<u>İSTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY</u>

FORCE CONTROL OF ROBOTIC MANIPULATORS IN COOPERATION

M.Sc. THESIS

Mateusz SZCZESIAK

Department of Mechatronic Engineering

Mechatronic Engineering Program

JUNE 2016

<u>İSTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY</u>

FORCE CONTROL OF ROBOTIC MANIPULATORS IN COOPERATION

M.Sc. THESIS

Mateusz SZCZESIAK
(518141009)

Department of Mechatronic Engineering

Mechatronic Engineering Program

Thesis Advisor: Assoc. Prof. Dr. Zeki Yağız BAYRAKTAROĞLU

JUNE 2016

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

İŞBIRLIKÇI ROBOT MANIPÜLATÖRLERIN KUVVET KONTROLÜ

YÜKSEK LİSANS TEZİ

Mateusz SZCZESIAK
(518141009)

Mekatronik Mühendisliği Anabilim Dalı

Mekatronik Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Zeki Yağız BAYRAKTAROĞLU

HAZIRAN 2016

Mateusz Szczesiak, a M.Sc. student of İTU Graduate School of Science Engineering and Technology student ID 518141009, successfully defended the thesis/dissertation entitled "**FORCE CONTROL OF ROBOTIC MANIPULATORS IN COOPERATION**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Doç. Dr. Zeki Yağız BAYRAKTAROĞLU**
İstanbul Technical University

....

Jury Members : **Prof. Dr. Ata MUĞAN**
İstanbul Technical University

....

Yrd. Doç. Dr. Dilek Bilgin TÜKEL
Doğuş University

....

Date of Submission : 02 May 2016
Date of Defense : 10 June 2016

To my wife Irem,

FOREWORD

I would like to express my deep appreciation and thanks for my advisor. This work is supported by ITU Institute of Science and Technology.

April 2016 Mateusz SZCZESIAK

TABLE OF CONTENTS

Page

ABBREVIATIONS

CoG	**:** Center of Gravity
IDE	**:** Integrated Development Environment
ITU	**:** İstanbul Teknik Üniversitesi
LLI	**:** Low Level Robotic Control Interface
LQE	**:** Linear Quadratic Estimator

SYMBOLS

a,b,c	: Scalars
$\boldsymbol{a,b,c}$ or $\boldsymbol{A,B,C}$	: Vectors (bold)
A,B,C	: Matrix operators
A^T	: Matrix transpose
A^{-1}	: Matrix inverse
A^+	: Matrix pseudoinverse
A_{12}	: Matrix elements
$f(), g()$	: Functions
$\hat{}$	: Skew-symmetric matrix
t	: Time
$\boldsymbol{\omega}$	: Angular velocity vector
$\boldsymbol{v}$	: Linear velocity vector
$\boldsymbol{\Gamma}$	: Joint torque vector
$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$	: Generalized coordinate vectors
$C(\boldsymbol{q}, \dot{\boldsymbol{q}})$	: Coriolis and centrifugal matrix
$A(\boldsymbol{q})$	: Inertia matrix
$\boldsymbol{Q}(\boldsymbol{q})$	: Gravity torque vector
$\boldsymbol{S}(\boldsymbol{q})$	: Spring torque vector
$\boldsymbol{\mathcal{T}}(\dot{\boldsymbol{q}})$	: Friction torque vector
$\boldsymbol{f}$	: Torque vector due to external loads
$\boldsymbol{F_f}$	: External, feedback force vector
$\boldsymbol{M_f}$	: External, feedback moment vector
J_e	: Jacobian of the end effector
J_{lc}	: Jacobian of the load cell
$K_p = k_p I$	: Gain Matrix K_p and the gain coefficient k_p

LIST OF TABLES

Page

LIST OF FIGURES

Page

FORCE CONTROL OF ROBOTIC MANIPULATORS IN COOPERATION

SUMMARY

Cooperation of robotic manipulators has been an important field of study in robotics. Where initially, robotic design, study, and performance was targeted toward accomplishing simple tasks, more recent emphasis is placed on accomplishing complex objectives that often require active cooperation of multiple robotic units. With increased presence of robotic devices, effective cooperation opens possibility for greater automation of various tasks. With proliferation of affordable actuators, controllers and sensors, more and more robots make their ways not only onto factory floors and production lines, but also into our homes.

The effective application of robotic cooperation leads right to that of human-robot cooperation. Here, robots can be used to assist human directly. A person acting as a master, as a leader, or even as a collaborator, can work side by side with a robot.

In the following pages, cooperation between master and a slave is explored. Master unit is tasked with executing predefined motion while carrying a load. The slave unit assists by sharing the weight of the item while following motion path projected by the master. Both units are separate and no data exchange takes place between two robots. The compliance of the slave manipulator is based strictly on the load feedback from the wrist mounted load cell. Load feedback is also used by the master to compensate for the reaction loads arising during cooperation.

The ultimate goal of the thesis was to implement above scheme on the actual hardware available at the Mechatronic Education and Research Center. Three Stäubli Rx160/Rx160L robotic units were available for testing. These are six axis robotic arms. All joints are rotary and are driven by brushless motors coupled with resolvers, and contain integrated parking brakes. Two of the robots are equipped with six-degree force/torque cells allowing for load feedback necessary for the implementation of the force control.

Initial research included literature survey. Previous work concerning cooperation of the Master-Slave couple, dating as far back as early 1970s, was analyzed. Theoretical background necessary for a proper description of the system was studied. While the computed torque control, and the direct force control structures are both well-developed subjects, the actual implementation is more difficult.

Modeling and simulation of the robot kinematics, dynamic and control was performed as the preliminary step for the implementation on the actual hardware. Model of the master robot was based on the inverse dynamics, computed torque control with PD position and orientation control loops. Slave unit uses direct force/torque PD control. Both use Newton-Euler algorithm for the calculation of the required joint torques. Matlab Simulink (R2014a) was used to simulate individual control structures for the master, the slave, as well as for the combined system.

Coupling between master and slave manipulators was done using ideal model of a linear and of torsional springs. Linear and angular displacements between both end

effectors were tracked and used to generate loads. Path planning for the master robot was based on via-point trajectory generation in task space using quintic splines. Ultimately, simulation studies allowed verification of the control algorithms and as a way to familiarize one with complete control structure. Furthermore, initial estimation of the required gains was achieved.

Experimental work began once satisfactory performance of the simulated system was obtained. Matlab code was rewritten into C language to be used in the controller. Using test software, control algorithm was coded and tested before uploading it onto the robot. This was done to ensure safety. While the opening performance left a lot to be desired, subsequent developments brought increased reliability.

Necessity of good characterization of the system was recognized early. As the dynamic performance of the feedback linearization strategy relies heavily on cancelation of nonlinear terms, good friction model, among others, was necessary. While positional control could achieve relatively good results, orientation control was virtually nonexistent without proper friction model. Due to very low inertia of the fifth and the sixth links friction forces dominated. Good friction model was then paramount to even actuate last two joints. Friction model, borrowed from the work of Waiboer (2007) has been adjusted for better performance. Additionally, characterization of robot's torsion spring, integral component of the second link, was done as discrepancies arose between available model and actual hardware.

Gain tuning was completed using Ziegler – Nichols method for a PD controller. Additional, manual tuning was done afterwards. As a way to further improve performance, integral control action was added. It has been noted, that both control loops, position and orientation control, are often in direct conflict. As such, there is often a tradeoff in terms of achievable positional and orientation control. Increasing one leads directly to deterioration of the other.

Force control of the slave manipulator was next to be implemented. While the same Newton-Euler algorithm was used, friction model was no longer crucial. Early experiments failed due to pre-preprogrammed tool frames in the load cell controller. Once corrected, performance improved dramatically. Further gain tuning achieved good response. Gripper weight at any orientations was compensated, and a separate algorithm was used to calculate weight of the held objects using force feedback form the load cell.

Two control modes for the slave were realized. In addition to pure compliance control, compliance control with orientation control was added. In the first mode, end effector would actively change orientation to eliminate force/torque error between desired and feedback loads. In the second mode, orientation loop was active and would attempt to hold initial orientation of the gripper while moving in response to external loads.

First cooperation experiments were performed with a human in place of a Master. Later, coupling between master and slave was accomplished by linking them with a compression spring. Experimental results are given.

Following pages' detail work outlined above. It is shown that the Master-Slave cooperation allows for simple yet effective coupling of two robotic manipulators. As both arms execute separate routines and no information sharing takes place, more slave units can be coupled to accomplish a given task. Furthermore, it is shown that the master unit can easily be replaced by a human being further extending applicability of the presented work.

İŞBİRLİKÇİ ROBOT MANİPÜLATÖRLERİN KUVVET KONTROLÜ

ÖZET

Robot kollarının işbirlikçi kontrolü robotbilimindeki önemli çalışma alanlarından biridir. Önceleri robotların kontrolcü tasarımı ve performans değerlendirmeleri tek bir robotun gerçekleştirebileceği basit görevlerin başarılmasına odaklanıyorken, günümüzde çoklu robot sistemlerinin aktif işbirliğini gerektiren karmaşık görevlerin yapılabilmesine önem verilmektedir. Robot kullanımının yaygınlaşması ve birden çok robotun işbirlikçi kontrolü yaklaşımlarıyla çeşitli görevlerin otomasyonu mümkün olabilmektedir. Eyleyici, algılayıcı ve kontrolcü maliyetlerinin giderek azalmasıyla artık robotlar sadece sanayi üretiminde değil, evlerimizde de kendilerine daha yaygın bir şekilde yer edinebilmektedir.

Robotların işbirlikçi kontrolünün verimli bir şekilde uygulanabilmesi insan-robot işbirliğine de imkan tanımaktadır. Bu çerçevede robotların insanlarla doğrudan fiziksel etkileşime girerek çeşitli görevlerde yardım etmesi mümkündür. Çeşitli fonksiyonları üstlenebilecek insan operatörler ile robotlar yan yana çalışabilmektedir.

Bu çalışmada, efendi/köle (master/slave) etkileşimi içindeki robotların işbirlikçi kontrolü irdelenmektedir. Efendi robot, yük taşırken önceden belirlenen referans hareketleri gerçekleştirmekle görevlidir. Köle robotun görevi ise yükün ağırlığını paylaşarak efendi robotun hareketlerini takip etmektedir. Servo kontrolleri birbirinden bağımsız olan iki robot arasında hiçbir veri paylaşımı gerçekleşmemektedir. Köle robotun uyumu hareketi (compliant motion) kolucuna yerleştirilmiş kuvvet/moment algılayıcısı kullanılarak sağlanmaktadır. Kuvvet geribeslemeleri ile, işbirlikçi çalışma esnasında iki robot arasında ortaya çıkan etkileşim kuvvetleri de kompanse edilebilmektedir.

Bu çalışmanın amacı, Mekatronik Eğitim ve Araştırma Merkezinde mevcut robotlar kullanılarak yukarda bahsedilen kontrolcülerin uygulanmasıdır. Deneysel çalışmalar için Merkezde üç adet 6 serbestlik dereceli Stäubli Rx160/Rx160L robot kol mevcuttur. Robotların tüm eklemleri dönel olup, fırçasız motorlarla tahrik edilmektedir. Robotların ikisinde 6 eksenli kuvvet/moment algılayıcıları mevcuttur.

Literatür araştırmasında, efendi/köle düzenindeki robotların işbirlikçi kontrolü ile ilgili 1970li senerlerden bu yana yayınlanan çalışmalar incelenmiştir. İşbirlikçi bir robot sisteminin uygun bir biçimde tanımlanabilmesi için gerekli teorik arkaplan araştırılmıştır. Bu çalışmada deneysel olarak uygulanan kontrol şemaları teorik olarak iyi bilinen yöntemler olsa da uygulamalarda karşılaşılan Pratik zorluklar mevcuttur.

İlk aşamada, robotların kinematik ve dinamik modelleri türetilmiş, control sistemleri tasarlanmış ve bilgisayarda dinamik benzetimleri gerçekleştirilmiştir. Efendi robotun hareket kontrolü ters dinamik modeli temel alan hesaplanmış moment yöntemiyle (computed torque control) gerçekleştirilmiştir. Köle robota ise doğrudan kuvvet kontrolü (explicit force control) uygulanmıştır. Robotların tekli ve efendi/köle düzenindeki dinamik davranışlarının benzetimi için Matlab Simulink yazılımı kullanılmıştır.

Efendi ve köle robotlar arasındaki eşleşme (coupling) ideal yay modelleri kullanılarak sağlanmıştır. Robot kol uçlarının birbirine göre yer değiştirmesi ölçülerek ortaya çıkacak etkileşim kuvvetleri hesaplanmıştır. Efendi robotun yörünge planlaması görev uzayında (task space) beşinci mertebe *spline* eğriler kullanılarak yapılmıştır. Dinamik benzetim çalışmaları ile kontrol algoritmaları test edilmiş ve gerçek sistemde uygulanacak kontrolcü kazançlarının ilk değerleri elde edilebilmiştir.

Benzetim çalışmalarının ardından deneysel çalışmalara başlanmıştır. Matlab kodları robot kontrolcülerinde kullanılacak şekilde C dilinde programlanmıştır. İlk deneylerle beraber, kabul edilebilir hareket kontrolü performansları için olabilediğince gerçekçi bir sürtünme modeline ihtiyaç duyulduğu anlaşılmıştır. Robotların özellikle bilek eklemlerindeki beşinci ve altıncı serbestlik derecelerinin dinamik davranışları bütünüyle sürtünme etkisi tarafından domine edilmektedir. Bu çalışmada, Wailboer'in (2007) tarafından önerilen sürtünme modeli esas alınmış ve performans iyileştirmesi için bazı geliştirmeler yapılmıştır. Ayrıca, robotun ikinci serbestlik derecesinde mevcut dengeleme yayının modellemesi de deneysel olarak gerçekleştirilmiştir.

PID kontrolcü kazançları önce Ziegler-Nichols yöntemi ile ayarlanmış, daha sonra hassas ayar değerleri deneysel olarak elde edilmiştir. Robot kol uçlarının konum ve açısal konum kontrollerinin birbiriyle çatıştığı gözlemlenmiştir. Konum ve açısal konumların kontrolünde kabul edilebilir performanslar elde edebilmek için her iki değişken arasında ödünleşime (trade-off) ihtiyaç duyulmaktadır.

Bir sonraki aşamada köle robotun kuvvet kontrolü test edilmiştir. Uç eyleyicinin (gripper) yol açtığı ek yükler görev uzayında hesaplanmış ve eklem uzayına dönüştürülerek çevrim içinde kompanse edilmiştir. Ayrıca, manipüle edilecek yüklerin kütle değerleri ve kütle merkezi konumlarını hesaplayan bir algoritma yazılarak uygulanmıştır.

Köle robot için iki farklı control şeması uygulanmıştır. Klasik uyum kontrolüne (compliance control) ek olarak, robot kol ucunun açısal konumunun ayrıca control edildiği alternative bir kapalı çevrim daha uygulanmıştır. Birinci şemada robot, kol ucunda ölçülen kuvvet/moment hatasına karşılık hareket ederek anlık durumunu (configuration) değiştirebilmektedir. İkinci şemada aktif bir açısal konum kontrolcüsü ile dış kuvvetlere maruz kalan robotun kol ucu açısal konumu istenen sabit bir referans durumunda tutulabilmektedir.

İlk işbirlikçi kontrol deneylerinde, efendi robot görevi bir insan operator tarafından gerçekleştirilmiştir. İlerleyen aşamalarda, efendi ve köle robotların kol uçları birbirine mekanik yay elemanıyla bağlanmış ve iki robotun işbirlikçi kontrolü gerçekleştirilmiştir.

