameters to be identified in a complex dynamic model. Knowing an analytic form of the dynamic model, one is able to isolate a subset of the system describing selected joints and the associated unknowns. This way, tests can be tailored to capture the dynamic response of the selected joints and the model unknowns can be determined in a modular fashion. These systematic approaches to system identification can be applied to nonlinear dynamic systems provided the joints can be physically constrained. The method of isolated joint allows for the least number of parameters to be identified at a time at the expense of neglecting input coupling effects. The method of coupling joint, however, is able to capture the coupling effects.

4.6.2 Experimental Results from System Identification

The system identification results using the method of isolated joint produces optimized parameters that generate simulations which are in excellent agreement with the measured data for the individual joints. However, when the last test of the locking sequence is carried out, the results show that there are inaccuracies in the previously determined parameters. As soon as two joints, joints 0 and 1, are set free, the model parameters can no longer generate a simulation of joint 0 that would match the measured data. Verification using the 6 DOF model confirms the inaccuracies of the parameters.

The method of isolated joint allows the identification of non-coupled joint parameters to be done readily. On the other hand, it does not account for input coupling behaviour. Based on this reasoning, another system identification method is proposed in an attempt to generate a model better capturing the entire system.

The alternative method, the method of coupling joints, provides a better overall system identification technique than that of the isolated joint. While isolated joint allows for minimal number of parameters to be identified at one time, the coupling joints method offers the opportunity to optimize coupled terms at the same time as non-coupled terms to ensure their importance is weighed fairly.

The verification of the model using a closed-loop system proves that it can adequately describe the dynamic behaviour of the Freedom 6S. With high unpredictability in the friction caused by the tendons, it is advisable to reroute or check the tendon routing frequently to ensure there is no entanglement of strings around the pulleys. The correct routing procedure is outlined in the MPB user manual [69].

Chapter 5

Control of Vibration during Virtual Wall Contact

5.1 Virtual Wall Contact

This work specifically studies the haptic effects of virtual wall contact. It is a known phenomenon, as discussed in Section 2.5, that it is difficult to create a solid wall in a virtual environment without vibration effects. An insufficient sampling rate can lead to instability when the user penetrates within a close range to the wall boundary. With a slow sampling rate, it is possible to have forces applied to the user even if the current position of the user is outside of the wall boundary. Inaccuracy in position feedback can lead to instability of the system. It is common for the user to feel a chattering sensation due to a slow sampling rate when they push against the wall boundary.

The goal of this chapter is to reduce the chattering upon the penetration of a virtual wall using a fuzzy logic approach. Starting with a traditional spring and damper system to describe the contact dynamics of a virtual wall, the dynamics are fuzzified and then fuzzy logic is applied to the spring constant (proportional gain) and damping (derivative gain) coefficients. A comparison in performance between using a traditional PD method and the fuzzy logic method is investigated using the Freedom 6S.

5.2 Test Setup for Virtual Wall Experiment

To investigate the validity of using a fuzzy controller in place of a linear proportional-derivative gain (PD) controller when implementing a virtual wall, a set of simple experiments are conducted using the Freedom 6S. Only the first joint is tested and the other five joints are physically locked in place so no relative motions occur. By reducing the number of DOF of Freedom 6S, the fuzzy logic implementation can be demonstrated without introducing the unnecessary complexity in dynamics resulting from a device with higher DOF.

A virtual wall is located at joint position equal to 0 radian. Penetration of the wall happens when the joint is moved to less than 0 radian. For each test, the user holds on to the end effector and moves the locked arm from free space (i.e. joint 0 position is larger than 0 radian) towards the wall at various approach speeds. The position and velocity of joint 0 are recorded for each test to indicate the level of chattering upon wall contact. A schematic is shown in Figure 5.1 describing the equipment experimental setup. The wall is modelled as a set of spring and damper in series. The determination of the spring stiffness (i.e. wall stiffness, P) and the damping coefficient (i.e. D) is described in Section 5.3 for using a PD controller approach. In terms of the fuzzy approach, the two parameters are fuzzified and the methodology used to determine these parameters are described in Section 5.5.

For both approaches, switching of gains is required when the user moves the link in and out of the virtual wall located at 0 radian. Torque, τ, is only computed from the controller (i.e. the PD controller or the fuzzified PD controller) when the joint position, q, is smaller than 0 radians. A block diagram in Figure 5.2 outlines the control strategy used to simulate a virtual wall.

5.3 Virtual Wall Parameters Modelled by a PD Controller

It is desirable to choose virtual wall parameters (i.e. P and D) to provide a solid feeling wall. These parameters can be viewed as PD controller gains as well as wall stiffness and damping. For a set of proportional and derivative (PD) gains with large magnitudes, a lot of vibration results when the manipulator is making contact

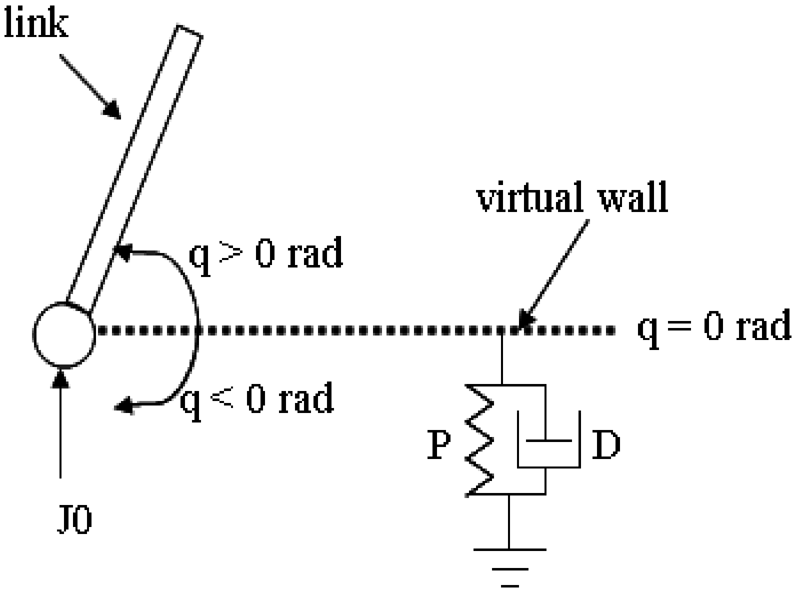


Figure 5.1: A schematic showing the virtual wall location with respect to the position of joint 0.

with the virtual wall. The linear control law which computes the wall contact force to be exerted onto the user is $u = k_1 * q + k_2 * \dot{q}$, where u is the torque, k_1 is the proportional gain, q is the joint position, k_2 is the derivative gain, and $\dot{q}$ is the joint velocity. k_1 and k_2 are chosen to be 100 and 100 respectively. This PD combination is good for demonstrating chattering phenomenon observed from virtual wall contact. The results from using this set of PD gains are to be compared with the results generated from the fuzzy parameters in the latter sections.

5.4 Deterministic Method in Fuzzy Control

The following sections discuss the development work that leads to the fuzzy controller design. The background assumes the reader has a basic level of understanding of fuzzy logic from Section 2.6.2.

In the work of Kubica [60], a method of mapping linear controllers into fuzzy controllers is developed. This is advantageous for fuzzy controller design because if the linear controller is known, the mapping provides an exact equivalence fuzzy

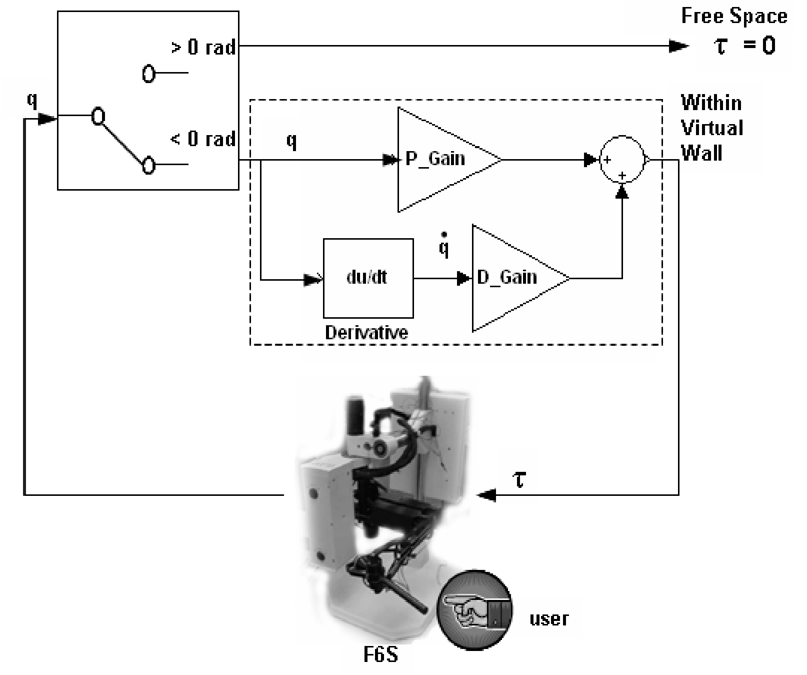


Figure 5.2: A block diagram showing the simulation of a virtual wall using PD parameters.

controller which is a good starting point from which fuzzy reasoning can be added to customize the design to achieve stability. Using a PD controller as a starting point, this section outlines the steps required to transform the linear controller into the fuzzy domain as done in [60].

Step 1: Creating membership functions from a linear controller

Assuming a system with two inputs and one output and the given linear controller takes the form of

$$\mathbf{y} = \mathbf{k}\mathbf{z} + \mathbf{c}$$

where $\boldsymbol{k}$ is the 1x2 gain matrix, $\boldsymbol{z}$ is the 2x1 state vector (e.g. position, velocity), $\boldsymbol{c}$ is the 2x1 offset vector.

The first step to the fuzzy mapping is to construct input memberships. However, for this mapping to be done correctly, there are three conditions which must be met when constructing the input membership functions.

1. Membership functions must be triangular in shape.
2. Fuzzy sets must be constructed such that the sum of the membership functions at a given input value must add up to 1. More specifically in mathematical terms:

   Given the two fuzzy sets A_i^0 and A_i^1 with the respective membership functions $\mu_{A_i^0}$ and $\mu_{A_i^1}$,

   $$\mu_{A_i^0}(z_i) + \mu_{A_i^1}(z_i) = 1 \tag{5.1}$$

   where A_i^0 refers to the 0^{th} fuzzy set of input i; A_i^1 refers to the 1^{st} fuzzy set of input i; $\mu_{A_i^0}$ is the membership function of the fuzzy set, A_i^0; $\mu_{A_i^1}$ is the membership function of the fuzzy set, A_i^1; i is the subscript referring the input (i.e. i = 1,2 for two inputs) ; z_i is the crisp input value.

   The domain of z_i is also constrained such that $\bar{A}_i^0 \leq z_i \leq \bar{A}_i^1$ where $\mu_{A_i^\rho}(\bar{A}_i^\rho)$=1 and $\rho = 0, 1$. In another word, the crisp input must be between the centroids of two consecutive fuzzy sets where each centroid has a membership function of value 1.
3. Each input has non-zero membership grades in only two fuzzy sets, A_i^0 and A_i^1 for i=1,2. This means that there can only be at most 2 fuzzy sets both contributing non-zero membership function values for each input.

When these three conditions are met, then linear interpolation can be used to find the membership function value of a given input. Note the relationship to Equation 5.1 is also included in the below equation.

$$\mu_{A_i^0}(z_i) = \frac{\bar{A}_i^1 - z_i}{\bar{A}_i^1 - \bar{A}_i^0} = 1 - \mu_{A_i^1}(z_i) \tag{5.2}$$

Step 2: Fuzzy Associative Memory

The Fuzzy Associative Memory (FAM) is the rule-base used to determine the fuzzy outputs given the fuzzy inputs. For this example of a 2 input and 1 output system, the FAM can be organized in the table as follows:

Each combination of fuzzy sets from each input is associated with a rule, noted as B^k, where k is the output rule number starting from 0. Table 5.1 shows the general structure of the FAM.

Step 3: Output Membership Functions

For the mapping to work correctly, output membership functions have to be fuzzy singletons located at $\bar{B}^0, ..., \bar{B}^{2n-1}$ for sets $B^0, ..., B^{2n-1}$, where n is the number of inputs. For the case of two inputs, one output then:

$$\begin{aligned} \bar{B}^0 &= k_1\bar{A}_1^0 + k_2\bar{A}_2^0 + c \\ \bar{B}^1 &= k_1\bar{A}_1^0 + k_2\bar{A}_2^1 + c \\ \bar{B}^2 &= k_1\bar{A}_1^1 + k_2\bar{A}_2^0 + c \\ \bar{B}^3 &= k_1\bar{A}_1^1 + k_2\bar{A}_2^1 + c \end{aligned} \tag{5.3}$$

Calculating using the control gains, k_1 and k_2, the fuzzy singletons are related to the fuzzy sets in the order presented in the FAM table.