Bu çalışmada sunulan deneysel sonuçlarla, efendi-köle düzeninde çalışan iki robotun işbirlikçi kontrolü ile basit bir biçimde eşleştirilebileceği (coupling) gösterilmektedir. Her iki robot birbirinden bağımsız çevrimlerle kontrol edildiğinden ve robotlar arasında veri paylaşımı yapılmadığından, verilen bir görevi yapmak üzere, bir efendi robot ile daha fazla sayıda köle robot eşleştirmesi mümkün olabilecektir. Ayrıca efendi robotun, mevcut çalışmanın uygulanabilirliğini genişletecek şekilde bir insan operatörü ile kolaylıkla yer değiştirebileceği gösterilmiştir.

1. INTRODUCTION

1.1. Motivation

Cooperative robotics involves cooperation of a number of robots in accomplishing a common task. Working together, robots can accomplish missions that may be too difficult or impossible to perform by a single unit. With greater proliferation of robotics, ability to exploit robot cooperation promises to expand their usage and application in numerous fields including industry, medical, space and defense.

There are numerous applications exemplifying benefits of multiple robots working together. In space, multiple robotic manipulators could be used to intercept, dock and service satellites in orbit. In an operating room, robotic arms can cooperate in performing surgeries. Assembly floor robots working together can perform faster and successfully compete complex industrial processes.

Another interesting area of cooperative robotics considers human to robot cooperation. Robotic platforms can be used to assist human beings. For example, guided by a human operator, robot may be used to take a proportionally greater load required to carry an object. A human would act as master/leader by imposing a path necessary to position the object at the right location. Similarly, a medical robot may respond to force/torque input from a person and assist disabled patient during motion by sharing the loads associated with the human locomotion process.

Examples included above represent just a small portion of what is possible with the robotic cooperation. Some of the application have already materialized while many more are on the drawing board of researches and designers all over the world.

1.2. Problem Definition

This research considers design, implementation and testing of computed torque control in task space with force feedback for the cooperation of robotic manipulators. One robot designated as the master executes preplanned motion using position and orientation control. The second robot, designated as slave, follow motion of the master based on force feedback alone. Both robots are to complete a task of transporting a load that would normally be difficult to handle by a single arm. All elements required to accomplish this task, including kinematics, dynamic, path planning and control must be simulated and then implemented on the actual hardware.

1.3. Document Outline

The following research begins with literature survey of the state of the art in the field of cooperative robotics. Short overview of cooperative robotics, with selected primary references, is presented. The description of the hardware follows next. Complete specification and constraints on the robotic arms are included. This is then followed by the theoretical background required for the implementation of the robot control. This include robot geometric model, kinematic, dynamics and the control theory itself. Various other topics are also considered. Chapter four presents simulations of the mathematical models for the master manipulator and the slave manipulator alike. This is then followed by the experimental work used to validate the theory. All work associated with the implementation of the robot control is presented. Experimental results close the chapter. Lastly, conclusions and recommendations are offered.

1.4. Literature Survey

Research work in the cooperation of robotic manipulators can be dated back to 1970s. These early papers cover preliminary concepts of cooperation of multiple robotic arms and include discussions of compliance, task control, and force control. Nakano, Ozaki, Ishida and Kato (1974) are the first to propose master-slave cooperation between two robots and to recognize importance of force control. In their paper, master robot follows a reference trajectory using position control. Slave robot follows motion imposed by the master using force control which generates correcting path. Both mechanical arms cooperate to manipulate and transfer an object.

The natural progression of the master-slave pair takes place with the publication by Luh and Zheng (1987). Here cooperation between leader and a follower is explored using a closed kinematic chain required to satisfy a set of holonomic constraints at every instant of time. Knowledge of the leader motion and the application of the constraints involving position, orientation, joint velocities and accelerations, allowed determination of the corresponding variables as well at the joint torques required by the follower. Furthermore, this allowed for the modification of the motion of the leader in the real, for example, to avoid a collision. This scheme however, would not permit online role switching between cooperating arms. Moreover, the above scheme was characterized by the excessive compliance required to guarantee smooth motion of the follower.

Development of non-master/slave control was the next chapter in the evolution of the robotic cooperation. This involved treating the cooperation as a system where task object variables are treated as references for the control of the individual arms. Additionally, interaction forces between the task object and the manipulators are used as feedbacks to be controlled directly.

Hybrid control is covered in the paper by Raibert and Craig (1981). This non-master/slave approach separates control into two parallel loops. First loop is responsible for position control while the second one implements force control. Output of each goes through selection matrix which prioritize one over the other along certain directions. Contributions from both are then summed. The separation naturally divides performed task into that of motion and force interaction but relies on proper choice of the selection matrix. Prevalence of the force loop can be enforcing in case of a contact with the environment.

Application of feedback linearization in cooperating robots can be found in paper by Tarn, Bejczy and Yun (1988). Complicated nonlinear problem is transformed into decoupled linear system. Task control is implemented. Two cases are modeled and compared, single system made up of a closed kinematic chain where dynamics of individual arms are not independent, and a case with separate arms with interactive forces and moments and end effector mounted load cells. It is observed that both cases may be used in different applications. While closed chain leads itself toward manipulators linked by a relatively rigid object, direct force control approach is preferable if the link is more elastic in nature.

Schneider and Cannon Jr. (1990) discuss robotic cooperation using impedance control first proposed by Hogan (1985). Here, control is exercised on the manipulated object itself. Instead of controlling individual variables such as position, velocity or force, a relation between them is enforced. In simplest practical application, it takes form of a second order system comprised of a mass, a dumper and a spring with external force action as an input. More complicated terms, for example, nonlinear potential forces, can be easily incorporated allowing obstacle avoidance. No explicit closing of the kinematic chain is required. End effector is treated as an impedance while the environment as an admittance. This allows for much better interaction with object and environment.

Whitsell and Artemiadis (2015) propose an interesting application of the impedance control. Using adaptive algorithm, role of a leader and that follower are dynamically switched. Impedance is changed depending on the present role of the manipulator. If the leader goes off tract, which leads to increase in interaction forces due to greater resistance from the follower, roles are switched.

Teleoperation has been an important application of robotic cooperation. An outside operator can effectively take the role of master, or that of a leader. Working together with multiple slaves, manipulation tasks can be accomplished from a distance. Moreover, haptic devices can be used directly as virtual master manipulators. A good overview of the area is found in paper by Cui, Tosunoglu, Roberts, Moore, and Reporter (2003).

Synchronous control relies on synchronization of motion of individual manipulators while each one tracks its desired trajectory. This is accomplished by simultaneously guarantying convergence of the differential position error between manipulators to zero (Sun & Mills, 2002). Control approach utilizes a form of a kinematic relation while relaxing requirement on explicit force control between individual robots. This strategy improves desired trajectory tracking between cooperating manipulators. Force synchronization for multi robot system has been proposed and simulated by Liu, Zhao, and Xiang (2014). Force synchronization has been defined as convergence, at the same rate, of contact force errors for multiple manipulators. This approach ensures that no excessive internal forces are generated in the manipulated object. Cross-coupling errors of force control between each manipulator pair has been synthesized and included in the control. Comparison with independent control of multiple manipulators was conducted. Force synchronization strategy proved superior in achieving convergence.

Adaptive control has been successfully employed in the cooperative robotics as a solution to parametric uncertainties of the dynamic model. Fuzzy adaptive controller by Gueaieb, Karray, and Al-Sharhan extends this approach to handle unstructured external disturbances (2004). There is no prior knowledge of the system dynamics which is assessed and approximated online during object handling and manipulation. Internal forces and tracking error converge to zero at the expense of increased computational cost associated with the complexity of the fuzzy logic engine.

Lee and Jung (2011) demonstrated Master-Slave cooperation in transporting an object between two balancing robots. While the master followed a desired path, slave robot applied necessary force required to maintain an object between both units. Impedance force control was utilized in addition to position, orientation and balance controls already required for a two wheeled balancing robots.

More recent application of a Master-Slave manipulator pair can be found in paper by Cheng, Wang, and Ma (2015). Force controller for a two finger robot is proposed utilizing standard PID control for the master and predictive control based on the grey system theory for the slave. This is done to harness past, present and future force information to pre-compensate force errors. Implementing this predictive control structure was done to better handle dynamically changing parameters of objects held by an industrial gripper.

An illustrative example of a human robot cooperation can be found in paper by Behnke and Stückler (2011). Humanoid robot is tasked with assisting a human in transferring a large object. Human directs motion of the item with assistance from the robot. Compliant arms allow quick response to movement of the manipulated object while mobile base of the robot restores nominal position of the arms. Compliance motion control has been applied to accomplish this task. Furthermore, real time vision has been employed for pose estimation. In addition, Lifting or setting down of the object could be perceived.

1.5. Hardware and Software

1.5.1. Stäubli Rx160 family manipulators

Mechatronic Education and Research Center at the Istanbul Technical University (ITU) Ayazağa campus offers access to three Stäubli RX160 series robotic manipulators. These are high precision, medium load, six degree industrial robotic arms. Two of the manipulators are models RX160 while the third one is model RX160L. Except for the length and mass of the link four, both models are identical. Figure 1.1 provides general overview of the Stäubli RX160 manipulator.

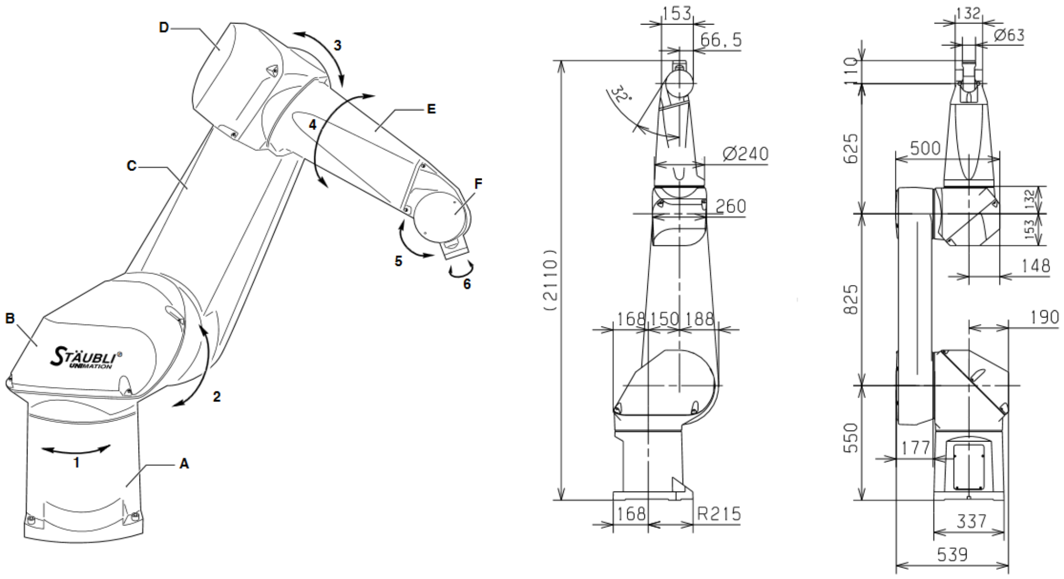


Figure 1.1 : Stäubli RX160 manipulator (Stäubli, Product Leaflet).

All joints of the manipulator are rotary and are driven by brushless motors with resolvers and brakes. Furthermore, two solenoid valves with connections are available for attachment of pneumatic gripper. Nominal load capacity for the RX160 and RX160L are 20kg and 14kg respectfully. Working temperature range is between +5°C to +40°C (Stäubli, Product Leaflet).

Additional parameters including joint working range, speed, and resolution are included in Table 1.1.

Table 1.1 : Stäubli RX160 amplitude, speed and resolution **(**Stäubli**)**.

Axis	1	2	3	4	5	6
Amplitude (°)	320	275	300	540	225	540
Working range (°)	± 160	± 137.5	± 150	± 270	+120/-105	270
Nominal speed (°)	165	150	190	295	260	440
Maximum speed (°/s)	200	200	255	315	390	870
Angular resolution (° , 10^-3)	0.042	0.042	0.054	0.062	0.12	0.17

Each manipulator is equipped with a pendant. Pendant allows for manual control of the robot and in addition, acts as a safety device enabling immediate power shutdown in case of an emergency.

Joint five and joint six of the manipulator are coupled. Rotation of joint five automatically moves joint six as per Figure 1.2. Torque commands sent to the manipulators are those on the output of each joint and the actual coupling is handled internally by low level control. Friction however, still requires proper modeling of the joint coupling. These effects may not be neglected.

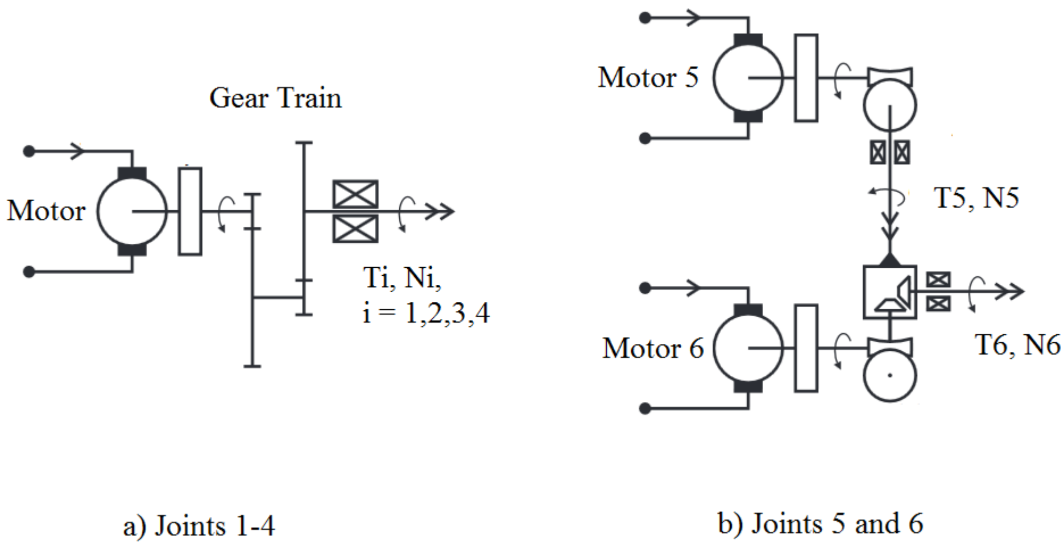


Figure 1.2 : Stäubli RX160 series joint system diagrams (Waiboer, 2007).

1.5.2. CS8C controller and low level interface

Low level interface of the robot can be accessed using LLI, Low Level Robotic Control Interface running on the CS8C controller. Serial connection allows for direct link with a PC. Build in commands permit access and modification of the low level parameters and functions. LLI package includes software development kit providing programming interface using c library. This allows development of control applications. Visual Studio 2010 solution is provided. Control subroutines are written using Microsoft IDE. Test module allows for extensive testing before package can be ported into Tornado IDE for final compilation. Once compiled, the package is uploaded into the controller for execution. Figure 1.3 provides overview of the controller/LLI architecture.

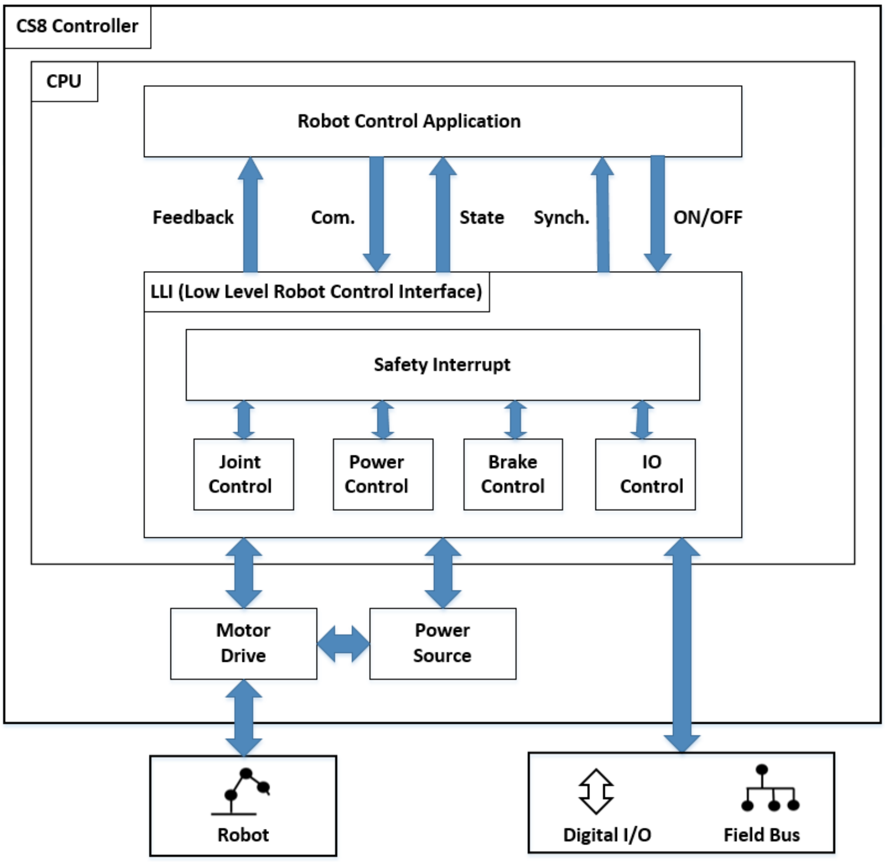


Figure 1.3 : Controller/LLI architecture.

The architecture of the software loop is outlined in Figure 1.4

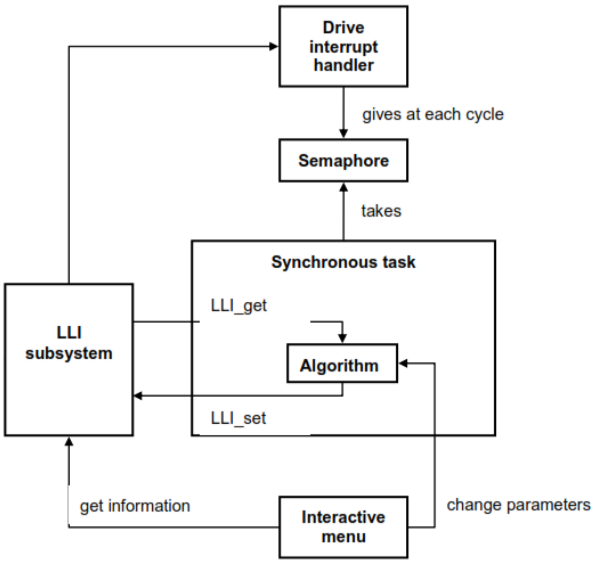


Figure 1.4 : Software architecture (Stäubli, 2012).

Evaluation of control parameters such as desired position, velocity, torque etc. takes place in the Algorithm block governed and timed by the Synchronous task. Access to low level functionality is provided by the LLI subsystem which calls Drive Interrupt Handler within context of hardware interrupt. This handler generates semaphores managing access. User input is available through Interactive menu.

1.5.3. ATI Delta six-axis force/torque transducer

Two of the manipulators are equipped with the ATI Delta model transducers. Delta series load cells are six axis force/torque transducers with silicon strain gauges. Both units are calibrated as per SI-660-60, see Table 1.2.

Table 1.2 : ATI Delta calibration specifications (ATI Industrial Automation).

Calibration	Sensing Range				Resolution			
	Fx,Fy	Fz	Tx,Ty	Tz	Fx,Fy	Fz	Tx,Ty	Tz
SI-165-15	165	495	15	15	1/32	1/16	1/528	1/528
SI-330-30	330	990	30	30	1/16	1/8	5/1333	5/1333
SI-660-60	660	1980	60	60	1/8	1/4	10/1333	10/1333

Values for Force are in (N), values for Torque are in (Nm)

2. SYSTEM MODELING

Following chapter presents mathematical theory used in describing a robotic system. This overview covers various concepts and included minimum amount of information required for the correct modeling of a physical system analyzed in the subsequent chapters.

2.1. Geometric Model

Direct geometric model of a robot provides location of the end effector as a function of joint parameters. It is a mapping from joint or configuration space to operational, or task space. Similarly, inverse geometric model gives mapping form task space back to joint space. In regard to the present work, configuration space is spanned by the six joint parameters or joint angles θ_i. These are necessary and sufficient to completely describe the location of the end effector. Likewise, operational space is defined as three dimensional Euclidean space $\mathbb{R}^3$ required to specify position of the end effector, and rotation group SO(3) to define end effector orientation.

In general, geometric description of a serial manipulator requires establishment of the fixed, base or world coordinate frame R_0 described by three parameters X_0, Y_0, Z_0. Additional, local coordinate frame R_i and corresponding parameters X_i, Y_i, Z_i are assigned to each link. All links are assumed perfectly rigid. As the individual links move, so do the corresponding local frames. Figure 3.1 illustrates location of base and local coordinate frames used throughout this work.

2.1.1. Forward geometric model

Accordingly, mathematical description of the robot`s end effector location, or its geometric model, is reduced to tracking of the local frames relative to the base frame. This is accomplished through rotational matrices R which relate orientation of individual frames to each other. Position of an object within a local frame can then be expressed in any other frame, sharing the same origin, through a series of transformations. For a serial robot with six revolute joins, this reduces to calculating composite rotational matrix R_{06} as per equation 2.1

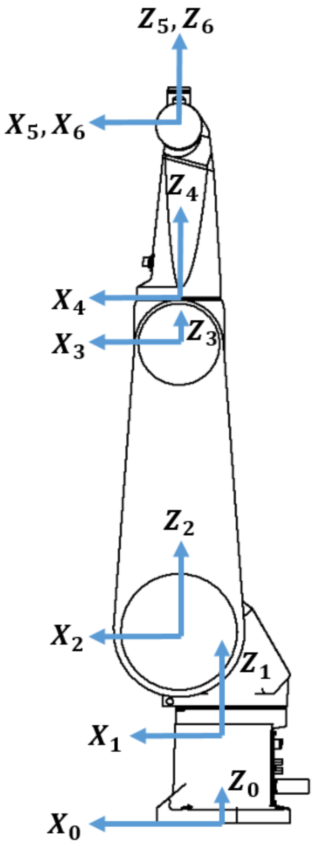


Figure 2.1 : Base and local coordinate frames with robot at home position.

$$R_{06} = R_{01}R_{12} \dots R_{56} \tag{2.1}$$

Individual rotation matrices are constructed based on the rotational axis of each joint. These include rotations about local Y for joint two, three and five, and about local Z for joints one, four and six. Individual rotation matrices are given in equation 2.2.

$$\begin{gathered} R_{ij} = \begin{pmatrix} \cos\theta_j & 0 & \sin\theta_j \\ 0 & 1 & 0 \\ -\sin\theta_j & 0 & \cos\theta_j \end{pmatrix} about\ Y_j \\ R_{ij} = \begin{pmatrix} \cos\theta_j & -\sin\theta_j & 0 \\ \sin\theta_j & \cos\theta_j & 0 \\ 0 & 0 & 1 \end{pmatrix} about\ Z_j \end{gathered} \tag{2.2}$$

Any local vector P_{jk} can be expressed in the base frame using the following equation.

$$\boldsymbol{P_{0k}} = R_{0j}\boldsymbol{P_{jk}} + \boldsymbol{P_{0j}} \tag{2.3}$$

Vector $\boldsymbol{P_{0j}}$, connecting origins of the world frame and the local frame j can be evaluated through recursion. Using local link lengths $\boldsymbol{l_{ij}}$, the final location of the end effector becomes:

$$\boldsymbol{P_{0e}} = R_{06}\boldsymbol{l_{6e}} + R_{05}\boldsymbol{l_{56}} + R_{04}\boldsymbol{l_{45}} + R_{03}\boldsymbol{l_{34}} + R_{02}\boldsymbol{l_{23}} + R_{01}\boldsymbol{l_{12}} + \boldsymbol{l_{01}} \tag{2.4}$$

2.1.2. Inverse geometric model

Inverse geometric model provides solution to reverse problem of determining joint configuration with known task position and orientation. It requires solving twelve nonlinear equations and presents a formidable problem. Furthermore, solution is not unique. In general, for a set or standard robot configuration, a geometric solution is employed. This involves decoupling of the first three rotation axes from the last three which comprise the wrist. End effector task position and thus position of the wrist constitutes input into the first problem. Using geometry, one can readily solve for the required joint angles assuming particular arm configuration. This provides R_{03}. Orientation of the wrist follows next. Using known orientation R_{06} and solving equation 2.5, provides set of equations in terms of three remaining joint angles.

$$R_{36} = {R_{03}}^T R_{06} \tag{2.5}$$

2.2. Description of Orientation

While mathematically convenient, rotation matrices are not the most practical way of expression orientation. Nine elements of a rotation matrix are not linearly independent. Three other methods for defining orientation were used in this work. These include: roll, pitch, yaw angles, axis/angle representation and quaternions.

2.2.1. Roll, Pitch, Yaw angles

Any rotational matrix R can be decomposed into successive rotations about base coordinate axes X_0, Y_0, Z_0. Rotation of ψ about X_0 followed by rotation of θ about Y_0 and finally rotation of ϕ about Z_0 gives following equality.

$$R = R_{z,\phi} R_{y,\theta} R_{x,\psi}$$
$$= \begin{pmatrix} C_\phi C_\theta & -S_\phi C_\psi + C_\phi S_\theta S_\psi & S_\phi C_\psi + C_\phi S_\theta C_\psi \\ S_\phi C_\theta & C_\phi C_\psi + S_\phi S_\theta S_\psi & -C_\phi S_\psi + S_\phi S_\theta C_\psi \\ -S_\theta & C_\theta S_\psi & C_\theta C_\psi \end{pmatrix} \quad \textbf{(2.6)}$$

Inverse problem requires solution of equation set 2.7 to obtain pitch angle θ, roll angle ϕ and yaw angle ψ. The solution remains nonsingular in the range $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$

$$\begin{aligned} \phi &= Atan2(r_{21}, r_{11}) \\ \theta &= Atan2\left(-r_{31}, C_\phi r_{11} + S_\phi r_{21}\right) \\ \psi &= Atan2\left(S_\phi r_{13} - C_\phi r_{23}, -S_\phi r_{12} + C_\phi r_{22}\right) \end{aligned} \quad \textbf{(2.7)}$$

In terms of tracking error, relative rotation matrix R_e can be obtained using feedback and desired matrices as per equation 2.8.

$$R_e = {R_f}^T R_d \quad \textbf{(2.8)}$$

Assuming relatively low tracking errors, this representation avoids singularity during extraction of roll, pitch and yaw angles.

2.2.2. Angle/Axis representation

Another representation of orientation uses angle α and an axis defined using vector $\boldsymbol{r}$. Rotation about $\boldsymbol{r}$ through angle α can be found using Rodrigues` rotation formula.

$$R = I + \sin\alpha\, \hat{\boldsymbol{r}} + (1 - \cos\alpha)\, \hat{\boldsymbol{r}} \quad \textbf{(2.9)}$$

Determining angle and axis from a known rotation matrix can be accomplished using following equation set.

$$\begin{aligned} \alpha &= cos^{-1}\left(\frac{Tr(R) - 1}{2}\right) \\ \boldsymbol{r} &= \frac{1}{2\sin\alpha} \begin{pmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{pmatrix} \end{aligned} \quad \textbf{(2.10)}$$

In case where $R_{11} + R_{22} + R_{33} = 3$, $\alpha = 0$. When $R_{11} + R_{22} + R_{33} = -1$, $\alpha = \pi$. The representation is not unique as $R(-\alpha, -\boldsymbol{r}) = R(\alpha, \boldsymbol{r})$.

Error orientation can be extracted from feedback and desired rotation matrices using individual row vectors. This leads to the following axis/angle orientation error.

$$\boldsymbol{O_e} = \frac{1}{2}(\boldsymbol{n_f} \times \boldsymbol{n_d} + \boldsymbol{s_f} \times \boldsymbol{s_d} + \boldsymbol{a_f} \times \boldsymbol{a_d}) \quad \textbf{(2.11)}$$

2.2.3. Quaternions

Lastly, quaternions offer unique solution within range $[-\pi, \pi]$. Rotational matrix can be found using rotation angle α and axis vector r by defining scalar and vector parts of quaternion as $\eta = \cos\frac{\alpha}{2}$ and $\boldsymbol{\epsilon} = \sin\frac{\alpha}{2}\boldsymbol{r}$.

$$R = (\eta^2 - \boldsymbol{\epsilon}^T\boldsymbol{\epsilon})I + 2\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T + 2\eta\hat{\boldsymbol{\epsilon}} \quad \textbf{(2.12)}$$

Inverse problem requires solution of the following equation set.

$$\begin{gathered} \eta = \frac{1}{2}\sqrt{R_{11} + R_{22} + R_{33} + 1} \\ \boldsymbol{\epsilon} = \frac{1}{2}\begin{pmatrix} sgn(R_{32} - R_{23})\sqrt{R_{11} - R_{22} - R_{33} + 1} \\ sgn(R_{13} - R_{31})\sqrt{R_{22} - R_{33} - R_{11} + 1} \\ sgn(R_{21} - R_{12})\sqrt{R_{33} - R_{11} - R_{22} + 1} \end{pmatrix} \end{gathered} \quad \textbf{(2.13)}$$

As is previous cases, orientation error is required for proper orientation control. Using error rotation matric as per equation 2.8, and extracting from it vector quaternion part, suitable error can be acquired.

$$\boldsymbol{O_e} = R_f\boldsymbol{\epsilon_{df}} \quad \textbf{(2.14)}$$

2.3. Forward and Inverse Kinematics

Relation between joint velocities and operational, or task velocities $\boldsymbol{V}$, composed of linear and angular parts, is described using Jacobian. This matrix operator provides direct mapping between the two.

$$\boldsymbol{V} = J(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \quad \textbf{(2.15)}$$

Basic derivation of the relationship can be obtained by analyzing linear, $\boldsymbol{v_i}$, and angular, $\boldsymbol{\omega_i}$, velocities for consecutive links, link vector $\boldsymbol{l_{i-1,i}}$, and axis of revolution vectors $\boldsymbol{h_i}$ as per equation 2.16. Please see Figure 2.2 for reference.

$$\begin{pmatrix}\boldsymbol{\omega_i}\\ \boldsymbol{v_i}\end{pmatrix} = \begin{pmatrix} I & 0\\ -\widehat{\boldsymbol{l_{i-1,i}}} & I\end{pmatrix}\begin{pmatrix}\boldsymbol{\omega_{i-1}}\\ \boldsymbol{v_{i-1}}\end{pmatrix} + \begin{pmatrix}\boldsymbol{h_i}\\ \boldsymbol{0}\end{pmatrix}\dot{\theta}_i \qquad \textbf{(2.16)}$$

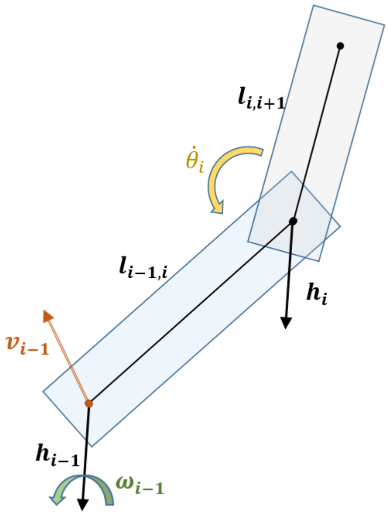


Figure 2.2 : Geometric and kinematic link parameters.