Step 4: Defuzzification

To change from the fuzzy domain to a crisp output value, the centroidal defuzzification process is required. For the example of two inputs and one output:

$$y_0 = \frac{\bar{B}^0\mu_{A_1^0}(z_1)\mu_{A_2^0}(z_2) + \bar{B}^1\mu_{A_1^0}(z_1)\mu_{A_2^1}(z_2) + \bar{B}^2\mu_{A_1^1}(z_1)\mu_{A_2^0}(z_2) + \bar{B}^3\mu_{A_1^1}(z_1)\mu_{A_2^1}(z_2)}{\mu_{A_1^0}(z_1)\mu_{A_2^0}(z_2) + \mu_{A_1^0}(z_1)\mu_{A_2^1}(z_2) + \mu_{A_1^1}(z_1)\mu_{A_2^0}(z_2) + \mu_{A_1^1}(z_1)\mu_{A_2^1}(z_2)}$$

Using the constraints from Equation 5.1, the denominator simplifies to 1. Then, the output calculation simplifies to:

	Input 1	
Input 2	A_1^0	A_1^1
A_2^0	B^0	B^2
A_2^1	B^1	B^3

Table 5.1: An example of a FAM.

$$y_0 = \bar{B}^3 + (\bar{B}^1 - \bar{B}^3)\mu_{A_1^0}(z_1) + (\bar{B}^2 - \bar{B}^3)\mu_{A_2^0}(z_2) \tag{5.4}$$

Following steps 1-4 described above, an equivalent fuzzy controller is produced given the linear controller. This can be confirmed by substituting Equations 5.2 and 5.3 into Equation 5.4:

$$\begin{aligned} y_0 &= k_1\bar{A}_1^1 + k_2\bar{A}_2^2 + c + k_1(\bar{A}_1^0 - \bar{A}_1^1)(\frac{\bar{A}_1^1 - z_1}{\bar{A}_1^1 - \bar{A}_1^0}) + k_2(\bar{A}_2^0 - \bar{A}_2^1)(\frac{\bar{A}_2^2 - z_2}{\bar{A}_2^1 - \bar{A}_2^0}) \\ &= k_1 z_1 + k_2 z_2 + c \\ &= \mathbf{kz} + c \end{aligned}$$

The substitutions indeed simplify to Equation 5.1 as shown.

5.5 Design of Fuzzy Controller

As outlined in Section 2.6.2 , the overall structure of the fuzzy controller can be described by three stages: 1) fuzzifying the inputs; 2) inferencing based on a fuzzy associative memory relating inputs to outputs; 3) defuzzifying the output. For our virtual wall application, the initial design of the fuzzy controller is constructed such that it has an equivalent performance as the linear PD controller described in Section 5.3. Afterwards, logical reasoning can be applied to construct the fuzzy associative memory (FAM) by modifying the initial design. In order to map the linear PD controller into an equivalent controller in the fuzzy domain, the membership functions must satisfy the set of constrains described in Section 5.4.

5.5.1 Input Membership Functions

For the one DOF Freedom 6S application, the two inputs, joint, q, position and joint velocity, dq, can be represented by two groups of membership functions. The three conditions outlined in Section 5.4 are met to ensure that the characteristics of the PD controller are preserved in the fuzzy domain. In order to determine the number of membership functions necessary, some initial testing using the PD gains are performed. From the recorded joint position, a range in which a lot of chattering occurs (i.e. the range of interest) would be allocated with a denser number of membership functions. Similar logic is applied to the joint velocity and the respective range of interest can be defined. Table 5.2 summarizes the universe of discourse, the range of interest, and the number of membership functions for each range. The number of membership functions listed in Table 5.2 is an initial estimate. If the result does not give a smooth transition within the workspace, more membership functions can be added.

Input	Universe of Discourse	Range of Interest
Joint Position	[-0.55851 0.69813] rad (13 MFs)	[-25e-3 5e-3] rad (5 MFs)
Joint Velocity	[-10 10] rad/s (13 MFs)	[-5 5] rad/s (7 MFs)

Table 5.2: The fuzzy domain for the two inputs and their respective number of membership functions. Note: "MFs" stands for membership functions.

The membership functions are presented graphically in Figures 5.3 to 5.5. Note that the 13 membership functions describing the universe of discourse for joint position (Figure 5.3) include the 5 densely-spaced membership functions within the range of interest. For joint velocity, the range of interest is described by 7 membership functions with a total of 13 membership functions describing the universe of discourse (Figure 5.5). For both the position and velocity, the mid-sets center at zero. Figure 5.4 is a zoom-in figure of Figure 5.3. Similarly, there are more densely-packed membership functions describing the range of interest in terms of joint velocity in Figure 5.5.

A specific set of naming conventions is used to name input and output membership functions. For input membership functions, the naming convention can be seen as labels on Figures 5.3 to 5.5. N^i, Z, and P^i refer to negative, close to zero and positive values respectively and i refers to the i^{th} fuzzy set. The subscript associated with the naming convention, as indicated on Figures 5.3 to 5.5, refers to the input that the membership is associated with (e.g. N_q^i describes negative values for joint position, P_{dq}^i describes positive values for joint velocity). If no subscript is present, then that membership function is referred to an output membership function (e.g. N^i alone describes negative values for output torque, see Figure 5.6).

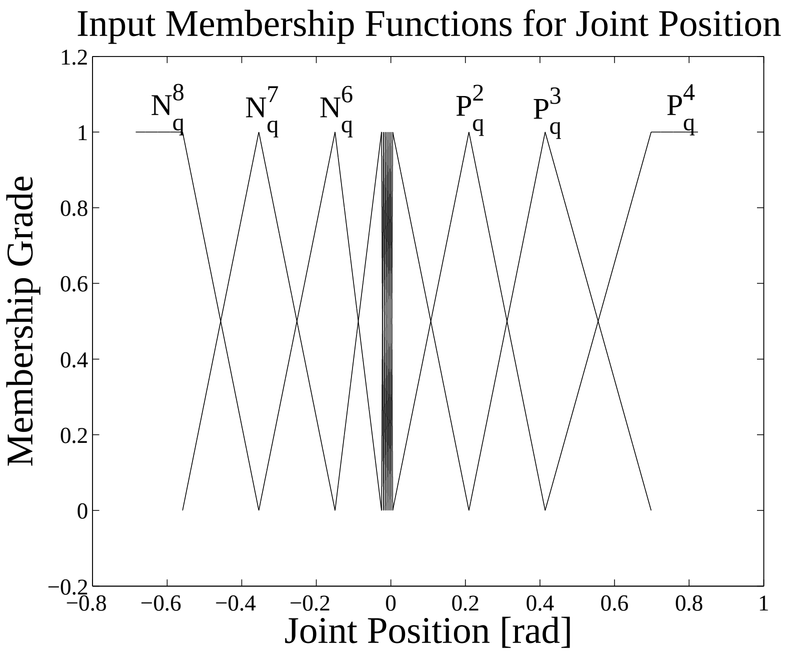


Figure 5.3: Input membership functions of joint position over the universe of discourse.

5.5.2 Fuzzy Associative Memory (FAM) and Output Membership Functions

The general concept of FAM is described in Section 2.6.2. FAM can be viewed as a rule-based of the form:

$$IF\ q\ is\ A\ AND\ \dot{q}\ is\ B\ THEN\ \tau\ is\ C.$$

where q is the joint position, A refers to an input membership function describing the joint position, $\dot{q}$ is the joint velocity, B refers to an input membership function describing the joint velocity, τ is the output torque, C refers to an output membership function describing the torque.

From the initial testing, the recorded motor current reveals that the motor output is at its maximum during chattering. An intuitive choice is to change the output sets to reduce the motor current in the input regions where chattering occurs. Later on, heuristic testing is done to determine the rules in the FAM with the objective of providing a stable and solid feel when in contact with the virtual wall.

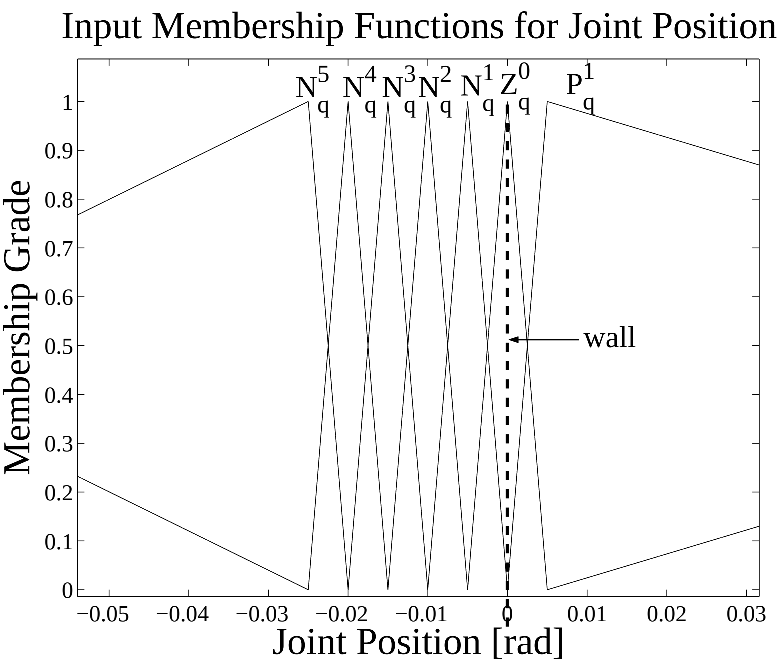


Figure 5.4: Input membership functions of joint position over the range of interest.

Figure 5.6 shows the output membership functions (i.e. N^{86},...N^1,P^1,...P^{31},Z^0...) with different joint position and velocity combinations. Again, the N, P, and Z refer to negative, positive, and zero values respectively of the output membership function and the associated superscripts refer to the index of the output fuzzy set. For example, N^{86} refers to the negative output membership function of the 86^{th} fuzzy set. With 13 membership functions for each input and allowing 1 output rule for every input condition, there are 13x13 output sets representing the commanded output torques to the motor. The following example demonstrates how one of the output rules, N^2, is calculated. Referring to Figure 5.6, the N^2 output set (highlighted in the table) results when joint position is within the domain described by the N_q^2 membership function (Figure 5.4) and when joint velocity is within the domain described the Z_{dq}^0 membership function (Figure 5.5). Then, N^2 is calculated by:

$$N^2 \quad = \quad k_1 \times N^2_{q\,max} + k_2 \times Z^0_{dq_{max}} \tag{5.5}$$

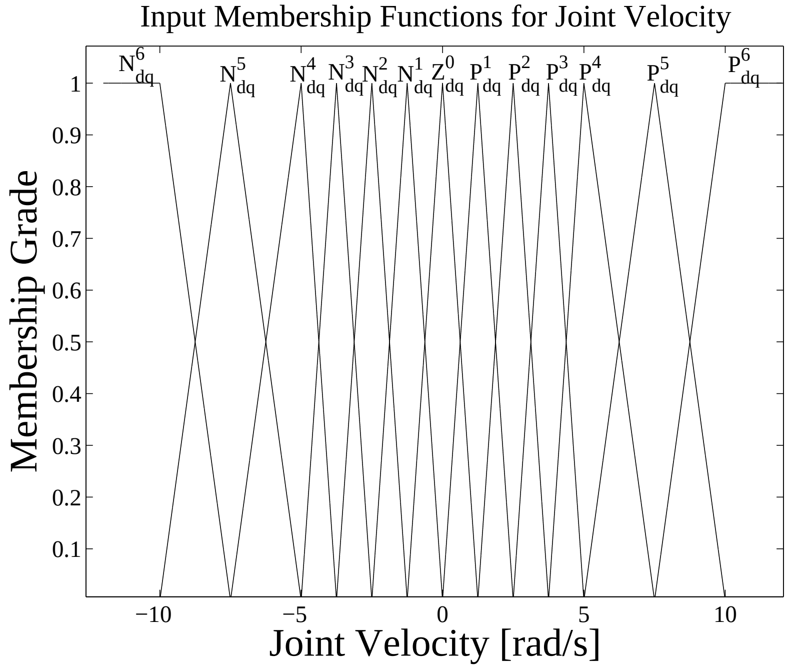
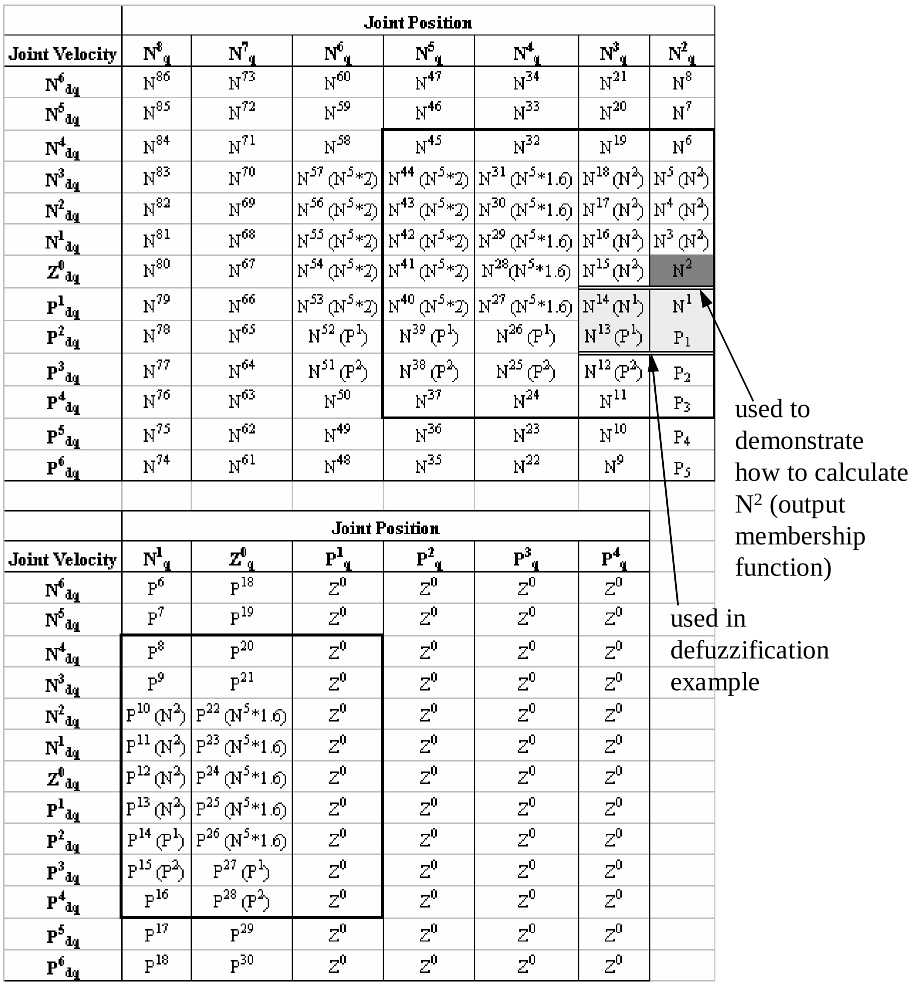


Figure 5.5: Input membership functions of joint velocity over the universe of discourse.

(5.6)

where N^2 represents the negative value for the 2^{nd} torque output set, k_1 is the proportional gain from the virtual wall parameters, $N^2_{q\,max}$ is the maximum value of the position input fuzzy set N^2_q, k_2 is the derivative gain from the virtual wall parameters, $Z^0_{dq_{max}}$ is the maximum value of the velocity input fuzzy set Z^0_{dq}.

Note that the ranges of the defined inputs listed in Figure 5.6 are over the entire universe of discourse (i.e. the 13x13 combinations of input membership functions) of the robot workspace which is not necessarily equal to the operating ranges. Only the ranges of interest are tested and the corresponding rules are modified. Physically, these ranges describe the operation when the link is close to the boundary of the virtual wall. Again, the ranges of interest are position within [-25e-3:5e-3] radians and velocity within [-5:5] radians/s. When these domains are fuzzified according to the input membership functions from Figures 5.4 and 5.5, they are equivalent to the position range described by N^5_q,...,N^1_q,Z^0_q,P^1_q and the velocity range described



	Joint Position						
Joint Velocity	N^8_q	N^7_q	N^6_q	N^5_q	N^4_q	N^3_q	N^2_q
N^6_{dq}	N^{86}	N^{73}	N^{60}	N^{47}	N^{34}	N^{21}	N^8
N^5_{dq}	N^{85}	N^{72}	N^{59}	N^{46}	N^{33}	N^{20}	N^7
N^4_{dq}	N^{84}	N^{71}	N^{58}	N^{45}	N^{32}	N^{19}	N^6
N^3_{dq}	N^{83}	N^{70}	$N^{57} (N^5*2)$	$N^{44} (N^5*2)$	$N^{31} (N^5*1.6)$	$N^{18} (N^2)$	$N^5 (N^2)$
N^2_{dq}	N^{82}	N^{69}	$N^{56} (N^5*2)$	$N^{43} (N^5*2)$	$N^{30} (N^5*1.6)$	$N^{17} (N^2)$	$N^4 (N^2)$
N^1_{dq}	N^{81}	N^{68}	$N^{55} (N^5*2)$	$N^{42} (N^5*2)$	$N^{29} (N^5*1.6)$	$N^{16} (N^2)$	$N^3 (N^2)$
Z^0_{dq}	N^{80}	N^{67}	$N^{54} (N^5*2)$	$N^{41} (N^5*2)$	$N^{28}(N^5*1.6)$	$N^{15} (N^2)$	N^2
P^1_{dq}	N^{79}	N^{66}	$N^{53} (N^5*2)$	$N^{40} (N^5*2)$	$N^{27} (N^5*1.6)$	$N^{14} (N^1)$	N^1
P^2_{dq}	N^{78}	N^{65}	$N^{52} (P^1)$	$N^{39} (P^1)$	$N^{26} (P^1)$	$N^{13} (P^1)$	P_1
P^3_{dq}	N^{77}	N^{64}	$N^{51} (P^2)$	$N^{38} (P^2)$	$N^{25} (P^2)$	$N^{12} (P^2)$	P_2
P^4_{dq}	N^{76}	N^{63}	N^{50}	N^{37}	N^{24}	N^{11}	P_3
P^5_{dq}	N^{75}	N^{62}	N^{49}	N^{36}	N^{23}	N^{10}	P_4
P^6_{dq}	N^{74}	N^{61}	N^{48}	N^{35}	N^{22}	N^9	P_5

	Joint Position					
Joint Velocity	N^1_q	Z^0_q	P^1_q	P^2_q	P^3_q	P^4_q
N^6_{dq}	P^6	P^{18}	Z^0	Z^0	Z^0	Z^0
N^5_{dq}	P^7	P^{19}	Z^0	Z^0	Z^0	Z^0
N^4_{dq}	P^8	P^{20}	Z^0	Z^0	Z^0	Z^0
N^3_{dq}	P^9	P^{21}	Z^0	Z^0	Z^0	Z^0
N^2_{dq}	$P^{10} (N^2)$	$P^{22} (N^5*1.6)$	Z^0	Z^0	Z^0	Z^0
N^1_{dq}	$P^{11} (N^2)$	$P^{23} (N^5*1.6)$	Z^0	Z^0	Z^0	Z^0
Z^0_{dq}	$P^{12} (N^2)$	$P^{24} (N^5*1.6)$	Z^0	Z^0	Z^0	Z^0
P^1_{dq}	$P^{13} (N^2)$	$P^{25} (N^5*1.6)$	Z^0	Z^0	Z^0	Z^0
P^2_{dq}	$P^{14} (P^1)$	$P^{26} (N^5*1.6)$	Z^0	Z^0	Z^0	Z^0
P^3_{dq}	$P^{15} (P^2)$	$P^{27} (P^1)$	Z^0	Z^0	Z^0	Z^0
P^4_{dq}	P^{16}	$P^{28} (P^2)$	Z^0	Z^0	Z^0	Z^0
P^5_{dq}	P^{17}	P^{29}	Z^0	Z^0	Z^0	Z^0
P^6_{dq}	P^{18}	P^{30}	Z^0	Z^0	Z^0	Z^0

used to demonstrate how to calculate N^2 (output membership function)

used in defuzzification example

Figure 5.6: Joint position and velocity FAM.

by N^4_{dq},...,N^1_{dq},Z^0_{dq},P^1_{dq},...,P^4_{dq}. This range is defined as everything inside the thick border lines on Figure 5.6.

Continuing to study Figure 5.6, the Z represents zero torque output set and this situation occurs when the joint position is greater than 0 radian (i.e. the locked arm is outside of the virtual wall). In terms of the other output sets, the prefix P indicates positive output torque set and the magnitude increase with the associated numeric suffix (i.e. P^{10} is a smaller torque than P^{11}). Similarly for the prefixes N, it refers to a negative torque output set (i.e. N^{47} is a more negative torque than N^{46}). The table contains the originally calculated rules based on the PD gains (i.e.

similar to the example done in Equation 5.6) as well as the modified rules based on heuristic testing which are indicated within "()" in Figure 5.6.

From the FAM, the rules are changed for the region near the boundary of the wall to minimize the chattering and within the operating range of input velocity. Physically, a positive calculated output membership value of torque in FAM means a pull on the user while a negative calculated torque means a push on the user. The original rules of N^{57},...,N^{53}, N^{44},...,N^{40}, N^{31},...,N^{27}, N^{18},...,N^{14} and N^{5},...,N^{3}, P^{10},...,P^{13}, P^{22},...,P^{26} are modified to reduce the magnitude of the output torque to achieve stability. Rules P^1 and P^2 are used when the user is in the wall boundary and when the user's velocity is slightly more positive (i.e. user is pulling away from the wall) to create a small resistance as the user to is exiting the wall. At higher velocity the rule is not modified because the velocity is out of the operating range.

5.5.3 Defuzzification

After calculating the output rules in the FAM, the defuzzification process is carried out to compute the output value. The defuzzification process is the computation of a weighted average of the individual centroids of the output sets for each of the contributing rules (see Section 5.4). This is best demonstrated with another example. Suppose given a joint position of -0.0125 radian (belongs to both the N_q^3 and N_q^2 membership functions) and a joint velocity of 1.8 rad/s (belongs to both the P_{dq}^1 and P_{dq}^2 membership functions), this combination of inputs is labelled on Figure 5.6 as "used in defuzzification example". To calculate the output membership function using the values determined from heuristic testing:

$$y_0 = \frac{N^1\mu_{N_q^3}\mu_{P_{dq}^1} + N^1\mu_{N_q^2}\mu_{P_{dq}^1} + P^1\mu_{N_q^3}\mu_{P_{dq}^2} + P^1\mu_{N_q^2}\mu_{P_{dq}^2}}{\mu_{N_q^3}\mu_{P^1 dq} + \mu_{N_q^2}\mu_{P_{dq}^1} + \mu_{N_q^3}\mu_{P_{dq}^2} + \mu_{N_q^2}\mu_{P_{dq}^2}}$$

where y_0 is the output torque, N^1 is a negative output rule value, $\mu_{N_q^3}$ is the value of the membership function N_q^3 evaluated at position = -0.0125, $\mu_{P_{dq}^1}$ is the value of the membership function P_{dq}^1 evaluated at velocity = 1.8, $\mu_{N_q^2}$ is the value of the membership function N_q^2 evaluated at position = -0.0125, $\mu_{P_{dq}^2}$ is the value of the membership function P_{dq}^2 evaluated at velocity = 1.8.

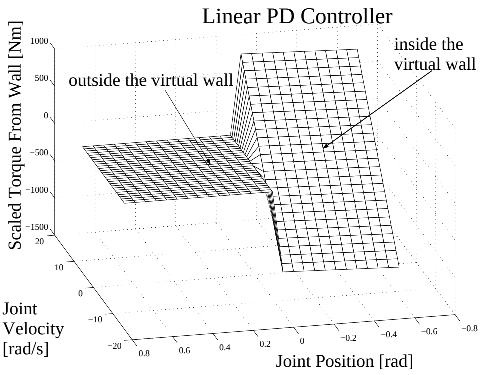


Figure 5.7: Relationship between output torque, joint position and velocity when using a linear PD controller.

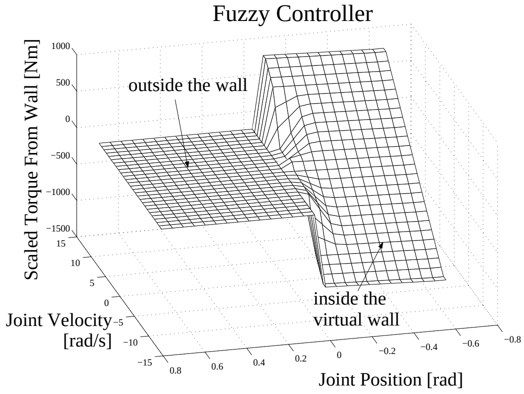


Figure 5.8: Relationship between output torque, joint position and velocity when using a fuzzy controller.

This will give an output torque value that is the average of the weights of the relevant membership functions.

5.6 Linear PD Controller and Fuzzy Controllers Comparison

Figures 5.7 and 5.8 graphically show the output torques resulting from the PD controller and the fuzzy controller respectively. In Figure 5.7, the horizontal plane (i.e. zero output torque) represents the region outside of the virtual wall. A plane with an inclined angle is the resultant torque (τ) calculated using a PD controller at different joint position (q) and velocity ($\dot{q}$) based on the linear relationship $\tau = Pq + D\dot{q}$. In Figure 5.8, a torque output resulting from the fuzzy controller is presented. The effect in output rule modifications near the boundary of the wall can be seen as the ripples in the mesh plot. To graphically accentuate the difference between PD and fuzzy controllers, Figure 5.9 presents a mesh plot of their difference. The difference in torque is based on the modifications done to the rules governing the output fuzzy sets of the operating range close to the virtual wall boundary.

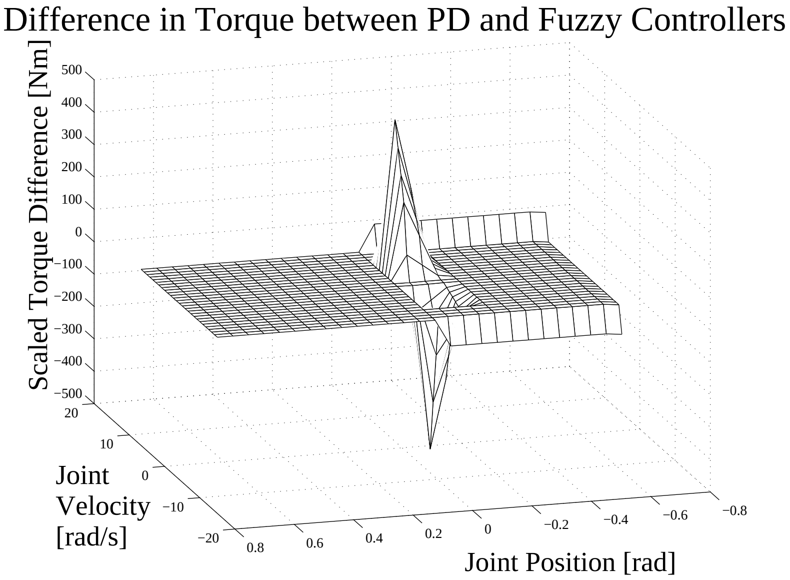


Figure 5.9: The difference in torque between using a PD controller and a fuzzy controller.

5.7 Experimental Result

5.7.1 A Comparison between PD and Fuzzy Controllers in Virtual Wall Contact Application

The objective of implementing the fuzzy controller is to reduce chatter for the user when encountering a virtual wall. A set of tests are done to tune the fuzzy controller to formulate the FAM in Section 5.5.2. This is done by first reducing the magnitude of torque through a reduction in the magnitude of the output rules in the operating range. Then, the magnitude of the output rules are gradually increased to a level where a solid wall contact is felt by the user without resulting in vibrations. Figures 5.10 to 5.12 summarize the results of using PD and fuzzy controller with different approach speeds (i.e. the speed at which the user hits the virtual wall). The three approach speeds are slow (speed $<$ 1 rad/s), medium (1 rad/s $<$ speed $<$ 2 rad/s), and fast (2 rad/s $<$ speed).