Establishing and collecting expressions for all relevant links, including end effector, and expressing result in matrix form, leads directly to Jacobian identification as per equation 2.15. This approach enables straightforward evaluation of the Jacobian at each time interval. It is suitable for numerical calculations.

Likewise, inverse relationship can be obtained using Jacobian inverse.

$$\dot{\boldsymbol{\theta}} = J(\boldsymbol{\theta})^{-1}\boldsymbol{V} \qquad \textbf{(2.17)}$$

Singular manipulator configuration are joint parameter sets at which Jacobian loses rank. This indicates that the manipulator has lost one or more degrees of freedom and certain directions of motion are unattainable. Furthermore, specific task velocities of the end effector may result in unbounded velocities in configuration space.

While extensive literature is devoted to dealing with singular configurations, no significant work has been devoted to this area of study. For the purpose of this thesis, singular configurations have been avoided all together. Manipulator operation has been confined to regions with safe joint parameter configurations.

2.4. Dynamics

Manipulator dynamics considers forces and moments and their effect on motion of the manipulator. Dynamic model leads to construction of equations of motion describing behavior of the system as it evolves in time. General form of the direct dynamics expressing joint accelerations as a function of generalized coordinates, joint torques and external forces and moments, as well as the inverse dynamics, are given in equation 2.18 and 2.19. Two separate, but equivalent formulations have been developed, and are presented below.

$$\ddot{\boldsymbol{q}} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\Gamma}, \boldsymbol{f}) = A(\boldsymbol{q})^{-1}\,[\,\boldsymbol{\Gamma} - \mathbf{H}(\boldsymbol{q}, \dot{\boldsymbol{q}})\,] \tag{2.18}$$

$$\boldsymbol{\Gamma} = g(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{f}) = A(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{H}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{2.19}$$

$$\mathbf{H}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{Q}(\boldsymbol{q}) + \mathbf{S}(\boldsymbol{q}) + \boldsymbol{\mathcal{T}}(\dot{\boldsymbol{q}}) + J^T\boldsymbol{f} \tag{2.20}$$

Torque due to externa loads is included as a vector $\boldsymbol{f}$ and is composed of 3x1 force and moment vectors $\boldsymbol{F_f}$ and $\boldsymbol{M_f}$. Friction torque discussed in the following pages is represented by the vector $\boldsymbol{\mathcal{T}}(\dot{\boldsymbol{q}})$. Joint two spring torque is captured as $\mathbf{S}(\boldsymbol{q})$ and the inertia matrix of the manipulator as $\mathbf{A}(\boldsymbol{q})$.

2.4.1. Euler-Lagrange formulation

Lagrange equations given in equation 2.21, and expressed in terms of generalized coordinates, present dynamics of a system it terms of its energy described by the Lagrangian function L.

$$\Gamma_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \qquad for\ i = 1,2 \dots n$$
$$L = E - U \tag{2.21}$$

Kinetic energy of the system E is expressed in equation 2.22 and includes both linear and angular velocities.

$$E = \frac{1}{2}\dot{\boldsymbol{q}}^{\boldsymbol{T}} A(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2.22}$$

The potential energy U is a function of joint positions, or vector $\boldsymbol{q}$, with gravitational potential energy playing the dominant role. Evaluation of the Lagrange equations gives the following inverse dynamics.

$$\boldsymbol{\Gamma} = A(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{Q}(\boldsymbol{q}) \qquad \textbf{(2.23)}$$

Coriolis and centrifugal torques given as $C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ are explicitly shown in equation 2.24.

$$\begin{aligned} C_{ij} &= \sum_{k-1}^{n} c_{i,jk}\, \dot{q}_k \\ c_{i,jk} &= \frac{1}{2}\left(\frac{\partial A_{ij}}{\partial q_k} + \frac{\partial A_{ik}}{\partial q_j} + \frac{\partial A_{jk}}{\partial q_i}\right) \end{aligned} \qquad \textbf{(2.24)}$$

Incorporating spring torque and friction torque contributions leads directly to inverse dynamics shown in equation 2.19.

2.4.2. Newton-Euler formulation

Newton-Euler formulation can be derived from Euler`s laws describing rate of change of linear and angular momentums. Here, individual links are analyzed one after another. Starting from the base, velocities and acceleration of all links are calculated. Once complete, backward recursion is performed as the forces and moments acting on each link in turn are obtained. Forward and backward algorithms are presented below.

Referring back to equation 2.5 and Figure 2.2, linear and angular velocities are obtained as follows.

$$\begin{aligned} \boldsymbol{\omega_i} &= \boldsymbol{\omega_{i-1}} + \boldsymbol{h_i}\dot{\theta} \\ \boldsymbol{v_i} &= \boldsymbol{v_{i-1}} + \boldsymbol{\omega_{i-1}} \times \boldsymbol{l_{i-1,i}} \end{aligned} \qquad \textbf{(2.25)}$$

Differentiating above equations gives us linear and angular accelerations of individual links.

$$\begin{aligned} \dot{\boldsymbol{\omega}}_{\boldsymbol{i}} &= \dot{\boldsymbol{\omega}}_{\boldsymbol{i-1}} + \boldsymbol{\omega_{i-1}} \times \boldsymbol{\omega_i} + \boldsymbol{h_i}\ddot{\theta} \\ \dot{\boldsymbol{v}}_{\boldsymbol{i}} &= \dot{\boldsymbol{v}}_{\boldsymbol{i-1}} + \dot{\boldsymbol{\omega}}_{\boldsymbol{i-1}} \times \boldsymbol{l_{i-1,i}} + \boldsymbol{\omega_{i-1}} \times (\boldsymbol{\omega_{i-1}} \times \boldsymbol{l_{i-1,i}}) \end{aligned} \qquad \textbf{(2.26)}$$

Summation of all forces and torques acting on individual links give us equations required for backward recursion. It begins with the end effector and moves back towers the base of the manipulator.

$$\begin{aligned} \boldsymbol{\tau_i} &= \boldsymbol{\tau_{i+1}} + \left(\boldsymbol{l}_{i,i+1} \times \boldsymbol{f_{i+1}}\right) + \frac{d}{dt}(I_i\boldsymbol{\omega_i}) + \boldsymbol{l_{ic}} \times \dot{\boldsymbol{v}}_i m_i \\ \boldsymbol{f_i} &= \boldsymbol{f_{i+1}} + m_i \frac{d}{dt}(\boldsymbol{v_i} + \boldsymbol{\omega_i} \times \boldsymbol{l_{ic}}) \end{aligned} \tag{2.27}$$

First and the second terms in the torque equation represent torque and force contributions from the upper link. Derivative term is the torque due to rotational acceleration. The last term stands for the contribution form the linear acceleration of the link. Similarly, first term in the force equation comes from the loads due to upper link, while the derivative term is due to linear and angular motion of the link currently analyzed. I_i is the inertia of the link with respect to its local frame and remains constant. Local vector linking origin of the link frame with the center of mass is labeled as $\boldsymbol{l_{ic}}$. Gravity is introduced as a constant g acting along z direction in the linear acceleration vector of the base link ($\dot{\boldsymbol{v}}_{\boldsymbol{0z}}$). Carrying through differentiation gives us the final set of equations for the backward recursion.

$$\begin{aligned} \boldsymbol{\tau_i} &= \boldsymbol{\tau_{i+1}} + \left(\boldsymbol{l}_{i,i+1} \times \boldsymbol{f_{i+1}}\right) + \boldsymbol{\omega_i} \times I_i\boldsymbol{\omega_i} + I_i\dot{\boldsymbol{\omega}}_{\boldsymbol{i}} + \boldsymbol{l_{ic}} \times \dot{\boldsymbol{v}}_i m_i \\ \boldsymbol{f_i} &= \boldsymbol{f_{i+1}} + \dot{\boldsymbol{v}}_i m_i + m_i\dot{\boldsymbol{\omega}}_{\boldsymbol{i}} \times \boldsymbol{l_{ic}} + m_i\boldsymbol{\omega_i} \times (\boldsymbol{\omega_i} \times \boldsymbol{l_{ic}}) \end{aligned} \tag{2.28}$$

Figure 2.3 illustrates forces and torques acting on the i^th link.

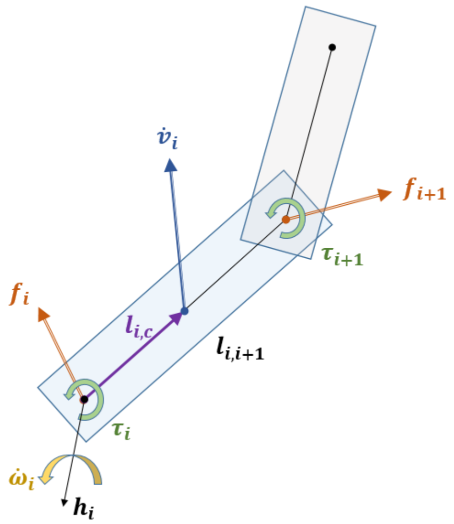


Figure 2.3 : Forces and torques acting on the i^th link.

2.4.3. Task space dynamics

Using Jacobian transformation between task and joint velocities as well as its derivative as per equation 2.29 allows one to transform joint space inverse dynamics of equation 2.19 into task space.

$$\dot{\boldsymbol{V}} = J(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \tag{2.29}$$

Resulting equation 2.30 is now in the form required for implementation of task space control.

$$\boldsymbol{\Gamma} = AJ^{-1}\left(\ddot{\boldsymbol{x}} - \dot{J}\dot{\boldsymbol{q}}\right) + \mathbf{H}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + J^T\boldsymbol{f} \tag{2.30}$$

While possible, analytical evaluation of the $\dot{J}\dot{\boldsymbol{q}}$ is not practical as it requires differentiation of already formidable expression for the Jacobian**.** Numerical evaluation of the term $\dot{J}\dot{\boldsymbol{q}}$ is performed using Spatial Operator Algebra. Required formalism is presented in Appendix B.

2.4.4. Friction

Friction model employed in this work comes from Waiboer (2007). Author has applied friction model to Stäubli RX90B robotic manipulator. Friction parameter estimation for Stäubli RX160 manipulator was done by Zengin (2013), and later modified by Akbaş (2015). Friction model is based on the friction of a helical gear pair. It is composed of two terms. Contact asperities dominate friction in low velocity regions associated with the Boundary Lubrication (BL). As the relative velocity increases so does the lubrication layer. Friction force attributed to surface contact drops. Hoverer, as the velocity continues to rise, viscous friction increases. This is a characteristic effect seen in the Mixed Lubrication (ML) zone. Further increase in velocity completely separates both surfaces. At this stage, overall friction continues to increase and the fore required to overcome it is that needed to shear the lubricant film. At this point, Elasto-Hydrodynamic Lubrication (EHL) region is reached. Friction model neglects Coulomb friction associated with dry friction. Figure 2.4 provides graphical illustration of the three friction zones mentioned above, as well as the general shape of the friction curve described by the helical gear friction model. Domains of the contact asperities and viscous friction are shown.

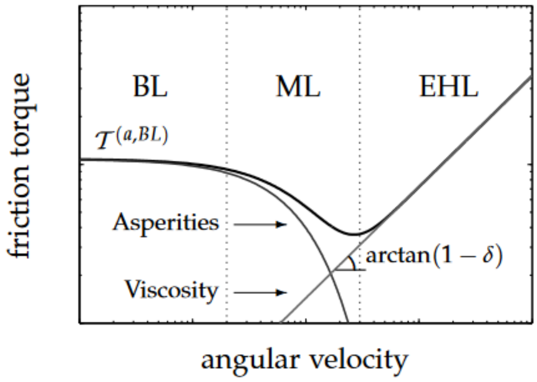


Figure 2.4 : Typical helical gear friction (Waiboer, 2007).

Friction torque for the first four joints was modeled on the equation 2.31.

$$\mathcal{T}_j = \mathcal{T}_j^{(a)} + \mathcal{T}_j^{(v)} = \mathcal{T}_j^{(a,BL)} e^{-(\dot{q}/q^{(s)})^{\delta_j^{(a)}}} + c_j{}^{(v)} \dot{q}_j{}^{(1-\delta_j^{(v)})} \tag{2.31}$$

Five unknown parameters presented in the above equation include asperity friction $\mathcal{T}_j^{(a,BL)}$, Stribeck velocity $c_j{}^{(v)}$, Stribek velocity power $\delta_j^{(a)}$, viscous friction coefficient $c_j{}^{(v)}$, and viscous friction power $\delta_j^{(v)}$. Experimental identification has been performed to obtain above parameter.

Coupling of the fifth and the sixth joints, as seen in Figure 1.2, requires a separate consideration. Additional friction model has been added to accommodate this effect.

Equations 2.32 through 2.35 describe friction model used in joint five and six.

$$\dot{q}_7 = \dot{q}_5 + \dot{q}_6 \tag{2.32}$$

$$\mathcal{T}_a = \mathcal{T}_5^{(a,BL)} e^{-\left(\frac{\dot{q}_5}{q_5{}^{(s)}}\right)^{\delta_5^{(a)}}} + c_5{}^{(v)} \dot{q}_5{}^{\left(1-\delta_5^{(v)}\right)} \tag{2.33}$$

$$\mathcal{T}_b = \mathcal{T}_6^{(a,BL)} e^{-\left(\frac{\dot{q}_6}{q_6{}^{(s)}}\right)^{\delta_6^{(a)}}} + \mathcal{T}_6^{(v,l)} \left(1 - e^{-\dot{q}_6/q_6{}^{(l)}}\right) \tag{2.34}$$

$$\mathcal{T}_c = \mathcal{T}_7^{(a,BL)} e^{-\left(\frac{\dot{q}_7}{q_7{}^{(s)}}\right)^{\delta_7^{(a)}}} + c_7{}^{(v)} \dot{q}_7{}^{\left(1-\delta_7^{(v)}\right)} \tag{2.35}$$

Finally, friction torque of joint five and six can be obtained as per equation set 2.36.

$$\mathcal{T}_5 = \mathcal{T}_a + \mathcal{T}_c$$
$$\mathcal{T}_6 = \mathcal{T}_b + \mathcal{T}_c \tag{2.36}$$

Friction parameters estimation by Zengin (2013) and Akbaş (2015) are presented.

Table 2.1 : Friction parameters estimation by Zengin (2013).

Model	1	2	3	4	a	b	c
$\mathcal{T}^{(a,BL)}$	55.3234	30.8468	40.4853	168.4421	2.6767	-0.7271	1.0903
$\dot{q}^{(s)}$	0.0710	0.0400	0.0610	0.0200	0.2500	0.3500	0.0800
$\delta^{(a)}$	0.7000	0.8000	0.6000	0.5500	0.7000	0.5000	0.5000
$c^{(v)}$	191.9090	107.0665	59.4372	24.1345	20.0239	-	8.3559
$\delta^{(v)}$	0.3837	-0.1724	0.3739	0.3562	0.3494	-	0.3907
$\mathcal{T}^{(v,l)}$	-	-	-	-		-1.3742	-
$\dot{q}^{(l)}$	-	-	-	-		4.6408	-

Table 2.2 : Friction parameters estimation by Akbaş (2015).