With all the approach speeds, the fuzzy controller provides a better feel of a solid wall than the PD controller. This is deduced by the relatively stable joint position

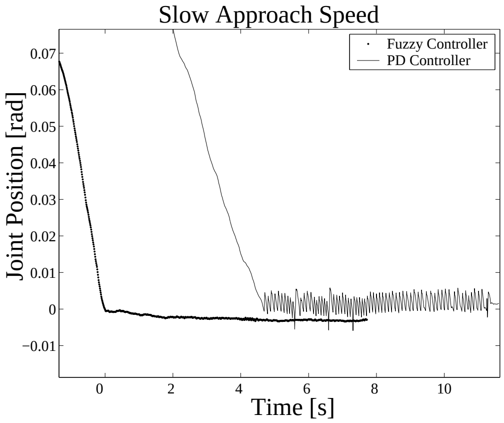


Figure 5.10: A comparison in joint position, when contacting the virtual wall with a slow approach speed, between using a PD controller and a fuzzy controller.

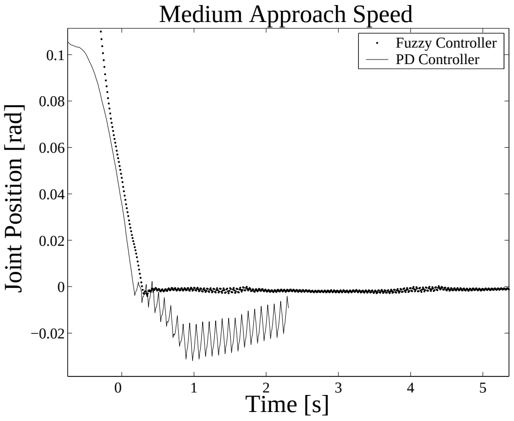


Figure 5.11: A comparison in joint position, when contacting the virtual wall with a medium approach speed, between using a PD controller and a fuzzy controller.

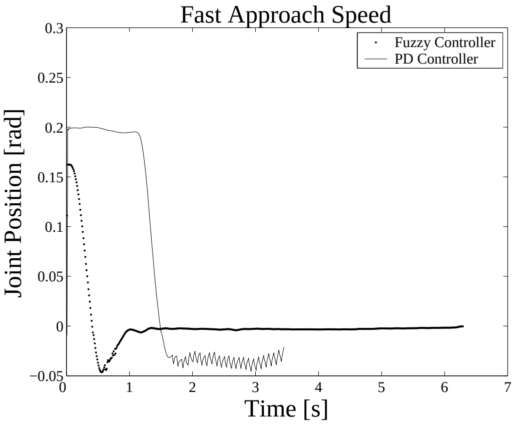


Figure 5.12: A comparison in joint position, when contacting the virtual wall with a fast approach speed, between using a PD controller and a fuzzy controller.

(at around 0 radian where the virtual wall is located) at all approach speeds when a fuzzy controller is implemented. When a PD controller is used, a lot of vibrations are felt by the user and it is evident in the recorded joint position at all approach speeds. This is a proof-of-concept for the effectiveness of using a fuzzy controller on virtual wall applications.

In addition to improving the sense of touch against a virtual wall, an unrelated observation from the results of these experiments is made. When using a PD controller, the steady state joint position, which is equivalent to the location of the wall boundary perceived by the user, differs with different approach speeds. To further investigate this hypothesis, an additional set of experiment is performed with the same virtual wall test set up.

5.7.2 The Relationship Between Perceived Wall Boundary and Approach Speed

This set of experiments is to investigate the relationship between the perceived wall boundary and the approach speed with a PD controller implemented as the virtual wall. Again, only the first joint is tested with the rest of the joints locked in place.

The approach speed is categorized the same way as Section 5.7.1: slow (speed < 1 rad/s), medium (1 rad/s $<$ speed < 2 rad/s), and fast (2 rad/s $<$ speed). The user is to swing the locked arm towards a virtual wall located at 0 radian at different speeds. The joint position and velocity are recorded until the steady state is reached. The test is repeated several times for each approach speed and the results are presented in Figure 5.13.

Only two tests for each speed are presented. The y-axes are scaled identically for the three graphs to facilitate comparison. The objective of these tests is to see if the user is able to correctly identify the wall boundary. For fast approach speeds, there are a lot of vibrations and the user perceives the wall boundary to be at about -0.05 radians (i.e. 0.05 radians penetrated into the wall). There is less penetration, only 0.02 radians, when this test is done with medium approach speeds. With the slow approach speed, the results exhibit the most ideal virtual wall contact with the least vibration and the least wall penetration. From these tests, it is deduced that when the approaching speed is fast, the user would perceive the boundary of the wall to be further into the actual wall boundary. Starting at the beginning

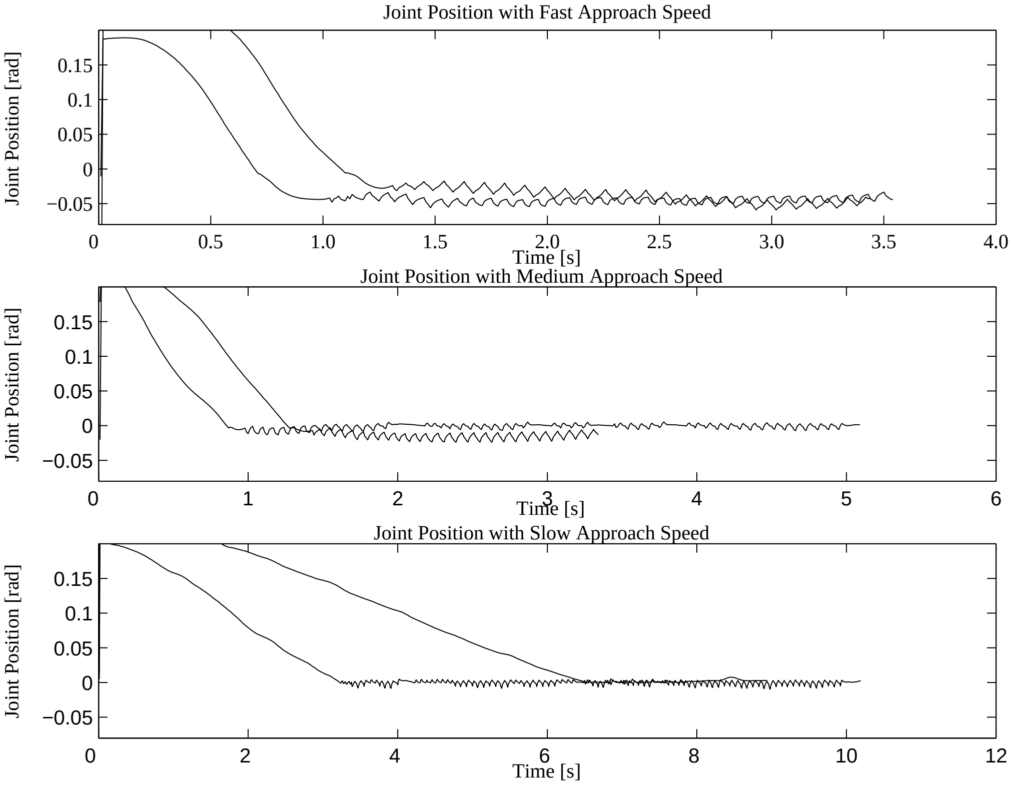


Figure 5.13: The result of using the a PD controller with different approaching speeds.

when the user moves from rest, the force exerted by the user is greater when the approaching speed is high. This greater exerted force from the user would cause more wall penetration. In return, a given set of PD parameters would return a high resistance force. Depending on the tightness of the grip of the user's hand, vibration of various magnitudes would result. With the vibration felt by the user, the grip tightness would generally increase and the exerted force towards the inside of the wall is likely to decrease. This is because the dampened vibration force makes it feels like that something has been hit. Hence, the vibration felt is altering the user's perception of the location of the wall boundary. It can be observed from the tests that the wall boundary is perceived more accurately when a slower approach speed is used.

5.8 Chapter Summary: Control of Vibration during Virtual Wall Contact

In this chapter, two different sets of virtual wall experiments using the joint 0 of Freedom 6S are performed. The first set of experiment is to compare the performance of a PD controller with that of a fuzzy controller. It is proven that a fuzzy controller has the capability to provide a solid feel to the user upon the contact of a virtual wall by the end effector. The implementation of the fuzzy controller can eliminate the vibration that would otherwise be produced by a PD controller implementation. Since the vibration is caused by hardware limitation (see Section 2.5), the implementation of a fuzzy controller is a feasible software solution for a hardware limitation.

The second set of experiments is to investigate the relationship between the approaching speed towards a virtual wall and the steady state position using a PD controller. It is concluded through repeated experiments that a higher approaching speed would lead to an inaccurate perception of the wall boundary. The chattering resulting from the high speed interaction is actually the cause of such altered perception. With a slower approaching speed, the wall boundary can be detected more accurately by the user.

Chapter 6

Conclusions and Future Research

There are two major objectives of this thesis. The first objective is the development and use of system identification methods to dynamically model the Freedom 6S haptic device. By verifying the correctness of the system identification approaches on the Freedom 6S, it demonstrates the potential of the modeling technique being applied to other haptic devices. The second objective of the thesis is to minimize the chattering effect during virtual wall contact by using fuzzy logic. This also illustrates the capacity of fuzzy logic in improving the traditional user and virtual wall interaction.

6.1 Summary of Results and Conclusions

The following sections summarize the results from this thesis in terms of the Freedom 6S modelling and the minimization of chattering in virtual wall contact with the use of fuzzy logic.

6.1.1 System Modelling

The Freedom 6S is a 6 DOF haptic device characterized by nonlinear system dynamics. Its overall model is the amalgamation of Lagrangian dynamics and additional non-linearities including LuGre friction and cable drag. With the large number of unknowns in this analytical model, two system identification methods are proposed to identify the unknowns systematically. Both of them rely on knowing the dynamic equations of the device to generate a subset of dynamic equations describing the joint(s) of interest. Through a locking sequence, test data is generated by exciting the different lock joint configurations with signals of varying amplitudes and frequencies. These test results are then used to create cost functions with the subset dynamic model used in an optimization routine. The optimization then determines the values of the unknown parameters that would best fit the measured data. By doing similar tests and optimizations on different lock joint configurations, all the unknowns are systematically identified.

The first identification method, the method of isolated joint, allows for unknowns to be grouped by the joint of the manipulator. The isolated joint method has the advantage of having the least number of unknowns to be identified at each step. The results presented in Section 4.2.3, however, show the lack of robustness of this technique. Despite the fact that at each step the optimization routine is able to determine the values that would best fit the behaviour of one joint, it is unable to produce a model that would accurately describe the device when joints are moving simultaneously. By only considering the behaviour of the joints individually, the subset of dynamic equations neglects the importance of coupling terms during optimization. Hence, the overall parameter identification generates unsatisfactory results.

The second identification method, the method of coupling joints, identifies the unknown parameters with more than one free joint at a time. Although the number of parameters to be identified at one time has increased, the coupling behaviour of the device can now be captured in the test data using this arrangement. As well, the coupling effects in the tendon-driven wrist become more apparent and the beating phenomenon observed is used to further confirm input couplings between the last two wrist joints. The parameters determined from this method is able to describe the overall dynamic system of the device as seen from the results presented in Section 4.4.

The final verification of the identified model is confirmed by using a closed-loop system with tuned PD gains. The simulation results, as discussed in Section 4.5, show promising similarities with the measured results for all joints. A very important observation made is that one has to make sure the routing of tendons are wounded such that little to no overlapping of tendons exists when tendons are wrapped over the pulleys. When routing of tendons is done properly, the interaction of inputs between the last two joints only affects the roll joint (joint 4) and not the yaw joint (joint 5).

6.1.2 Control of Vibration during Virtual Wall Contact

Again using the Freedom 6S as a haptic device, the vibration phenomenon resulting from the user-to-virtual-wall interaction is explored. Traditionally, a virtual wall, modelled by a spring and a damper, might cause a lot of vibration to the user near the wall boundary due to low sampling rate and/or position sensors with insufficient resolution. An easy and cost-effective way to reduce such “jittering” sensation from the device is to use a fuzzy controller in place of a PD controller. Starting from a PD controller, a final fuzzy controller is designed using fuzzy reasoning. In Section 5.7.1, the performances of using a PD controller and of using a fuzzy controller on a virtual wall application are compared. The fuzzy controller has the capability of reducing the jittering effects around the wall boundary and allows the user to feel the wall surface more effectively. Essentially, a more systematic approach to the development of a fuzzy logic controller is presented.

6.1.3 Relationship between Approaching Speed of the User and the Steady State Position using a PD Controller

An additional observation is made from the virtual wall experiments using a PD controller (i.e. a spring and damper system). It appears that the higher the user approach speed, the more inaccurate the perception of the location of the wall boundary. High approach speed usually causes much chattering around the wall boundary, known generally as the discretization phenomenon, due to the quick change of position with time. With much vibration, the user’s perception of the wall boundary is further into the actual wall boundary because more force is likely to be exerted by the user in an attempt to grab onto the jittering end effector. If the user approaches the wall slowly, less vibration occurs and the user is less inclined to output as much force and hence, the steady state occurs closer to the wall boundary.

6.2 Future Research

The general applications extending from this thesis include the use of the system identification methods proposed to model other haptic devices. Also, the systematic approach on designing fuzzy logic controllers is applicable for general fuzzy controller designs in other applications. A few specific applications of the Freedom 6S model and future research topics for using fuzzy logic on virtual wall simulations are suggested in the following sections.

6.2.1 Model of Freedom 6S

The complete identified dynamic model of Freedom 6S can now be used to create better haptic effects. With the mass, inertia and friction characteristics known, it is desirable to have these quantities "eliminated" so that the user does not feel the load of the device during a free-space work mode. For similar operating points as the inputs used in Section 4.3, the parameters of the model can be used to calculate the torque commands required for the motors to output the equivalent torques to cancel out the inertia and friction effects felt by the user. At this point, a free-space and low-resistance three dimensional motion can be achieved.

With the availability of this dynamic model, it facilities all the future research on controller designs and implementation of Freedom 6S. With a simulation of 15 minutes for 1 minute of input data, it allows for relatively quick prediction of the behaviour of the hardware.

6.2.2 Fuzzy Controller on Virtual Wall Application

In this thesis, the fuzzy controller is applied to a 1 DOF system only. It is ideal to confirm the findings from this simple system by implementing the same type of controller on systems with higher DOFs. With increasing DOFs, the interaction between the joints must be considered and modifications to the FAM must be made to accommodate for coupling effects between joints.

With the availability of the Freedom 6S dynamic model, it is possible to simulate virtual wall contact with an additional human model representing the input to the virtual environment system. This facilitates the offline tuning of the fuzzy logic controller to the desired characteristics of the virtual environment (i.e. rigidity of a virtual wall). In addition, it is ideal to develop strategies on tuning the output rules automatically instead of using purely heuristic testing. This can potentially shorten the development time of the fuzzy controller and it enables a more thorough search of suitable parameters for the output sets.

Appendix A

The Relationship between the Kinematics Derived from DH Representation and from the SDK

The end frame (f_e) produced by the kinematics based on the DH convention measures the endpoint based on the origin at f_0 of Figure 3.2. The Cartesian endpoint described by the SDK, however, measures from an origin located coincidentally with the endpoint when all joints are at 0 radians. In addition, the orientation of the frames are different. The Cartesian endpoint conversion between the two systems is provided in Equations A.1 to A.3. The units are in meters.

$$\begin{aligned} SDK_x &= -(DH_y + 0.23) \quad &(A.1)\\ SDK_y &= DH_x + 0.2 \quad &(A.2)\\ SDK_z &= DH_z \quad &(A.3) \end{aligned}$$

Appendix B

Derivation of Equations of Motion from Euler-Lagrange Equations

The derivation of equations of motion from Euler-Lagrange equations is a summary from [28]. The Lagrangian is defined as:

$$L = K - V \tag{B.1}$$

where K is the kinetic energy and V is the potential energy.

The Euler-Lagrange equation of motion is:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_j} - \frac{L}{q_j} = \tau_j \tag{B.2}$$

where L is the Lagrangian, q_j is the general coordinate of the j-th frame, $j = 1, ..., n$, n is the number of generalized coordinates.

To obtain the general equation of motion from the Euler-Lagrange equations, two conditions must be satisfied:

1. The kinetic energy is a quadratic function of the velocity vector.

   Kinetic energy of a robot, which can be expressed as a function of the velocity vector, can be written as:

   $$K = \frac{1}{2}\dot{\mathbf{q}}^T\mathbf{D}(\mathbf{q})\dot{\mathbf{q}} \tag{B.3}$$

   where D is the nxn inertia matrix.

2. The potential energy of the system is independent of velocity.

   The potential energy describing a system can be calculated by:

   $$V = \sum_p m_p \mathbf{g} s_p \tag{B.4}$$

   where m_p is the mass of the p-th object , g is the gravity vector, s_p is the displacement vector measured from a fixed inertial frame to the center of mass of the p-th object.

   The calculation of potential energy depends on the position of the object only and it is independent of velocity.

Now calculating the terms required for Equation B.2:

$$\begin{aligned} L &= K - V \\ &= \frac{1}{2}\sum_{i,j} d_{ij}(\mathbf{q})\dot{q}_i\dot{q}_j - V(\mathbf{q}) \end{aligned}$$

where d_{ij} is the i,j-th element of the D matrix.

Then partially differentiating L with respect to $\dot{q}_k$:

$$\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj}(\mathbf{q})\dot{q}_j \tag{B.5}$$

Differentiating Equation B.5:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj}(\mathbf{q})\ddot{q}_j + \sum_{i,j}\frac{\partial d_{kj}}{\partial q_i}\dot{q}_i\dot{q}_j \tag{B.6}$$

Now, partially differentiating L with respect to q_k:

$$\frac{\partial L}{\partial q_k} = \frac{1}{2}\sum_{i,j}\frac{\partial d_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j - \frac{\partial V}{\partial q_k} \tag{B.7}$$

Substituting Equations B.6 and B.7 into Equation B.2:

$$\sum_j d_{kj}(\mathbf{q})\ddot{q}_j + \sum_{i,j}(\frac{\partial d_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{ij}}{\partial q_k})\dot{q}_i\dot{q}_j + \frac{\partial V}{\partial q_k} = \tau_k \tag{B.8}$$

By noting the symmetry in the summation and by interchanging the order of summation, the second term can be rewritten as:

$$\sum_{i,j}(\frac{\partial d_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{ij}}{\partial q_k})\dot{q}_i\dot{q}_j = \sum_{i,j}\underbrace{\frac{1}{2}(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k})}\dot{q}_i\dot{q}_j \tag{B.9}$$

The term emphasized by the underbraced bracket is called a Christoffel symbol, $c_{i,j,k}$.

Renaming the gravitational effect term in Equation B.2:

$$\phi_k = \frac{\partial V}{\partial q_k} \tag{B.10}$$

By combining Equations B.9 and B.10, the equations of motion as seen in Equation 3.5 results:

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \phi(\mathbf{q}) = \tau \tag{B.11}$$

where D, the inertia matrix, is defined as:

$$D = \sum_{i=1}^{n} m_i J_{v_{ci}}(q)^T J_{v_{ci}}(q) + J_{\omega_i}(q)^T R_i(q) I_i R_i(q)^T J_{\omega_i}(q) \tag{B.12}$$

where n is the number of links of the robot, m_i is the mass of link i, $J_{v_{ci}}$ is the 3x6 linear portion of the Jacobian matrix based on the center of mass of link i, J_{ω_i} is the 3x6 angular portion of the Jacobian matrix based on the center of mass of link i, R_i is the 3x3 rotational matrix describing link i based on an inertial reference frame, I_i is the 3x3 inertia matrix.

The k,j-th element of C, the Christoffel matrix, is calculated as:

$$C_{kj} = \sum_{i=1}^{n} \frac{1}{2}(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k})\dot{q}_i \tag{B.13}$$

where d_{kj} is the k,j-th element of the $\boldsymbol{D}$ matrix, q_i is the joint position of link i, $\dot{q}_i$ is the joint velocity of joint i.