Model	1	2	3	4	a	b	c
$\mathcal{T}^{(a,BL)}$	30	10	20	20	1.0	0.5	1
$\dot{q}^{(s)}$	0.0710	0.0400	0.0610	0.0200	0.2500	0.3500	0.0800
$\delta^{(a)}$	0.7000	0.8000	0.6000	0.5500	0.7000	0.5000	0.5000
$c^{(v)}$	120	50	20	10	10	-	5
$\delta^{(v)}$	0.3837	-0.1724	0.3739	0.3562	0.3494	-	0.3907
$\mathcal{T}^{(v,l)}$	-	-	-	-		-1.3742	-
$\dot{q}^{(l)}$	-	-	-	-		4.6408	-

2.5. Control Theory

Successful control of a robotic manipulator allows determination of required inputs necessary for the end effector to execute commanded motion. Theory and control architectures are presented for computed torque control, or inverse dynamic control, as well as for parallel hybrid position/force control. Both of these control methods were successfully applied in the control of the Master/Slave manipulators.

2.5.1. Computed torque control

Computed torque control relies on the method of feedback linearization. Here, nonlinear terms in the equations of motion are canceled out, reducing original system to control of a linear problem. Proper identification of the nonlinearities is necessary for robust implementation of this control strategy. This is achieved using Euler-

Newton algorithm for the inertial, Coriolis, centrifugal and gravitation torques, Proper model identification is used to describe and compensate for the torques resulting from internal spring and joint friction. This is the default control method used by the Master manipulator.

Recalling equation 2.30, and implementing feedback law of the form indicated in the equation 2.37 reduces problem to one shown in equation 2.38.

$$\boldsymbol{\Gamma} = AJ^{-1}\left(\boldsymbol{w} - \dot{J}\dot{\boldsymbol{q}}\right) + \mathbf{H}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + J^T\boldsymbol{f} \quad \textbf{(2.37)}$$

$$\ddot{\boldsymbol{x}} = \boldsymbol{w} \quad \textbf{(2.38)}$$

Term $\boldsymbol{w}$ can now be selected to implement effective control. For trajectory tracking PD controller, $\boldsymbol{w}$ can be chosen as per equation 2.39 with gains defined as $K = kI$.

$$\boldsymbol{w} = K_p\left(\boldsymbol{x_d} - \boldsymbol{x_f}\right) + K_d\left(\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \dot{\boldsymbol{x}}_{\boldsymbol{f}}\right) + \ddot{\boldsymbol{x}}_{\boldsymbol{d}} \quad \textbf{(2.39)}$$

Defining tracking error as per equation 2.40, transforms problem into a second order error equation 2.41.

$$\boldsymbol{x_e} = \boldsymbol{x_d} - \boldsymbol{x_f} \quad \textbf{(2.40)}$$

$$\ddot{\boldsymbol{x}}_{\boldsymbol{e}} + K_d\dot{\boldsymbol{x}}_{\boldsymbol{e}} + \boldsymbol{x_e}K_p = \mathbf{0} \quad \textbf{(2.41)}$$

This equation indicates tracking problem is exponentially stable and controllable for a positive definite choice of control gains K_p and K_d.

Inverse dynamics problem is presented graphically in Figure 2.5.

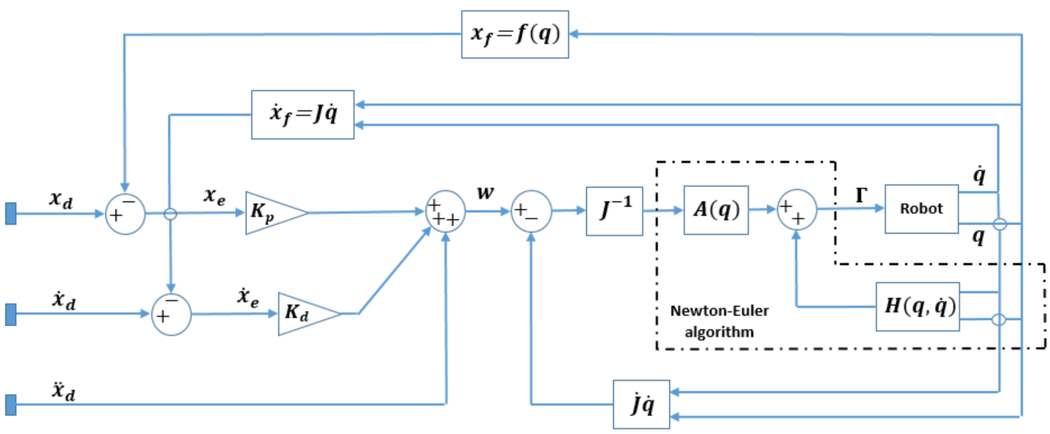


Figure 2.5 : Computed torque control in task space.

Orientation control can be accomplished in the identical manner with the proper selection of the orientation error $\boldsymbol{O_e}$. Vector $\boldsymbol{w}$ is decomposed onto two, vector $\boldsymbol{w_p}$ responsible for positional control and vector $\boldsymbol{w_o}$ in control of orientation**.**

Control law and error dynamics for the axis/angle formulation is given in equation 2.42 and 2.43 respectively.

$$\boldsymbol{w_o} = \dot{\boldsymbol{\omega}}_{\boldsymbol{d}} + K_{do}(\boldsymbol{\omega_d} - \boldsymbol{\omega_f}) + K_{po}\boldsymbol{O_e} \tag{2.42}$$

$$\dot{\boldsymbol{\omega}}_{\boldsymbol{e}} + K_{do}\boldsymbol{\omega_e} + K_{po}\boldsymbol{O_e} = \boldsymbol{0} \tag{2.43}$$

Using quaternions to express orientation error, as per equation 2.14, results in:

$$\boldsymbol{w_o} = \dot{\boldsymbol{\omega}}_{\boldsymbol{d}} + K_{do}(\boldsymbol{\omega_d} - \boldsymbol{\omega_f}) + K_{po}R_f\boldsymbol{\epsilon_{df}} \tag{2.44}$$

$$\dot{\boldsymbol{\omega}}_{\boldsymbol{e}} + K_{do}\boldsymbol{\omega_e} + K_{po}R_f\boldsymbol{\epsilon_{df}} = \boldsymbol{0} \tag{2.45}$$

In contrast to position error dynamics, above error equations are not linear and require proper justification of their stability. Asymptotic stability of both systems is proven using Lyapunov theory. Details are presented in Appendix C.

2.5.2. Hybrid position/force control

Version of parallel hybrid position/force control presented bellow can be seen as an extension of the inverse dynamic control. In addition to position and orientation, parallel force control is introduced. Torque contribution from the force control is added to output of the computed torque controller. Total joint torque is then send to the manipulator.

The control method allows for simultaneous control of position and force. However, selection of directions spanned by both controllers is required. Position or velocity controlled directions need to be orthogonal to force controlled directions in the compliant frame. Separation of the controllers is accomplished by introducing compliance selection matrix S. Setting diagonal elements of the selection matrix to one enables position control, while setting them to zero, enables force control along the particular directions. Figure 2.6 illustrates implementation of the hybrid position/force controller.

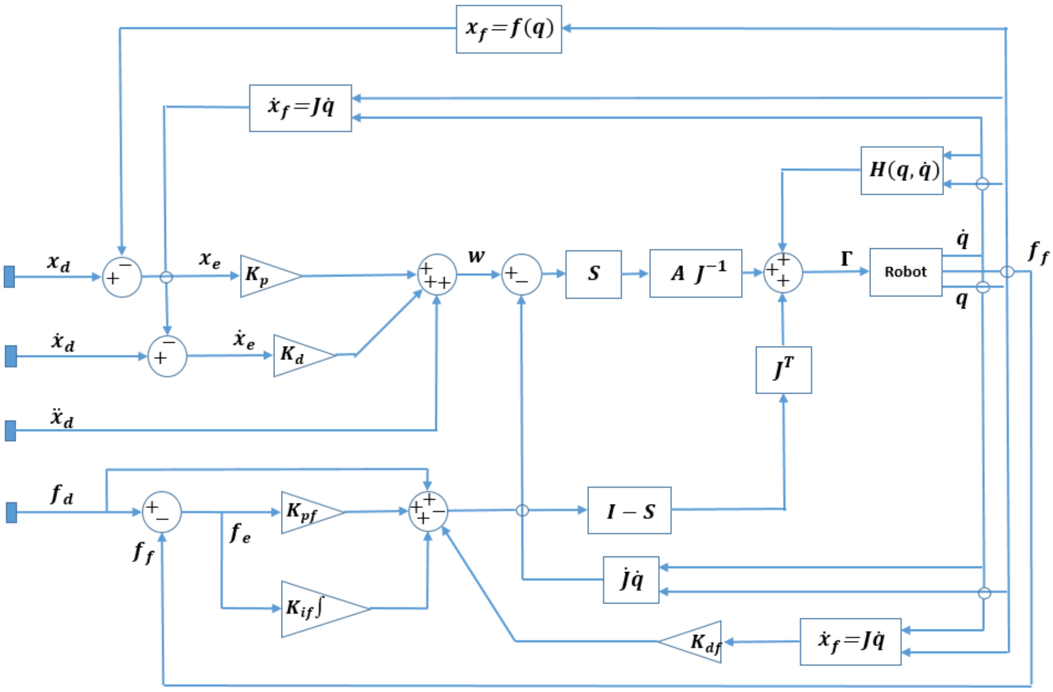


Figure 2.6 : Hybrid position-force control.

In addition to position and orientation control described in the previous section, force control can be implemented using control law as per equation 2.46.

$$\boldsymbol{\Gamma} = J^T(\boldsymbol{f_d} + k_{pf}(\boldsymbol{f_d} - \boldsymbol{f_f}) - k_{df}\dot{\boldsymbol{x}}_{\boldsymbol{f}} + k_{if}\int_{t_0}^{t}(\boldsymbol{f_d} - \boldsymbol{f_f})dt) \qquad \textbf{(2.46)}$$

Instead of using derivative of the noisy force signal, feedback velocity vector is used.

Control of fully compliant Slave manipulator has been based on hybrid position-force control architecture where compliance selection matrix is a zero matrix. As such, force controller is used along all linear and angular directions.

2.5.3. Sliding mode control

Robotic systems are highly susceptible to parametric uncertainties and external disturbances. Number of robust control methods has been proposed to rectify this problem. Sliding mode control has been successfully employed when dealing with bounded uncertainties.

Defining sliding surface as:

$$\boldsymbol{s} = \dot{\boldsymbol{x}}_{\boldsymbol{e}} + \Lambda \boldsymbol{x}_{\boldsymbol{e}} \qquad \textbf{(2.47)}$$

we notice that whenever on the surface, $\boldsymbol{s} = \mathbf{0}$, and for Λ positive definite, the system is governed by the asymptotically stable, first order equation.

Modifying control law $\boldsymbol{w}$ found in equation 2.39 as:

$$\boldsymbol{w} = \Lambda \dot{\boldsymbol{x}}_{\boldsymbol{e}} + K_1 \boldsymbol{s} + K_2 sng(\boldsymbol{s}) + \ddot{\boldsymbol{x}}_{\boldsymbol{d}} \tag{2.48}$$

gives following error dynamics:

$$\ddot{\boldsymbol{x}}_{\boldsymbol{e}} + \Lambda \dot{\boldsymbol{x}}_{\boldsymbol{e}} + K_1 \boldsymbol{s} + K_2 sng(\boldsymbol{s}) = \mathbf{0} \tag{2.49}$$

We define Lyapunov function and calculate resulting derivative.

$$V = \frac{1}{2} \boldsymbol{s}\, \boldsymbol{s}^T \tag{2.50}$$

$$\begin{aligned} \dot{V} &= \boldsymbol{s}^T \dot{\boldsymbol{s}} \\ &= \boldsymbol{s}^T (\ddot{\boldsymbol{x}}_{\boldsymbol{e}} + \Lambda \dot{\boldsymbol{x}}_{\boldsymbol{e}}) \\ &= \boldsymbol{s}^T (K_1 \boldsymbol{s} - K_2 sng(\boldsymbol{s})) \end{aligned} \tag{2.51}$$

In presence of model uncertainties, additional term $\boldsymbol{u}$ is present,

$$\ddot{\boldsymbol{x}}_{\boldsymbol{e}} + \Lambda \dot{\boldsymbol{x}}_{\boldsymbol{e}} + K_1 \boldsymbol{s} + K_2 sng(\boldsymbol{s}) = \boldsymbol{u} \tag{2.52}$$

defined as

$$\boldsymbol{u} = \tilde{J} \tilde{A}^{-1} Y \Delta p \tag{2.53}$$

where Δp is the estimation error of dynamic parameters, $Y(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ is the dynamic regression matrix and ˜ stands for estimated robot parameters.

Next, $\dot{V}$ becomes:

$$\dot{V} = \boldsymbol{s}^T (\boldsymbol{u} - K_1 \boldsymbol{s} - K_2 sng(\boldsymbol{s})) \tag{2.54}$$

If we then assume upper limit on the magnitude of the uncertainties and select gain K_2 such as:

$$K_2 \geq |\boldsymbol{u}_{max}| \tag{2.55}$$

the Lyapunov derivative becomes

$$\dot{V} \le -\boldsymbol{s}^T K_1 \boldsymbol{s} \le 0 \quad \textbf{(2.56)}$$

Further, assuming

$$K_1 > 0 \quad \textbf{(2.57)}$$

leads to negative semidefinite derivative of the Lyapunov function. This ensures that the error dynamics equation, in the presence of uncertainties, and with the proper selection of the gains as per equation 2.55 and 2.57, is stable. Furthermore, asymptotic stability can be further deduced by noticing that in the case of semidefinite $\dot{V}$, only acceptable solution is $\boldsymbol{s} = \boldsymbol{0}$ for which unique solution is $\boldsymbol{x_e} = \boldsymbol{0}$.

Sliding mode control can thus compensate for uncertainties in the system model and can bring it to the sliding surface. However, this is accomplished at the expense of high actuator activity resulting from control discontinuity $K_2 sng(\boldsymbol{s})$. This often leads to chattering, an undesirable feature of the sliding mode control. In addition, system uncertainties must be estimated and bounded as per equation 2.55.

2.6. Signal Processing

Feedback data are critical in the implementation of manipulator control. This is complicated by inherent noise associated with transducers used this application.

Linear Quadratic Estimator (LQE) or Kalman filter, has been chosen as primary filter for both, joint velocities and force/torque signals. It is a recursive algorithm that utilizes measurement data, and mathematical process model, to produce estimates of unknown variables.

Given sequence of noisy measurements, true state of the system can be evaluated. Evolution of a state from x_{k-1} to x_k as well as measurement z_k of the true state is governed by the set of linear equations.

$$\boldsymbol{x_k} = F_k \boldsymbol{x_{k-1}} + B_k \boldsymbol{u_k} + \boldsymbol{w_k} \quad \textbf{(2.58)}$$

$$\boldsymbol{z_k} = H_k \boldsymbol{x_k} + \boldsymbol{v_k} \quad \textbf{(2.59)}$$

Here, F_k is the state transition model, B_k is control input model, $\boldsymbol{u_k}$ is the control vector, and $\boldsymbol{w_k}$ is the process noise. In the second equation, H_k is the measurement model and $\boldsymbol{v_k}$ represents measurement noise. Both, process and measurement noise are assumed to be sampled form a zero mean normal distribution with covariance Q_k and R_k respectively.