Appendix C

Scripts from Maple: Generating Inertia, Christoffel, and Gravity Effect Matrices

The following code can be saved as a .txt to be executed in Maple when all “-” are replaced by “#”. This script generates the inertia D, Christoffel C, and gravity effect ϕ matrices.

```
restart:
–declare Maple Libraries with(LinearAlgebra):
————————————————————————-
–declare symbolics for angles, angular velocities, angular accelerations, base angles
–joint angles
q := Vector[row] ( 6, [ q1, q2, q3, q4, q5, q6 ] ):
dq:= Vector[row] ( 6, [ dq1, dq2, dq3, dq4, dq5, dq6 ] ):
ddq:=Vector[column] ( 6, [ ddq1, ddq2, ddq3, ddq4, ddq5, ddq6 ] ):
–base angles (set manually)
b := Vector[column] ( 3, [ b1, b2, b3]): –b3 is not used
–length of each links
– L1 is NOT used since the first link needs three coordinates to describe (i.e. L1x, L1y, L1z )
-these three are the cm coordinate measured from frame 0 and will be defined in the section below
L := Vector[row] (7, [L1, L2, L3, L4,L5, L6, L7] ):
L1x:
L1y:
L1z:
–mass of each link
m := Vector[row] ( 7, [ m1, m2, m3, m4, m5, m6, m7 ] ):
————————————————————————-
–declare symbolics for location of center of mass wrt frame assignment
```

```
–coordinates for center of mass of each link
xyz1 := Vector[column] ( 3, [ x1, y1, z1 ] ): xyz2 := Vector[column] ( 3, [ x2, y2, z2 ] ):
xyz3 := Vector[column] ( 3, [ x3, y3,z3 ] ):
xyz4 := Vector[column] ( 3, [ x4, y4, z4 ] ):
xyz5 := Vector[column] ( 3, [ x5, y5, z5 ] ):
xyz6 := Vector[column] ( 3, [x6, y6, z6 ] ):
xyz7 := Vector[column] ( 3, [ x7, y7, z7 ] ):
————————————————————————————————————-
–declare symbolics for inertia of each link
–inertia of each link - note inertia matrix: Transpose(I) = Transpose
I1 := Matrix( 3, 3, [ [ I1_11, I1_12, I1_13 ], [ I1_12, I1_22, I1_23 ], [ I1_13, I1_23, I1_33 ] ] ):
I2 := Matrix( 3, 3, [ [I2_11, I2_12, I2_13 ], [ I2_12, I2_22, I2_23 ], [ I2_13, I2_23, I2_33 ] ] ):
I3 := Matrix( 3, 3, [ [ I3_11, I3_12, I3_13 ], [ I3_12, I3_22, I3_23 ], [ I3_13, I3_23, I3_33 ] ] ):
I4 := Matrix( 3, 3, [ [I4_11, I4_12, I4_13 ], [ I4_12, I4_22, I4_23 ], [ I4_13, I4_23, I4_33 ] ] ):
I5 := Matrix( 3, 3, [ [ I5_11, I5_12, I5_13 ],[ I5_12, I5_22, I5_23 ], [ I5_13, I5_23, I5_33 ] ] ):
I6 := Matrix( 3, 3, [ [I6_11, I6_12, I6_13 ], [ I6_12, I6_22, I6_23 ], [ I6_13, I6_23, I6_33 ] ] ):
I7 := Matrix( 3, 3, [ [I7_11, I7_12, I7_13 ], [ I7_12, I7_22, I7_23], [ I7_13, I7_23, I7_33 ] ] ):
————————————————————————————————————-
–declare constants
DOF := 6:
NumOfLinks := 7:
g:
–sub in this
Lx;
Lz;
————————————————————————————————————-
–Procedure
————————————————————————————————————-
TransMtx := proc( Theta, Trans_z, Trans_x, Alpha )
Matrix( 4, 4, [ [ cos(Theta), -sin(Theta)*cos(Alpha), sin(Theta)*sin(Alpha), Trans_x*cos(Theta) ], [ sin(Theta),
cos(Theta)*cos(Alpha), -cos(Theta)*sin(Alpha), Trans_x*sin(Theta) ], [ 0, sin(Alpha), cos(Alpha), Trans_z ], [ 0,
0, 0, 1 ] ] ):
end proc:
————————————————————————————————————-
DH1_f0_f1 := proc()
f0_f1 := TransMtx( q[1], 0, 0, Pi/2 ):
end proc:
————————————————————————————————————-
DH1_f1_f2 := proc()
f1_f2 := TransMtx( q[2], L[2], 0, -Pi/2):
end proc:
————————————————————————————————————-
DH1_f2_f3 := proc()
f2_f3 := TransMtx( q[3]-Pi/2, 0, 0, -Pi/2 ):
end proc:
————————————————————————————————————-
DH1_f3_f4 := proc()
f3_f4 := TransMtx( q[4], - L[5]-L[6], 0, Pi/2 ):
end proc:
————————————————————————————————————-
DH1_f4_f5 := proc()
```

```
f4_f5 := TransMtx( q[6]-Pi/2, 0, 0, -Pi/2 ):
end proc:
————————————————————————-
DH1_f5_fe := proc()
f5_fe := TransMtx( q[5], L[7], 0, 0 ):
end proc:
————————————————————————-
DH2_f0_f1 := proc()
f0_f1 := TransMtx( q[1], 0, 0, Pi/2 ):
end proc:
————————————————————————-
DH2_f1_f2 := proc()
f1_f2 := TransMtx( q[2], 0, 0, -Pi/2):
end proc:
————————————————————————-
DH2_f2_f3 := proc()
f2_f3 := TransMtx( q[3], 0, L[3], 0 ):
end proc:
————————————————————————-
DH2_f3_fe := proc()
f3_fe := TransMtx( -q[3]-Pi/2, 0, L[4], 0 ):
end proc:
————————————————————————-
DH3_f0_fe := proc()
f0_f1 := TransMtx( q[1]+Pi/2, L1z, L1x, 0 ):
TranslationMtx := Matrix ( 4, 4, [ [ 1, 0, 0, 0 ], [ 0, 1, 0, L1y ], [ 0, 0, 1, 0 ], [ 0, 0, 0, 1 ] ] ):
f0_fe := f0_f1 . TranslationMtx:
end proc:
————————————————————————-
FindTransVec := proc( A )
Vector[column]( 3, [ A[1,4], A[2,4], A[3,4] ] ):
end proc:
————————————————————————-
FindRotMtx := proc( A )
Matrix( 3, 3, [ [ A[1,1], A[1,2], A[1,3] ], [ A[2,1], A[2,2], A[2,3] ], [ A[3,1], A[3,2], A[3,3] ] ] ):
end proc:
————————————————————————-
FindJv := proc( d )
-d comes in as 3x1 vectors
Jv := Matrix( 3, DOF ):
for i from 1 by 1 to DOF do
for j from 1 by 1 to 3 do
Jv[j,i] := diff( d[j], q[i] ):
end do:
end do:
Jv:
end proc:
————————————————————————-
Findw := proc( R, JointAngle )
-R is 3x3 Rotation matrix
Temp := Matrix( 3, 3 ):
```

```
for col from 1 by 1 to 3 do
for row from 1 by 1 to 3 do
–diff function is used on algebraic expr, not matrix
Temp[ row, col ] := diff( R[ row, col ], JointAngle ):
end do:
end do:

wMtx := Temp . Transpose( R ):
w := Vector[column]( 3, [ wMtx[3,2], wMtx[1,3], wMtx[2,1] ] ):

end proc:
————————————————————————————-
FindJw := proc( R1, R2, R3, R4, R5, R6 )
-each input is a 3x3
w1 := Findw( R1, q[1] ):
w2 := Findw( R2, q[2] ):
w3 := Findw( R3, q[3] ):
w4 := Findw( R4, q[4] ):
w5 := Findw( R5, q[5] ):
w6 := Findw( R6, q[6] ):
Jw := Matrix( 3, DOF, [ w1, w2, w3, w4, w5, w6 ] ):
end proc:

————————————————————————————-
NonNumericScalarMultiplyMatrix3x6 := proc( s_in1, Mtx )
Matrix( 3, 6, [ [ Mtx[1,1] * s_in1, Mtx[1,2] * s_in1, Mtx[1,3] * s_in1, Mtx[1,4] * s_in1, Mtx[1,5] * s_in1, Mtx[1,6] *
s_in1 ], [ Mtx[2,1] * s_in1, Mtx[2,2] * s_in1, Mtx[2,3] * s_in1
end proc:
————————————————————————————- NonNumericScalarMultiplyVector3x1 := proc( s_in2, Vec )
Vector[column]( 3, [ Vec[1] * s_in2, Vec[2] * s_in2, Vec[3] * s_in2 ] ):
end proc:
————————————————————————————-
MassMtx := proc( m_in, Jv_in, Jw_in, R_in, I_in )
-m_in = 1x1; symbolic or numeric
-Jv_in = 3x6; symbolic
-Jw_in = 3x6; symbolic
-R_in = 3x3; symbolic
-I_in = 3x3; symbolic or numeric
MassM_perLink := Matrix( 6, 6 ):
MassM_perLink := Transpose( NonNumericScalarMultiplyMatrix3x6( m_in, Jv_in ) ) . Jv_in + Transpose( Jw_in )
. R_in . I_in . Transpose( R_in ) . Jw_in:
end proc:
————————————————————————————-
ChrisSym := proc()
-SymMassMtx = 6x6 6DOF mass matrix
for k from 1 by 1 to 6 do
for i from 1 by 1 to 6 do
for j from 1 by 1 to 6 do
c[ i, j, k ] := 0.5 * ( diff( SymMassMtx[k,j], q[i] ) + diff( SymMassMtx[k,i], q[j] ) - diff( SymMassMtx[i,j], q[k] ) ):
k;
i;
```

```
j;
end do:
end do:
end do:
for k from 1 by 1 to 6 do
for j from 1 by 1 to 6 do
for i from 1 by 1 to 6 do
if (i = 1) then
C6x6[ k, j ] := c[ i, j, k ] * dq[i]
else
C6x6[ k, j ] := c[ i, j, k ] * dq[i] + C6x6[ k, j ]
end if:
k;
j;
i;
end do:
end do:
end do:
Matrix( 6, 6, [ [ C6x6[1,1], C6x6[1,2], C6x6[1,3], C6x6[1,4], C6x6[1,5], C6x6[1,6] ], [ C6x6[2,1], C6x6[2,2], C6x6[2,3],
C6x6[2,4], C6x6[2,5], C6x6[2,6] ], [ C6x6[3,1], C6x6[3,2], C6x6[3,3], C6x6[3,4], C6x6[3,5], C6x6[3,6] ], [ C6x6[4,1],
C6x6[4,2], C6x6[4,3], C6x6[4,4], C6x6[4,5], C6x6[4,6] ], [ C6x6[5,1], C6x6[5,2], C6x6[5,3], C6x6[5,4], C6x6[5,5],
C6x6[5,6] ], [ C6x6[6,1], C6x6[6,2], C6x6[6,3], C6x6[6,4], C6x6[6,5], C6x6[6,6] ] ] ):
end proc:
——————————————————————————————-
GravityVec := proc()
fb1_b2_b0 := TransMtx( b[1], 0, Lx, Pi/2 ) . TransMtx( b[2], Lz, 0, 0 ):
with(linalg):
Temp := inverse( fb1_b2_b0 ):
with(LinearAlgebra):
– Vector representing gravity vector in terms of frame b0
TempVec := Vector[column] ( 3, [ Temp[1,2], Temp[2,2], Temp[3,2] ] ):
– Mapping TempVec into frame 0
Vector[column] ( 3, [ TempVec[2], TempVec[1], -TempVec[3] ] ):
end proc:
——————————————————————————————-
Phi := proc()
-each input is a 3x1 vector locating the center of gravity of each link with respect to frame 0
V := 0:
V1 := 0;
V2 := 0;
V3 := 0;
V4 := 0;
V5 := 0;
V6 := 0;
V7 := 0;

-use for other orientation for base angle
s := GravityVec():
-s := Vector[column] ( 3, [ 0, 0, 1 ] ):
V1 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[1],
d_m1 ):
```

```
V2 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[2],
d_m2 ):
V3 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[3],
d_m3 ):
V4 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[4],
d_m4 ):
V5 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[5],
d_m5 ):
V6 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[6],
d_m6 ):
V7 := Transpose ( NonNumericScalarMultiplyVector3x1( g, s ) ) . NonNumericScalarMultiplyVector3x1( m[7],
d_m7 ):
V := V1 + V2 + V3 + V4 + V5 + V6 + V7:
for i from 1 by 1 to DOF do
Phi_Return[i] := diff( V, q[i] ):
end do:
Vector[column]( 6, [ Phi_Return[1], Phi_Return[2], Phi_Return[3], Phi_Return[4], Phi_Return[5], Phi_Return[6] ] );
end proc:
——————————————————————————-
Get3x1From4x1 := proc( Vec )
Vector[column]( 3, [ Vec[1], Vec[2], Vec[3] ] ):
end proc:
——————————————————————————-
–START main program
printlevel := 4:
——————————————————————————-
–Part 1 – 4x4 Transformation Matrice describing each link
EmptyMtx3x3 := Matrix( 3, 3 ):
–DH1 Forward Kinematics and Jacobian T_DH1_1 := DH1_f0_f1():
R_DH1_1 := FindRotMtx ( T_DH1_1 ):
T_DH1_2 := T_DH1_1 . DH1_f1_f2():
R_DH1_2 := FindRotMtx ( T_DH1_2 ):
T_DH1_3 := T_DH1_2 . DH1_f2_f3():
R_DH1_3 := FindRotMtx ( T_DH1_3 ):
T_DH1_4 := T_DH1_3 . DH1_f3_f4():
R_DH1_4 := FindRotMtx ( T_DH1_4 ):
T_DH1_5 := T_DH1_4 . DH1_f4_f5():
R_DH1_5 := FindRotMtx ( T_DH1_5 ):
T_DH1_e := T_DH1_5 . DH1_f5_fe():
R_DH1_e := FindRotMtx ( T_DH1_e ):
T_m2 := T_DH1_2:
-d_m2 := FindTransVec( T_m2 ):
d_m2 := Get3x1From4x1( T_m2 . Vector[column]( 4, [ xyz2[1], xyz2[2], xyz2[3], 1 ] ) ):
R_m2 := FindRotMtx( T_m2 ):
Jv_m2 := simplify( FindJv( d_m2 ) ):
Jw_m2 := simplify( FindJw(R_DH1_1, R_DH1_2, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3 ) ):
T_m5 := T_DH1_3:
-d_m5 := FindTransVec( T_m5 ):
d_m5 := Get3x1From4x1( T_m5 . Vector[column]( 4, [ xyz5[1], xyz5[2], xyz5[3], 1 ] ) ):
R_m5 := FindRotMtx( T_m5 ):
Jv_m5 := simplify( FindJv( d_m5 ) ):
```

```
Jw_m5 := simplify( FindJw( R_DH1_1, R_DH1_2, R_DH1_3, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3 ) ):
T_m6 := T_DH1_4:
-d_m6 := FindTransVec( T_m6 ):
d_m6 := Get3x1From4x1( T_m6 . Vector[column]( 4, [ xyz6[1], xyz6[2], xyz6[3], 1 ] ) ):
R_m6 := FindRotMtx( T_m6 ):
Jv_m6 := simplify( FindJv( d_m6 ) ):
Jw_m6 := simplify( FindJw( R_DH1_1, R_DH1_2, R_DH1_3, R_DH1_4, EmptyMtx3x3, EmptyMtx3x3 ) ):
T_m7 := T_DH1_e :
-d_m7 := FindTransVec( T_m7 ):
d_m7 := Get3x1From4x1( T_m7 . Vector[column]( 4, [ xyz7[1], xyz7[2], xyz7[3], 1 ] ) ):
R_m7 := FindRotMtx( T_m7 ):
Jv_m7 := simplify( FindJv( d_m7 ) ):
——— DH1_f4_f5 transformation involves J5 (i.e. q[6] )
——— DH1_f5_fe transformation involves J4 (i.e. q[5] )
——— The Rotational Matrix order must be reversed before inputting to FindJw
Jw_m7 := simplify( FindJw( R_DH1_1, R_DH1_2, R_DH1_3, R_DH1_4, R_DH1_e, R_DH1_5 ) ):
–DH2 Forward Kinematics and Jacobian
T_DH2_1 := DH2_f0_f1():
R_DH2_1 := FindRotMtx ( T_DH2_1 ):
T_DH2_2 := T_DH2_1 . DH2_f1_f2():
R_DH2_2 := FindRotMtx ( T_DH2_2 ):
T_DH2_3 := T_DH2_2 . DH2_f2_f3():
R_DH2_3 := FindRotMtx ( T_DH2_3 ):
T_DH2_e := T_DH2_3 . DH2_f3_fe():
R_DH2_e := FindRotMtx ( T_DH2_e ):
T_m3 := T_DH2_3:
-d_m3 := FindTransVec( T_m3 ):
d_m3 := Get3x1From4x1( T_m3 . Vector[column]( 4, [ xyz3[1], xyz3[2], xyz3[3], 1 ] ) ):
R_m3 := FindRotMtx( T_m3 ):
Jv_m3 := simplify( FindJv( d_m3 ) ):
Jw_m3 := simplify( FindJw( R_DH2_1, R_DH2_2, R_DH2_3, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3 ) ):
T_m4 := T_DH2_e:
-d_m4 := FindTransVec( T_m4 ):
d_m4 := Get3x1From4x1( T_m4 . Vector[column]( 4, [ xyz4[1], xyz4[2], xyz4[3], 1 ] ) ):
R_m4 := FindRotMtx( T_m4 ):
Jv_m4 := simplify( FindJv( d_m4 ) ):
Jw_m4 := simplify( FindJw( R_DH2_1, R_DH2_2, R_DH2_e, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3 ) ):
–DH3 Forward Kinematics and Jacobian T_m1 := DH3_f0_fe():
-d_m1:= FindTransVec( T_m1 ):
d_m1 := Get3x1From4x1( T_m1 . Vector[column]( 4, [ xyz1[1], xyz1[2], xyz1[3], 1 ] ) ):
R_m1 := FindRotMtx( T_m1 ):
Jv_m1 := simplify( FindJv( d_m1 ) ):
Jw_m1 := simplify( FindJw( R_m1, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3, EmptyMtx3x3
) ):
—————————————————————————-
–Part 2 – 6x6 Mass Matrix
DoneForwardKin;
–Making mass matrix of each link
MassMtx1 := simplify(MassMtx( m[1], Jv_m1, Jw_m1, R_m1, I1 ) ):
DoneReadingMassMtx1;
MassMtx2 := MassMtx( m[2], Jv_m2, Jw_m2, R_m2, I2 ):
```

```
DoneReadingMassMtx2;
MassMtx3 := MassMtx( m[3], Jv_m3, Jw_m3, R_m3, I3 ):
DoneReadingMassMtx3;
MassMtx4 := MassMtx( m[4], Jv_m4, Jw_m4, R_m4, I4 ):
DoneReadingMassMtx4;
MassMtx5 := MassMtx( m[5], Jv_m5, Jw_m5, R_m5, I5 ):
DoneReadingMassMtx5;
MassMtx6 := MassMtx( m[6], Jv_m6, Jw_m6, R_m6, I6 ):
DoneReadingMassMtx6;
MassMtx7 := MassMtx( m[7], Jv_m7, Jw_m7, R_m7, I7 ):
DoneReadingMassMtx7;
–Overall mass matrix
SymMassMtx := MassMtx1 + MassMtx2 + MassMtx3 + MassMtx4 + MassMtx5 + MassMtx6 + MassMtx7:
SymMassMtx := simplify( SymMassMtx ):
DoneSimplifyingSymMassMtx;
————————————————————————-
–Part 3 – 6x6 Christoffel symbols
–Making Christoffel symbol of each link
SymChrisSym :=ChrisSym(): SymChrisSym := simplify( SymChrisSym ):
DoneSimplifyingSymChrisSym;
————————————————————————-
–Part 4 – 6x1 Phi
–Making mass matrix of each link
SymPhi := simplify( Phi() ): DoneSimplifyingSymPhi;
————————————————————————-
DoneReading;
```

Appendix D

Initial Guesses and Final Values of Freedom 6S Parameters

The initial values from obtained from SolidWorks and the final parameters determined from both system identification methods are presented in the table below.

Parameter	Initial Guess	Result from Isolated Joint	Result from Coupled Joint
Kp_0	1.2032	3.2663	1.6534
Kp_1	8e-1	2.7194	1.5205
Kp_2	1	5.595e-1	4.6094
Kp_3	12	1.18876e1	1.4591e1
Kp_4	1.3149e1	1.3062	1.3149e1
Kp_5	5.4164	2.9052	5.4164
Ks_0	1e-1	1.209e-1	1.0469e-1
Ks_1	15694e-1	1.93e-2	9.4145e-2
Ks_2	4.5936e-2	4.59-2	1.311e-2
Ks_3	8e-3	6.4e-3	6.4599e-3
Ks_4	3.6946e-3	1.5e-3	3.6946e-3
Ks_5	7.8684e-3	1.1e-2	7.8684e-3
q_{eq_0}	0	-2.847e-1	-2.1791-2
q_{eq_1}	1.6204e-1	1.094e-1	1.9126e-1
q_{eq_2}	0	-9.89e-2	-4.5725e-2
q_{eq_3}	2.9589e-1	2.599e-1	3.7468e-1
q_{eq_4}	-2.5e-1	5.130e-1	-2.5e-1
q_{eq_5}	-4.1365e-1	-6.155e-1	-4.1365e-1
σ_{0_0}	8e-2	3.8115	6.4282e-2
σ_{0_1}	8e-3	1.2292	2.2977e-2
σ_{0_2}	3.3475e-1	3.348e-1	1.7837e-1
σ_{0_3}	5	3.8865	6.7042
σ_{0_4}	1.7618e-1	1.6134	1.7618e-1
σ_{0_5}	2.1098e-1	6.120e-1	2.1098e-1

Parameter	Initial Guess	Result from Isolated Joint	Result from Coupled Joint
$\sigma1_0$	1e-1	1e-3	1.2118e-1
$\sigma1_1$	9.6706e-3	3.73e-2	2.414e-2
$\sigma1_2$	2.0594e-2	2.06e-2	9.8753e-2
$\sigma1_3$	2e-3	9-4	6.7031e-2
$\sigma1_4$	2.9459e-2	7.5e-3	2.9459e-2
$\sigma1_5$	4.6825e-6	1e-6	4.6825e-6
$\sigma2_0$	1e-3	4.57e-2	3.513e-3
$\sigma2_1$	1.5004e-3	1.32e-2	3.0305e-3
$\sigma2_2$	8.8839e-4	9e-4	9.4327e-3
$\sigma2_3$	7.2352e-2	6.10e-2	6.0302e-2
$\sigma2_4$	8.476e-4	1e-4	8.476e-4
$\sigma2_5$	0.0019215	9.375e-4	0.0019215
v_{s_0}	5e-2	2.225e-1	4.2042e-2
v_{s_1}	5e-1	1.2545	1.3265e-1
v_{s_2}	1.8957e-1	1.896e-1	1.9292e-2
v_{s_3}	4.7645e-1	6.872e-1	7.5417e-1
v_{s_4}	1.9765e-1	2.645e-1	1.9765e-1
v_{s_5}	8.0225e-9	1e-8	8.0225e-9
Fs_0	1e-2	5.89e-2	1.4427e-2
Fs_1	4.7458e-2	2.39e-2	7.0802e-2
Fs_2	5.8533e-3	5.9e-3	7.1487e-2
Fs_3	6.4191e-2	7.63e-2	1.0651e-1
Fs_4	4.6506e-2	4.6e-3	4.6506e-2
Fs_5	6.2877e-2	5.44e-2	6.2877e-2
Fc_0	1e-2	3.33e-2	8.2281e-3
Fc_1	4.7458e-2	1e-5	3.0103e-2
Fc_2	8.4123e-3	8.4e-3	4.4363e-2
Fc_3	5e-3	1.3e-3	1.199e-2
Fc_4	2.2096e-2	4.4e-3	2.2096e-2
Fc_5	2e-2	1.93e-2	2e-2
$FrictionBias_0$	1e-1	na	1.8525
$FrictionBias_1$	1.119	na	1.1186
$FrictionBias_2$	1	na	1.6506
$Gearing4To5$	1	na	1.01
$Gearing5To4$	0.2	na	0.20362
I1_33	-5.68e-3	2.84e-2	1.44e-3
m1	1.1639	7.759e-1	1.098
x1	2.779e-2	3.97e-2	3.7962e-2
y1	6.37e-3	4.9e-3	5.117e-3

Parameter	Initial Guess	Result from Isolated Joint	Result from Coupled Joint
I2_11	1.518e-2	6.9e-3	1.4817e-2
I2_12	-2.22e-5	-1e-6	2e-7
I2_13	-1.29e-3	0	2.4494e-5
I2_22	1.5926e-3	7.2389e-4	1.5916e-3
I2_23	-2.004e-5	-1.8e-8	3.6e-9
I2_33	-1.29e-3	6.5e-3	2.4552e-3
m2	1.972e-1	2.966e-1	3.7559e-1
x2	-2.6e-3	-2e-3	-2.3218e-3
y2	2.2919e-1	1.763e-1	2.0635e-1
z2	2.47e-3	1.9e-3	2.1748e-3
I3_11	-1.86e-4	9.3e-3	8.1086e-3
I3_12	8.462e-8	-4.231e-7	-1.8682e-5
I3_13	-1e-4	0	8.2503e-8
I3_22	2.09e-2	9.5e-3	2.0872e-2
I3_23	-2.403e-7	1.2015e-6	5.7477e-7
I3_33	2.893e-5	2.8993e-5	2.893e-5
m3	6.2087e-1	6.209e-1	6.2087e-1
x3	-2e-3	-2e-3	-2e-3
y3	-2e-3	-2e-3	-2e-3
z3	-1.365e-2	-1.95e-2	-1.3862e-2
I4_11	2.1012e-3	9.5511e-4	2.1006e-3
I4_12	2.22e-5	-1e-6	-2e-7
I4_13	6.562e-7	-3.281e-6	-2.7218e-5
I4_22	1.21e-2	5.5e-3	8.3644e-3
I4_23	7.4934e-5	0	1.8487e-6
I4_33	-1.8951e-4	9.4753e-4	-1.7449e-4
m4	6.79e-2	6.79e-2	6.79e-2
x4	-1.3951e-1	-1.993e-1	-1.4726e-1
y4	-2.6e-3	-2e-3	-1.8605e-3
z4	1.3e-2	1e-2	1.6083e-2
I5_11	4.62e-3	2.1e-3	4.5974e-3
I5_12	-2.022e-5	-1e-7	-2.022e-5
I5_13	2.022e-5	1e-7	1.8053e-5
I5_22	5.079e-5	5.079e-5	5.079e-5
I5_23	2.2009e-5	9.1311e-7	6.9037e-6
I5_33	-3.419e-5	1.7095e-4	1.2204e-4
m5	2.0068e-1	2.007e-1	2.0068e-1
x5	2e-3	2e-3	2e-3
y5	-2.6e-3	-2e-3	-2.5994e-3
z5	-5.04e-2	-5.04e-2	-5.04e-2

Parameter	Initial Guess	Result from Isolated Joint	Result from Coupled Joint
I6_11	1.1345e-4	1.1345e-4	1.1345e-4
I6_12	1e-7	1e-7	1e-7
I6_13	-1e-7	-1e-7	-1e-7
I6_22	1.9046e-005	5.7137e-5	8.5117e-6
I6_23	-4.2346e-7	-4.2346e-7	-4.2346e-7
I6_33	2.0357e-5	2.0357e-5	2.0357e-5
m6	3.985e-2	2.43e-2	4.9604e-2
x6	2e-3	2e-3	1.9672e-3
y6	4.689e-2	4.69e-2	4.689e-2
z6	-5.3e-4	6.047e-4	-6.3571e-4
I7_11	1.3671e-4	1.0499e-4	1.3671e-4
I7_12	-1.6781e-6	-1e-6	-1.6781e-6
I7_13	-4.8383e-8	-3.54e-8	-4.8383e-8
I7_22	2.7242e-3	9.8447e-5	2.7242e-3
I7_23	-1.9798e-6	-1e-6	-1.9798e-6
I7_33	2.2739e-6	5.9470e-6	2.2739e-6
m7	6.0061e-2	1.96e-2	6.0061e-2
x7	-8.4021e-4	-3e-3	-8.4021e-4
y7	-1.1861e-3	-3e-3	-1.1861e-3
z7	-1.4518e-1	-1.654e-1	-1.4518e-1

Appendix E

Determination of Upper and Lower Bounds used for Optimization

Inertia is related to mass and centroid position by:

$$I = \int r^2 \, \mathrm{d}m \tag{E.1}$$

where r is the distance measured from the center of mass to the location of dm, m is the mass of the object.

For ease of calculation for a rough estimation, the mass moment of inertia relationship of a point mass is used to calculate the error propagated in inertia parameter.
Assuming a point mass, its mass moment of inertia is:

$$I = mr^2 \tag{E.2}$$

In Section 4.1.3, a mass error of +/-50 percent and a centroid error of +/-30 percent are specified. The estimated error for inertia parameters can be calculated as follows:

$$Error_{inertia} = \frac{(m + \delta m) \times (r + \delta r)^2 - mr^2}{mr^2} \tag{E.3}$$

After expansion, simplification and collecting like terms:

$$Error_{inertia} = 2 \times \frac{\delta r}{r} + (\frac{\delta r}{r})^2 + \frac{\delta m}{m} \tag{E.4}$$

As previously specified,

$$\frac{\delta r}{r} = 0.3 \tag{E.5}$$

$$\frac{\delta m}{m} = 0.5 \tag{E.6}$$

Substituting the above value into E.4, the error in inertia parameters is calculated to be +/-1.20 or 120 percent.

Appendix F

Test Setup

The main hardware required are outlined in this appendix and a block diagram of the overall setup is presented in Figure F. More detailed equipment setup information is documented in [69].

F.1 Freedom 6S Hand Controller

The robot is assembled as outlined by the mechanical assembly manual provided by MPB Technologies Inc [71]. For the experiment, the configurable base angles (i.e. the additional DOFs to the 6 motorized joints) are fixed: base angle 1 is set to 0 degrees and base angle 2 is set to 180 degrees, as read from dial labels on the device. This is the configuration from which all the testings are done.

An interface panel is attached to the back of the robot and it has ports with cables connecting to the current amplifier box, the power supply box, and the analog port of the analog-to-digital card (ADC) mounted inside the PC.

F.2 Power Supply

The power supply unit is the power source to the sensors and motors of the Freedom 6S. It provides +/-15 V DC for the position sensors and +/-28 V DC at 12 A to actuate the motors. It is connected to the interface panel of the device, the current amplifier, and power outlet.

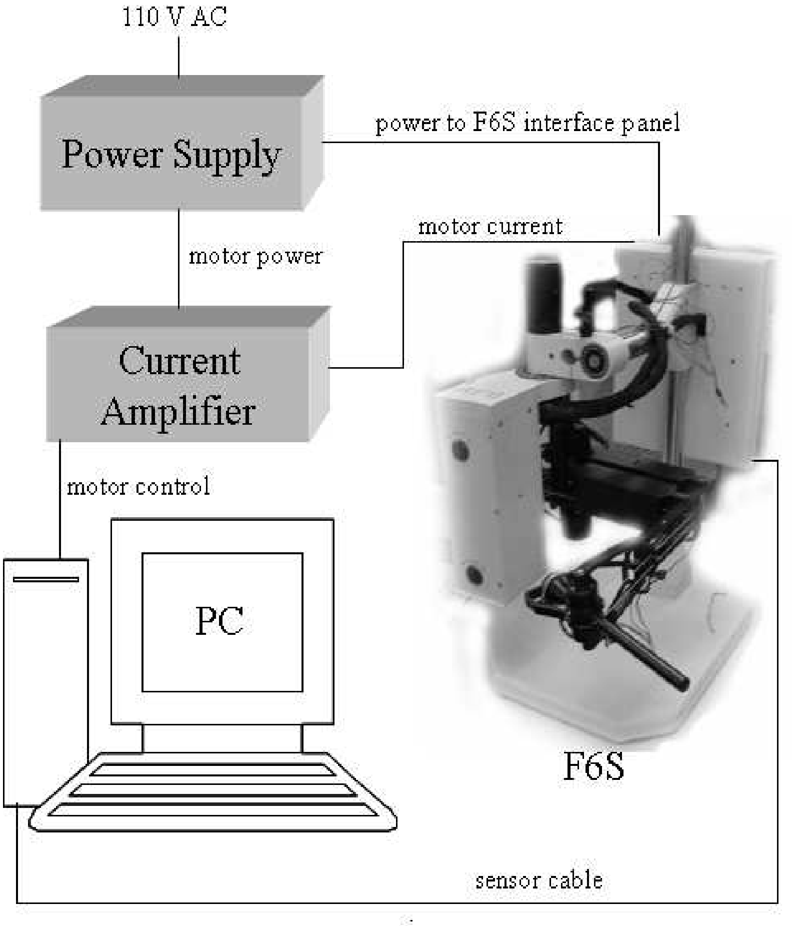


Figure F.1: Equipment setup.

F.3 Current Amplifier

The current amplifier amplifies signals from the PC to motors. It connects to the digital-to-analog (DAC) card in the PC, the interface panel of the device, and to the power supply.

F.4 PC

There are two interface cards that facilitate the Freedom 6S software to interact with the device: an ADC to receive input measurement of sensor data and a DAC to output current signals to control the motors. Detail specification and installation procedures are documented in the MPB user manual [69].

References

[1] Immersion. (2005) What is haptics. Immersion. 801 Fox Lane, San Jose, California 95131 USA. [Online]. Available: http://www.immersion.com/corporate/background/

[2] Webster's. (2005) Webster's online dictionary the rosetta edition. [Online]. Available: http://www.websters-online-dictionary.org

[3] Immersion. (2005) Physiology of haptics. Immersion. 801 Fox Lane, San Jose, California 95131 USA. [Online]. Available: http://www.immersion.com/corporate/press_room/what_is_haptics.php

[4] N. Ando, K. Morioka, P. T. Szemes, and H. Hashimoto, "Development of 6-dof haptic interface for tele-micromanipulation systems," in *IECON '01 IEEE 27th Annual Conf.*, vol. 1, Denver, USA, Nov. 2001, pp. 444–449.