Kalman filter is usually performed in two distinct phases. First, in the prediction phase estimation of the current state is done based on the state from previous time interval. The predicted state is then updated based on available measurements. This take s place during the update phase. Complete algorithm is presented below.

$$\boldsymbol{\bar{x}_{k|k-1}} = F_k \boldsymbol{\bar{x}_{k-1|k-1}} + B_k \boldsymbol{u_k} \tag{2.60}$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \tag{2.61}$$

$$\boldsymbol{\bar{y}_k} = \boldsymbol{z_k} - H_k \boldsymbol{\bar{x}_{k|k-1}} \tag{2.62}$$

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{2.63}$$

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{2.64}$$

$$\boldsymbol{\bar{x}_{k|k}} = \boldsymbol{\bar{x}_{k|k-1}} + K_k \boldsymbol{\bar{y}_k} \tag{2.65}$$

First two equations constitute the prediction phase of the filter. Remaining four equations perform update phase. Matrices $P, S,$ and K are predicted estimate covariance, residual covariance and the optimal Kalman gain respectively.

The above equations reduce substantially when applied to a problem of one dimensional signal filtering. As no active process takes place control vector $\boldsymbol{u_k}$ is zero. As the measurement is of state value with addition of noise, H becomes a scalar of magnitude one, as is state transition matrix F.Noise covariance matrices R and Q become scalars r and q correspondingly. Final equations applicable in the presented case are:

$$\bar{x}_{k|k-1} = \bar{x}_{k-1|k-1} \tag{2.66}$$

$$p_{k|k-1} = p_{k-1|k-1} + q \tag{2.67}$$

$$\bar{y}_k = z_k - \bar{x}_{k|k-1} \tag{2.68}$$

$$s_k = p_{k|k-1} + r \tag{2.69}$$

$$k_k = p_{k|k-1} S_k^{-1} \tag{2.70}$$

$$\bar{x}_{k|k} = \bar{x}_{k|k-1} + k_k \bar{y}_k \tag{2.71}$$

Algorithm can be initiated by the specifying noise covariance r and q, and setting $x_0 = 0,\ p_0 = 0$, and $k_0 = 0$. Practical application of the outlined theory can be found in the experimental chapter.

2.7. Trajectory Generation

Execution of desired motion requires proper construction of trajectory for the manipulator to follow. Trajectory generation in task space can be generated using polynomial functions describing evolution of position and orientation as a function of time. Using axis/angle formulation, orientation of the end effector can be tracked by the changing vector r and rotation angle α. Knowing initial and final orientation permits calculation of the relative rotational transformation as per equation 2.72.

$$R = {R_1}^T R_2 \tag{2.72}$$

Extraction of the angle α, and rotational axis r from R can be done using following equations.

$$\cos\alpha = \frac{1}{2}(R_{11} + R_{22} + R_{33} - 1) \tag{2.73}$$

$$\sin\alpha = \frac{1}{2}\sqrt{(R_{32} - R_{23})^2 + (R_{13} - R_{31})^2 + (R_{21} - R_{12})^2} \tag{2.74}$$

$$\alpha = atan2(\sin\alpha, \cos\alpha) \tag{2.75}$$

$$\boldsymbol{r} = \frac{1}{2\sin\alpha}\begin{pmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{pmatrix} \tag{2.76}$$

In case where $\cos\alpha < 0$, following equations are used:

$$\sin\alpha = \frac{1}{2}\sqrt{1-\cos\alpha^2} \tag{2.77}$$

$$r_x = sign(R_{32}-R_{23})\sqrt{\frac{R_{11}-\cos\alpha}{1-\cos\alpha}} \tag{2.78}$$

$$r_y = sign(R_{13}-R_{31})\sqrt{\frac{R_{22}-\cos\alpha}{1-\cos\alpha}} \tag{2.79}$$

$$r_z = sign(R_{21}-R_{12})\sqrt{\frac{R_{33}-\cos\alpha}{1-\cos\alpha}} \tag{2.80}$$

For each trajectory segment, calculation of a single rotational axis $\boldsymbol{r}$, which remains constant, and an angle α, which evolves from zero to its final value, is performed.

Specification of boundary conditions such as initial and final positions, velocities and accelerations, both for position and orientation, allows for a construction of polynomial functions describing motion of the manipulator from point one to point two. Polynomial coefficients for position vector $\boldsymbol{x}$ and angle α and calculated offline and then reused to create incremental desired position and orientation of the end effector. Coefficients for a quantic polynomial describing motion between point $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ with period of motion T are as follows:

$$\boldsymbol{a_0} = \boldsymbol{x_1} \tag{2.81}$$

$$\boldsymbol{a_1} = \dot{\boldsymbol{x}}_{\boldsymbol{1}} \tag{2.82}$$

$$\boldsymbol{a_2} = \ddot{\boldsymbol{x}}_{\boldsymbol{1}}/\boldsymbol{2} \tag{2.83}$$

$$\boldsymbol{a_3} = \frac{20(\boldsymbol{x_2}-\boldsymbol{x_1}) + T(\ddot{\boldsymbol{x}}_{\boldsymbol{2}}T - 3\ddot{\boldsymbol{x}}_{\boldsymbol{1}}T - 8\dot{\boldsymbol{x}}_{\boldsymbol{2}} - 12\dot{\boldsymbol{x}}_{\boldsymbol{1}})}{2T^3} \tag{2.84}$$

$$\boldsymbol{a_4} = \frac{-30(\boldsymbol{x_2}-\boldsymbol{x_1}) + T(-2\ddot{\boldsymbol{x}}_{\boldsymbol{2}}T + 3\ddot{\boldsymbol{x}}_{\boldsymbol{1}}T + 14\dot{\boldsymbol{x}}_{\boldsymbol{2}} + 16\dot{\boldsymbol{x}}_{\boldsymbol{1}})}{2T^4} \tag{2.85}$$

$$\boldsymbol{a_5} = \frac{12(\boldsymbol{x_2}-\boldsymbol{x_1}) + T(\ddot{\boldsymbol{x}}_{\boldsymbol{2}}T - \ddot{\boldsymbol{x}}_{\boldsymbol{1}}T - 6\dot{\boldsymbol{x}}_{\boldsymbol{2}} - 6\dot{\boldsymbol{x}}_{\boldsymbol{1}})}{2T^5} \tag{2.86}$$

Coefficients of the rotation angle α take identical form but each is a scalar equation as opposed to vector equations shown above.

Reconstruction of desired position takes place online using previously stored values of $\boldsymbol{a_0} \dots . \boldsymbol{a_4}$.

$$\boldsymbol{x_d}(t) = \boldsymbol{a_0} + \boldsymbol{a_1}t + \boldsymbol{a_2}t^2 + \boldsymbol{a_3}t^3 + \boldsymbol{a_4}t^4 + \boldsymbol{a_5}t^5 \tag{2.87}$$

$$\dot{\boldsymbol{x}}_{\boldsymbol{d}}(t) = \boldsymbol{a_1} + 2\boldsymbol{a_2}t + 3\boldsymbol{a_3}t^2 + 4\boldsymbol{a_4}t^3 + 5\boldsymbol{a_5}t^4 \tag{2.88}$$

$$\ddot{\boldsymbol{x}}_{\boldsymbol{d}}(t) = 2\boldsymbol{a_2} + 6\boldsymbol{a_3}t + 12\boldsymbol{a_4}t^2 + 20\boldsymbol{a_5}t^3 \tag{2.89}$$

Angle $\alpha(t), \dot{\alpha}(t)$, and, $\ddot{\alpha}(t)$ are obtained using same procedure. Recovery of the desired orientation requires calculation of the rotational matrix at every time interval. This is done using equation 2.9 which restores relative rotation matrix. Finally, multiplying by rotational matrix R_1 provides desired rotation matrix at a time t expressed in base frame. Desired angular velocity, again in the base frame, can be obtained using equation 2.90.

$$\boldsymbol{\omega_d}(t) = R_1\, \boldsymbol{r}\, \dot{\alpha}(t) \tag{2.90}$$

Angular acceleration vector follows:

$$\dot{\boldsymbol{\omega}}_{\boldsymbol{d}}(t) = R_1\, \boldsymbol{r}\, \ddot{\alpha}(t) \tag{2.91}$$

Multipoint trajectories can be performed through proper selection of boundary conditions.

3. SIMULATION STUDIES

Robotics and control theory presented in the previous chapter has been applied in constructing models using Matlab Simulink (R2014a). Individual models for the Master and the Slave were built and finally combined together to create a complete system. Results of numerous simulations are presented. At this stage, no specific performance criteria were used to achieve a particular response.

3.1. Master Manipulator

Block diagram for the Master model is depicted in Figure 3.1. The loop begins with the introduction of the reference trajectory. At each time instant a new pose, position and orientation in task space is provided. This is combined with the feedback signal which upon exiting forward kinematic block is also expressed in task space. Both signals are used to calculate resulting error to be used in the PD controller illustrated in Figure 3.2. Orientation error is calculated using angle/axis formulism. Both position and orientation control utilize linear and angular acceleration feedforward respectfully. The resulting 6x1 vector w is then combined with the $\dot{J}_e\dot{\boldsymbol{\theta}}$ and converted to joint space using ${J_e}^{-1}$. The output, $\ddot{\boldsymbol{\theta}}$, is then feed into Newton-Euler algorithm.

Master control architecture is built around feedback linearization outlined previously. For complete cancelation of the linear and nonlinear dynamic terms, torque contributions from the Newton-Euler algorithm describing gravitational, centrifugal and Coriolis forces are combined with the friction and spring contributions. Resulting joint torque $\boldsymbol{\tau}$ is then sent to the robot`s low level control. For the purpose of the simulation, robot block cancels out contributions from all the dynamic terms, applies inverse of the inertia matrix and integrates signal twice. Therefore, at heart, this is a second order system characterized by the inertia matrix and controlled using proportional and derivative terms. Finally, output of the robot bloc consists of joint angles and velocities vectors.

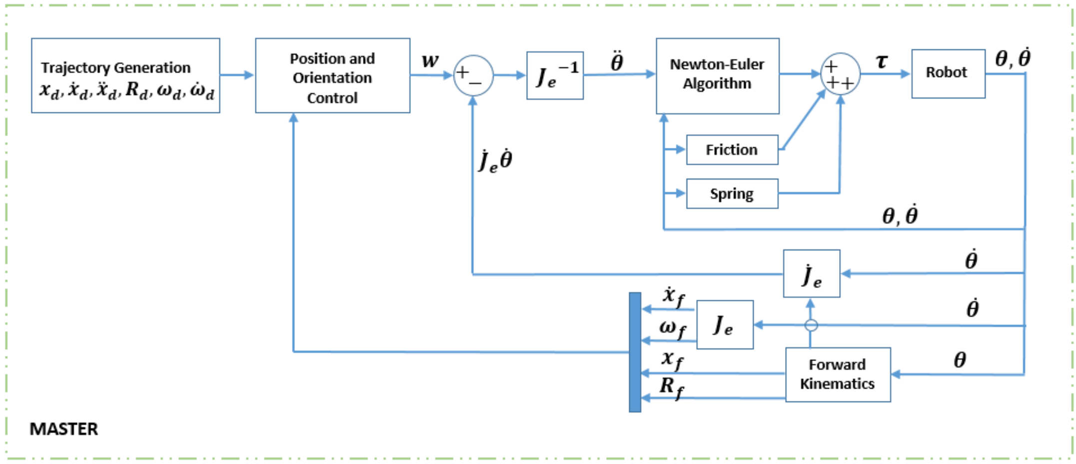


Figure 3.1 : Master manipulator block diagram.

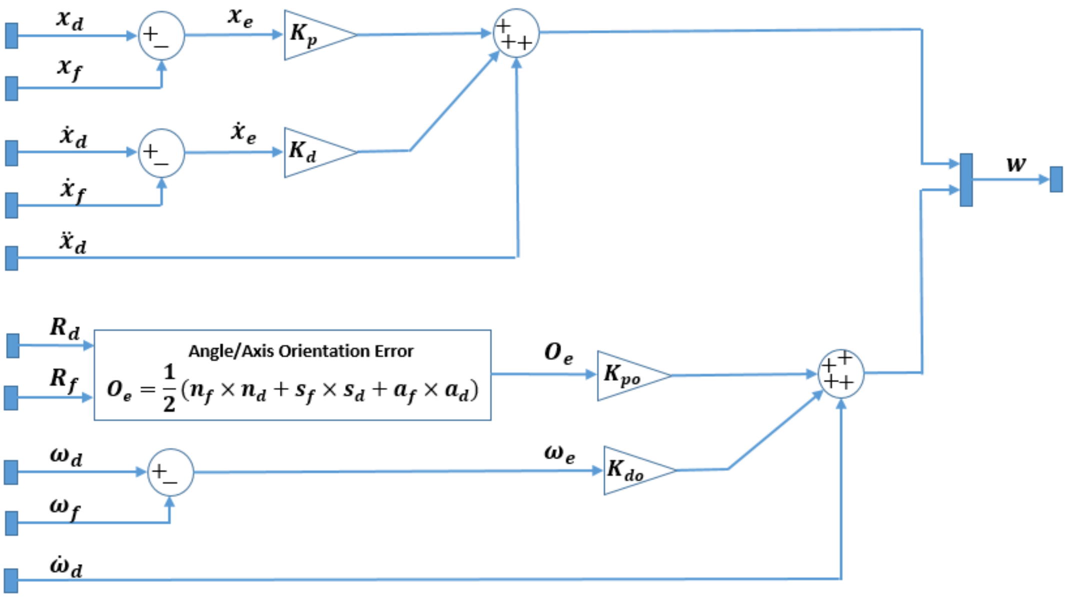


Figure 3.2 : Position and orientation control of the master manipulator.

Estimation of the control gains for the position and orientation was performed manually. Qualitative inspection revealed that gains of $k_p = 100$ and $k_d = 20$, same for position and orientation, were more than sufficient. Figure 3.4 and Figure 3.5 illustrate results of the simulations for gains $k_p = 9, k_d = 6$ and $k_p = 100, k_d = 20$. Table 3.1 gives position points, joint angles and time at each trajectory point. While position in the table is expressed in joint space, actual path planning and control was performed in task space.

Clearly, second set of gains results in better tracking performance and lower overall error in terms of position and orientation. Again, it is understood that the gains were not representative of the actual hardware and proper gain tuning would have to be performed later.

Table 3.1 : Reference trajectory data for the Master.

Point	Joint Position (°)						Time (s)
	Joint _1	Joint _2	Joint _3	Joint _4	Joint _5	Joint _6	
1	0	-20	110	0	90	0	0
2	45	45	45	45	45	45	5
3	45	45	45	45	45	45	6

3.1.1. Joint velocity noise

Measurement of the joint velocities is done using resolvers. Velocity measurements signal is often noisy. To simulate noise at the robot output, Matlab Band-Limited White Noise block was introduced into the feedback loop as per Figure 3.3. The simulation failed to provide results within acceptable timeframe. Oscillation of the velocity signal at the beginning of motion leads to high number of zero-crossings which results in a failure to converge. This outcome stresses the importance of a clean velocity signal required for the actual system. Suitable filtering will be required to resolve this issue.

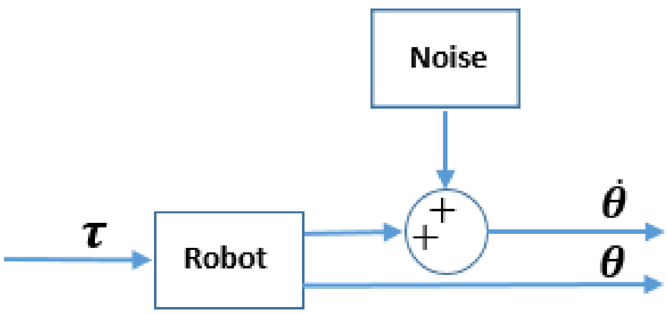


Figure 3.3 : Joint velocity noise.