[5] N. Cauche, A. Delchambre, P. Rouiller, P. Helmer, C. Baur, and R. Clavel, "Rotational force-feedback wrist," in *Proc. Int. IEEE Symposium on Assembly and Task Planning*, Besançon, France, July 2003, pp. 210–215.

[6] C. Avizzano, F. Bargagli, A. Frisoli, and M. Bergamasco, "The hand force feedback: analysis and control of a haptic device for the human-hand," in *IEEE Int. Conf. on Systems, Man, and Cybernetics*, vol. 2, Nashville, USA, 2000, pp. 989–994.

[7] J. Kennedy, *Handbook of Perception.* New York, USA: Academic Press, 1978.

[8] C. R. Flatau, F. J. Greeb, and R. A. Booker, "Some preliminary correlations between control modes of manipulator systems and their performance

indices," in *Proc. 1st Nat. Conf. Remotely Manned Syst.*, Pasadena, CA, 1973, pp. 189–198.

[9] H. Fornoff and W. G. Thornton, "Experimental evaluation of remote manipulator systems," in *Proc. 1st Nat. Conf. Remotely Manned Syst.*, Pasadena, CA, 1973, pp. 43–61.

[10] J. W. Hill and A. J. Sword, "Studies to design and develope improved remote manipulator systems," in *Proc. 1st Nat. Conf. Remotely Manned Syst.*, Pasadena, CA, 1973, pp. 351–368.

[11] A. K. Bejezy and J. K. S. Jr., "Controlling remote manipulators through kinesthetic coupling," in *Comput. Mech. Eng.*, vol. 2, no. 1, 1983, pp. 48–60.

[12] A. M. Sabatini, M. Bergamasco, and P. Dario, "Force feedback-based telemicromanipulation for robot surgery on soft tissues," in *Proceedings of the Annual International Conference of the IEEE Engineering in Engineering in Medicine and Biology Society*, vol. 3, Nov. 1989, pp. 890–891.

[13] D. Kriegman, D. Siegel, S. Narasimhan, J. Hollerbach, and G. Gerpheide, "Computational architecture for the utah/mit hand," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 2, Mar. 1985, pp. 918–924.

[14] J. P. Merlet, "Force-feedback control of parallel manipulators," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, Apr. 1988, pp. 1484–1489.

[15] M. Ching, "A five-bar-linkage force reflecting interface for a virtual reality system via implicit force control," Master's thesis, University of Waterloo, Waterloo, ON, Canada, 1996.

[16] Y. Hirata and M. Sato, "3-dimensional interface device for virtual work space," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, July 1992, pp. 889–896.

[17] SensAble Technologies, Inc. (2005) Phantom devices. SensAble Technologies, Inc. 15 Constitution Way, Woburn, MA 01801 USA. [Online]. Available: http://www.sensable.com/products/phantom_ghost/phantom.asp

[18] E. Acosta, B. Stephens, B. Temkin, T. M. Krummel, P. J. Gorman, J. A. Griswold, and S. A. Deeb, "Development of a haptic virtual environment," in *Proceedings. 12th IEEE Symposium on Computer-Based Medical Systems*, June 1999, pp. 35–39.

[19] M. McLaughlin, G. Sukhatme, P. Wei, Z. Weirong, and J. Parks, "Performance and co-presence in heterogeneous haptic collaboration," in *Proceedings 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2003, pp. 285–291.

[20] S. Wang and M. A. Srinivasan, "The role of torque in haptic perception of object location in virtual environments," in *Proceedings. 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2003, pp. 302–309.

[21] G. Upperman, A. Suzuki, and M. O'Malley, "Comparison of human haptic size discrimination performance in simulated environments with varying levels of force and stiffness," in *Proceedings 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2004, pp. 169 – 175.

[22] F. Miyazaki, S. Matsubayashi, T. Yoshimi, and S. Arimoto, "A new control methodology toward advanced teleoperation of master-slave robot systems," in *Proceedings 1986 IEEE International Conference on Robotics and Automation*, vol. 3, 1986, pp. 997–1002.

[23] J. N. Anderson and M. E. Pittelkau, "Reflex-action position control of robots: comparison with impedance control," in *Proceedings Southeastcon IEEE Conference*, Apr. 1988, pp. 408–412.

[24] T. E. Alberts and D. I. Soloway, "Force control of a multi-arm robot system," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 3, Apr. 1988, pp. 1490–1496.

[25] S. Komada and K. Ohnishi, “Force feedback control of robot manipulator by the acceleration tracing orientation method,” in *IEEE Transactions on Industrial Electronics*, Feb. 1990, pp. 6–12.

[26] L. Ni and D. W. L. Wang, “A gain-switching control scheme for position-error-based force-reflecting teleoperation,” in *Proceedings 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2002, pp. 239–246.

[27] N. Hogan, “Impedance control: An approach to manipulation,” in *ASME J. Dyn. Sys. Meas. contr.*, 1985, pp. 8–15.

[28] M. W. Spong and M. Vidyasagar, *Robot Dynamics and Control*. Canada: John Wiley and Sons, Inc., 1989.

[29] T. Suzuki, “Impedance force control of free-joint manipulators with one motor under gravity,” in *IEEE/RSJ International Conference onIntelligent Robots and System*, Oct. 2002, pp. 2274–2279.

[30] D. Matko, R. Kamnik, and T. Bajd, “Adaptive impedance force control of an industrial manipulator,” in *Proceedings of the IEEE International Symposium on Industrial Electronics*, vol. 1, July 1999, pp. 129–133.

[31] J. Seul, T. C. Hsia, and R. G. Bonitz, “On robust impedance force control of robot manipulators,” in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 3, Apr. 1997, pp. 2057–2062.

[32] M. H. Raibert and J. J. Craig, “Hybrid position/force control of manipulators,” in *ASME J. Dyn. Sys., Meas. Contr*, vol. 102, 1981, pp. 126–133.

[33] D. Wang, K. Tuer, L. Ni, and P. Porciello, “Conducting a real-time remote handshake with haptics,” in *Proceedings 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2004, p. 291.

[34] T. H. Massie, “Design of a three degree of freedom force-reflecting haptic interface,” Master’s thesis, MIT, Cambridge, MA, USA, 1993.

[35] V. Hayward, P. Gregorio, O. Astley, S. Greenish, M. Doyon, L. Lessard, J. McDougall, I. Sinclair, S. Boelen, X. Chen, J.-P. Demers, J. Poulin, I. Benguigui, N. Almey, B. Makuc, and X. Zhang, "Freedom-7: A high fidelity seven axis haptic device with application to surgical training," *Lecture Notes in Control and Information Science 232*, pp. 445–456, 1998.

[36] C. Trautman and D. Wang, "Experimental h∞ control of a single flexible link with a shoulder joint," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 1, May 1995, pp. 1235–1241.

[37] K. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," in *IEEE Transactions on Neural Networks.*, vol. 1, No. 1, 1990.

[38] S. R. Chu, R. Shoureshi, and M. Tenorio, "Neural networks for system identification," in *IEEE Control Systems Magazine.*, 1990.

[39] R. Ahmed, "Identification of nonlinear dynamic systems using a rapid neural network," in *The 27th Annual Conference of the IEEE Industrial Electronics Society, 2001. IECON '01.*, vol. 3, Nov. 2001, pp. 1734 – 1739.

[40] M. O. Efe, O. Kaynak, and I. J. Rudas, "A novel computationally intelligent architecture for identification and control of nonlinear systems," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 3, May 1999, pp. 2073 – 2077.

[41] G. Yang and J. Meng, "Nonlinear identification and control using a generalized fuzzy neural network," in *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 2, Dec. 2002, pp. 1363 – 1368.

[42] K. Chon and R. J. Cohen, "Linear and nonlinear ARMA model parameter estimation using an artificial neural network," in *IEEE Transactions on Biomedical Engineering.*, vol. 44, No. 3, 1997, pp. 168 – 174.

[43] D. R. Madill, "Modelling and control of a haptic interface: A mechatronics approach," Ph.D. dissertation, University of Waterloo, Waterloo, ON, Canada, 1998.

[44] B. Bluethmann, S. Ananthakrishnan, J. Scheerer, T. Faddis, and R. Greenway, "Experiments in dexterous hybrid force and position control of a master/slave electrohydraulic manipulator," in *Proc. IEEE/RSJ International Conference on Human Robot Interaction and Cooperative Robots.*, vol. 3, No. 3, Aug. 1995, pp. 27–32.

[45] E. Wilson and S. M. Rock, "Gradient-based parameter optimization for systems containing discrete-valued functions," in *International Journal of Robust and Nonlinear Control*, May 2002, pp. 1009–1028.

[46] M. H. Mickle and T. W. Sze, *Optimization in Systems Engineering.* Pennsylvania, USA: Intext Educational Publishers, 1972.

[47] The Math Works Inc. (2003, Aug.) Matlab help: Optimization toolbox/standard algorithms/constrained optimization/. The Math Works, Inc. Natick, Massachusetts, USA.

[48] J. E. Colgate, P. E. Grafing, M. C. Stanley, and G. Schenkel, "Implementation of stiff virtual walls in force-reflecting interfaces," in *IEEE Annual International Symposium on Virtual Reality*, Sept. 1993, pp. 202–208.

[49] D. R. Madill, D. W. L. Wang, and M. C. Ching, "A nonlinear observer for minimizing quantization effects in virtual walls," in *Proc. ASME Dyn. Syst. Control Div.*, vol. 67, 1999, pp. 335–343.

[50] M. Goldfarb and J. Wang, "Passive stiffness simulation with rate-independent hysteresis," in *Proc. ASME Dyn. Syst. Control Div.*, vol. 67, 1999, pp. 345–350.

[51] J. E. Colgate and G. G. Schenkel, "Passivity of a class of sampled-data systems: Application to haptic interfaces," in *J. Robot. Syst.*, vol. 14, no. 1, 1997, pp. 37–47.

[52] J. P. Kim and J. Ryu, "Energy bounding algorithm based on passivity theorem for stable haptic interaction control," in *Proceedings. 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Mar. 2004, pp. 351 – 357.

[53] L. A. Zadeh, “Fuzzy logic,” *Computer*, vol. 21, Issue 4, pp. 83–93, Apr. 1988.

[54] H. S. Zadeh, “Constrained multi-objective optimization of a fuzzy logic controller-applications in a non-linear system and a goal-seeking rover,” in *Proceedings of 2003 IEEE Conference on Control Applications*, vol. 1, June 2003, pp. 240–244.

[55] S. Galichet and L. Foulloy, “Fuzzy controllers: synthesis and equivalences,” in *IEEE Transactions on Fuzzy Systems*, vol. 3, May 1995, pp. 140–148.

[56] J. C. Ryu and C. K. Park, “A hybrid-fuzzy controller design for the control of a flexible finger,” in *38th Annual Conference Proceedings of the SICE*, July 1999, pp. 1233 – 1238.

[57] H. O. Wang, K. Tanaka, and M. F. Griffin, “An approach to fuzzy control of nonlinear systems: stability and design issues,” in *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 1, Feb. 1996, pp. 14 – 23.

[58] J. Parthiban. (1996) Surprise 96 fuzzy logic and its uses article 2 fuzzy logic introduction. [Online]. Available: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol2/jp6/article2.html

[59] The Math Works Inc. (2003, Aug.) Matlab help: Fuzzy logic toolbox/tutorial/foundations of fuzzy logic/membership functions/. The Math Works, Inc. Natick, Massachusetts, USA.

[60] E. Kubica, D. Madill, and D. Wang, “Designing stable mimo fuzzy controllers,” in *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, vol. 35, no. 2, Apr. 2005, pp. 372 – 380.

[61] 20-sim. (2004) Static friction phenomena. 20-sim. Drienerlolaan 5 HO-8266 7522 NB Enschede Netherlands. [Online]. Available: http://www.20sim.com/webhelp4/library/iconic_diagrams/Mechanical/Friction/Static_Friction_Models.htm

[62] C. Turner, “Development of an internet visual telepresence system,” Master’s thesis, University of Waterloo, Waterloo, ON, Canada, 2000.

[63] P. Dahl, "A solid friction model," Aerospace Corp., El Segundo, CA, Tech. Rep.

[64] B. Armstrong-Helouvry, *Control of Machines with Friction.* Boston, MA, USA: Kluwer, 1991.

[65] D. Kostić, R. Hensen, B. de Jager, and M. Steinbuch, "Modelling and identification of an rrr-robot," in *Proc. IEEE Conf. Decision and Control.*, Orlando, USA, Dec. 2001, pp. 1144–1149.

[66] Y. Zhu and P. R. Pagilla, "Static and dynamic friction compensation in trajectory tracking control of robots," in *Proc. IEEE Int. Conf. Robotics and Automation.*, Washington, USA, May 2002, pp. 2644–2649.

[67] C. de Wit, H. Olsson, and P. L. K.J. Astrom, "A new model for control of systems with friction," in *IEEE Transactions on Automatic Control*, vol. 40, No. 3, Mar. 1995, pp. 419–425.

[68] W. Garcia-Gabin, E. F. Camacho, and D. Zambrano, "Improving gpc tuning for non-minimum phase systems," in *Controlo'2000: 4th Portugese Conference on Automatic Control*, 2000, pp. 66–71.

[69] MPB Technologies Inc., "Freedom 6s force feedback hand controller user manual," MPB Technologies Inc., Tech. Rep., 2001.

[70] tcaep.co.uk. (2005) Science equation beat frequency. Institute of Physics. [Online]. Available: http://www.20sim.com/webhelp4/library/iconic_diagrams/Mechanical/Friction/Static_Friction_Models.htm

[71] I. Sinclair, *Mechanical Assembly Freedom 6S Hand Controller*, MPB Technologies Inc., May 2000.