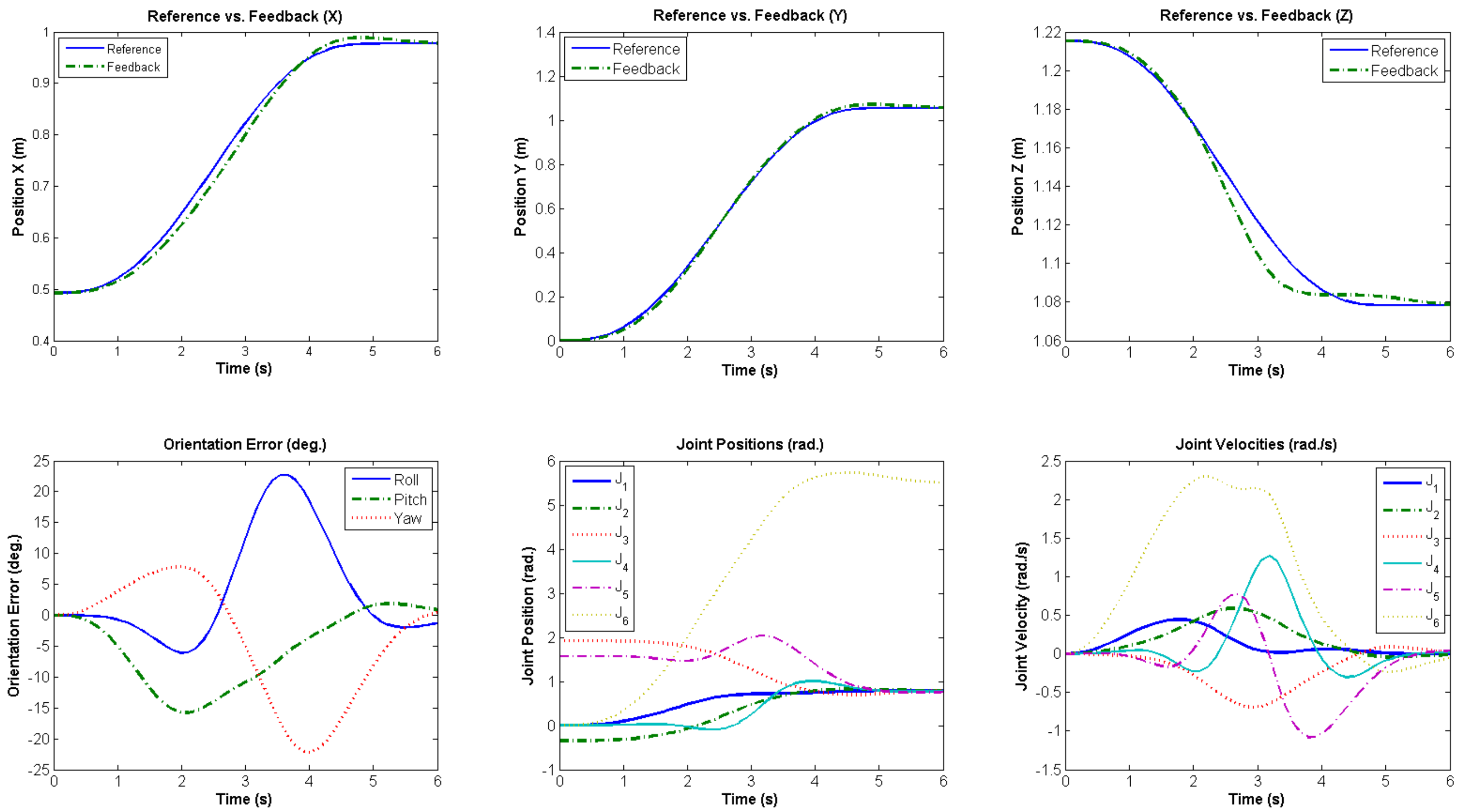


Figure 3.4 : Simulation results for $k_p = 9, k_d = 6$.

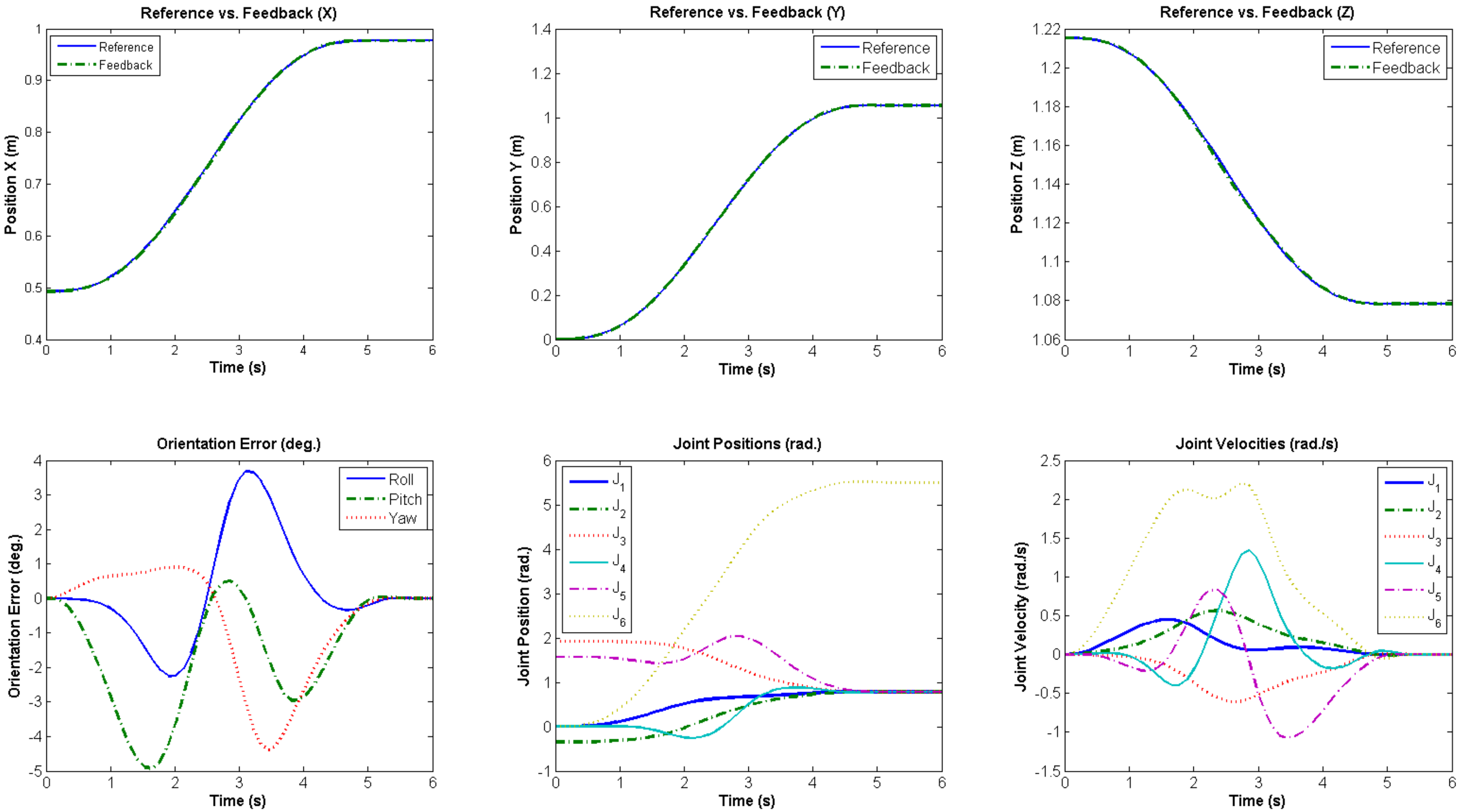


Figure 3.5 : Simulation results for $k_p = 100, k_d = 20$.

3.2. Slave Manipulator

3.2.1. Overall architecture

Basic block diagram for the Slave manipulator is shown in Figure 3.6. Gravity compensation is accomplished once again using Newton-Euler algorithm. Output torque is then combined with torque due to joint two torsional spring. Finally, force control signal is added.

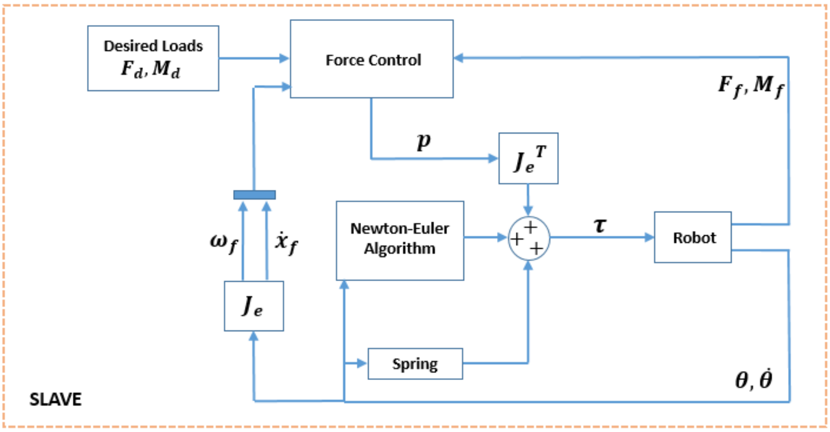


Figure 3.6 : Slave manipulator block diagram.

3.2.2. Simulation

Above model was modified to better test response of the system. To simulate presence of the master manipulator, trajectory generator was added. Trajectory position and orientation errors between desired and feedback signals is used to generate loads. Modified diagram is shown in Figure 3.7.

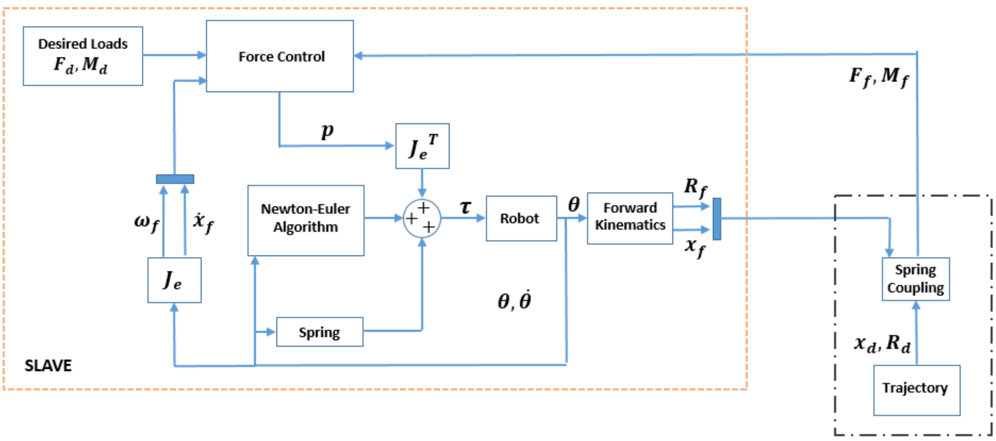


Figure 3.7 : Modified slave manipulator block diagram.

Spring coupling generates forces and moments using ideal linear and torsional springs with stiffness k_e and k_t. Spring model, shown in Figure 3.8, uses Axis/Angle orientation error to generate moments due to end effector orientation shift.

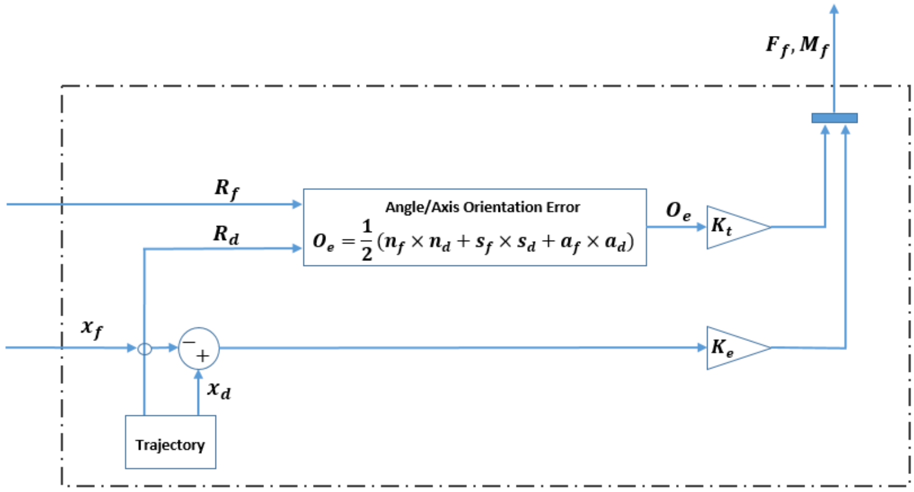


Figure 3.8 : Spring coupling model.

Force control of the slave manipulator is shown in Figure 3.9. It utilizes force/moment feedforward, proportional action acting on the force/moment error, and the derivative action on the feedback velocities. Force/moment derivatives are not used as the original signals are usually noisy.

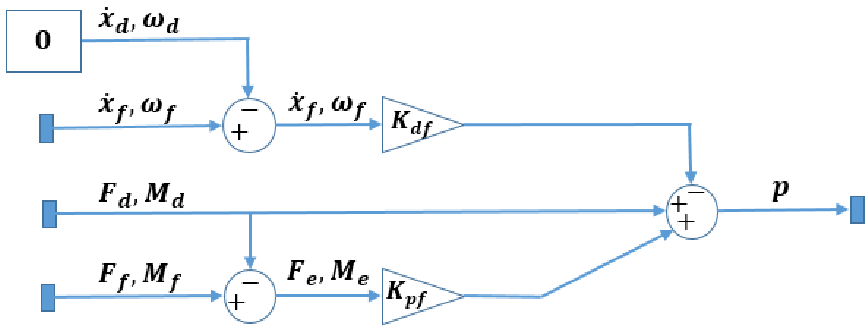


Figure 3.9 : Slave manipulator force control diagram.

Following parameters were used to simulate the system. Force gains were set at $k_{pf} = 9$ and $k_{df} = 6$. Stiffness for the linear and torsional springs were set at $k_e = 200\ N/m$ and $k_t = 5\ Nm/O_e$. respectively. It is easy to see, with desired loads set to zero, that the spring stiffness act as additional proportional gain. Table 3.2 lists start and end points of a trajectory along global x direction. Trajectory was generated using quantic polynomial in task space. Results of the simulation are resented in Figure 3.10.

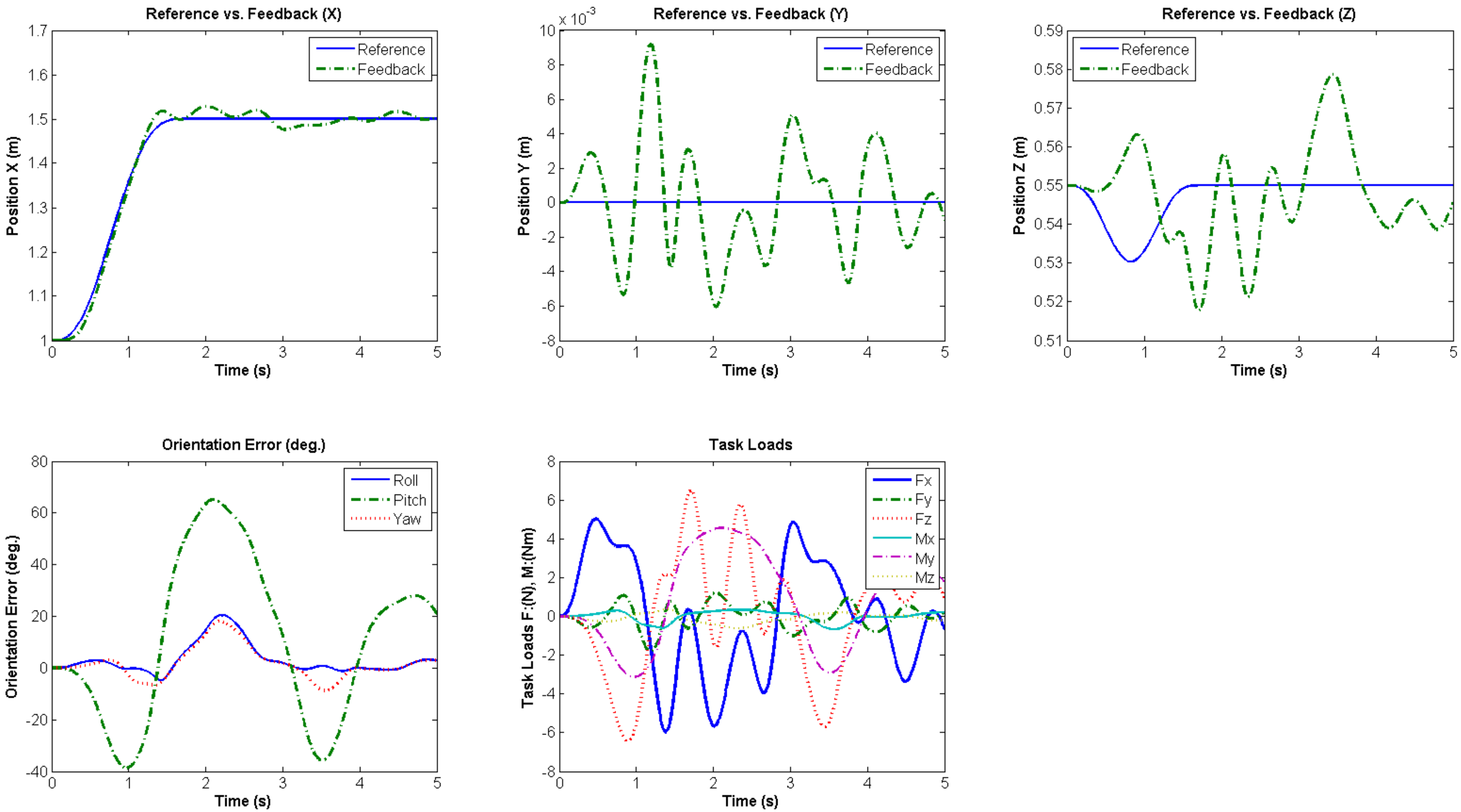


Figure 3.10 : Slave manipulator force response for $\boldsymbol{k_{pf} = 9, k_{df} = 6}$ with linear and torsional spring coupling.

Table 3.2 : Reference trajectory data for the slave manipulator.

Point	Task Position (m) and Orientation (°)						Time (s)
	x	y	z	Roll	Pitch	Yaw	
Start	1	0	0.55	0	90	0	0
End	1.5	0	0.55	0	90	0	1.688

While system response in terms of position looks relatively good, orientation is not satisfactory. There is a large orientation error that continues to persist through out timeframe of the simulation.

System gains were later adjusted to improve performance. Simulation with $k_{pf} = 100$ and $k_{df} = 20$ was performed. Results are shown in Figure 3.13. This time, we see much better performance in terms of following the trajectory. While positional and orientation errors are much lower, orientation still oscillates around desired value. One thing to notice it the sensitivity of the system. It takes less than a Newton to effectively reposition the Slave. It may be the case that the gain levels used in the simulation are too high for a practical application.

3.2.3. Orientation control

Previous example illustrated modeling of a system using spring coupling. Experimental implementation of this setup is much more challenging. While linear spring model is a relatively good representation of an actual compression spring, there is no simple setup that would correspond to a 3D torsional spring used in the simulation. More robust model of the actual slave robot was required.

Oftentimes, it is desirable to control end effector`s orientation. Number of operations require specified orientation. With this in mind, an orientation control loop was added to the slave manipulator. The control subsystem is identical to one found in the Master design illustrated earlier. Again, simple PD control as per Figure 3.2 was utilized in this task. Here, desired angular velocities and angular accelerations were set to zero. Block diagram of the complete system is portrayed in Figure 3.11. Trajectory generator is used to provide desired orientation at any instant of timer. It is then combined with the feedback orientation to generate orientation error. Orientation control generates corrective signal such that the existing error is driven to zero.

This time, stiffness of the torsional spring k_t was set to zero with orientation gains set at $k_{po} = 100$ and $k_{do} = 20$. Results obtained during the simulation are included in Figure 3.14 and are comparable with the ones obtained during simulation with torsional springs coupling.

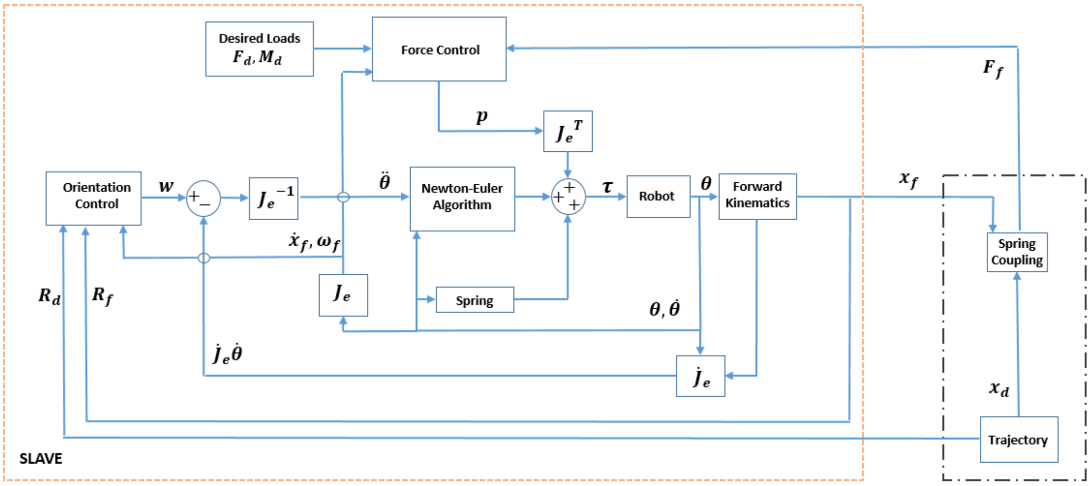


Figure 3.11 : Slave manipulator with orientation control.

3.2.4. Load cell noise

In general, load cell signals are often quite noisy due to signal amplification. Generic illustration is given in Figure 3.12. To simulate potential noise in the feedback signal, white noise was added to the feedback loop investigated above. Matlab Band-Limited White Noise block was used to simulate the disturbance using noise power of 0.00005 and simulating at a sample time of 0.004 s. Results are shown in Figure 3.15. The effect of added noise at the stated power level appears to be negligible in terms of following specified trajectory at the meter, or centimeter scale. When inspecting response along the Y direction, oscillation at the millimeter scale is more pronounced. In general, signal noise at this level does not appear to cause significant problems.

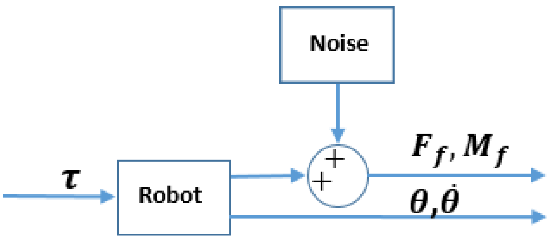


Figure 3.12 : Load cell noise.

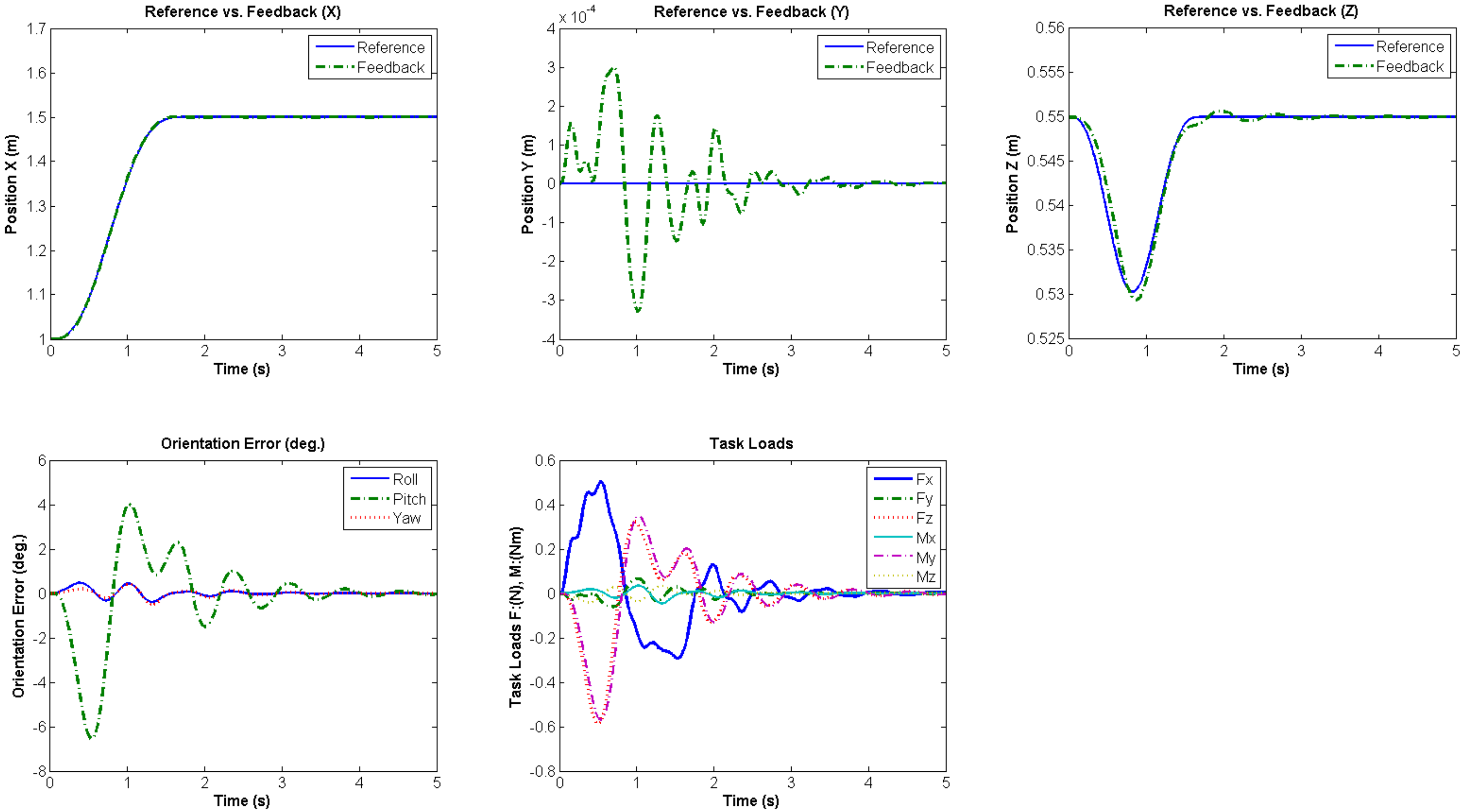


Figure 3.13 : Slave manipulator force response for $k_{pf} = 100, k_{df} = 20$ with linear and torsional spring coupling.

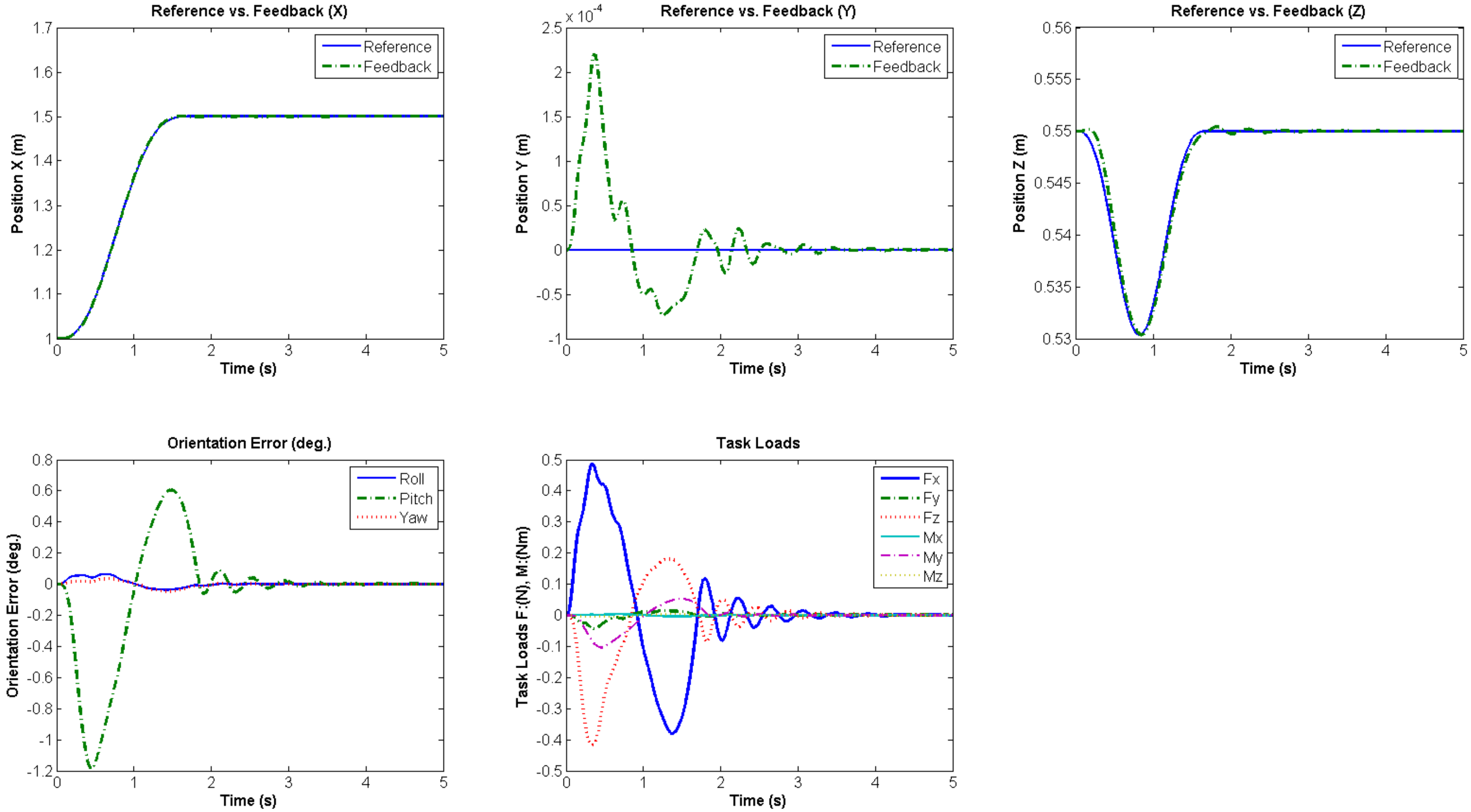


Figure 3.14 : Slave manipulator with orientation control, $k_{po} = 100, k_{do} = 20$, and linear spring coupling.

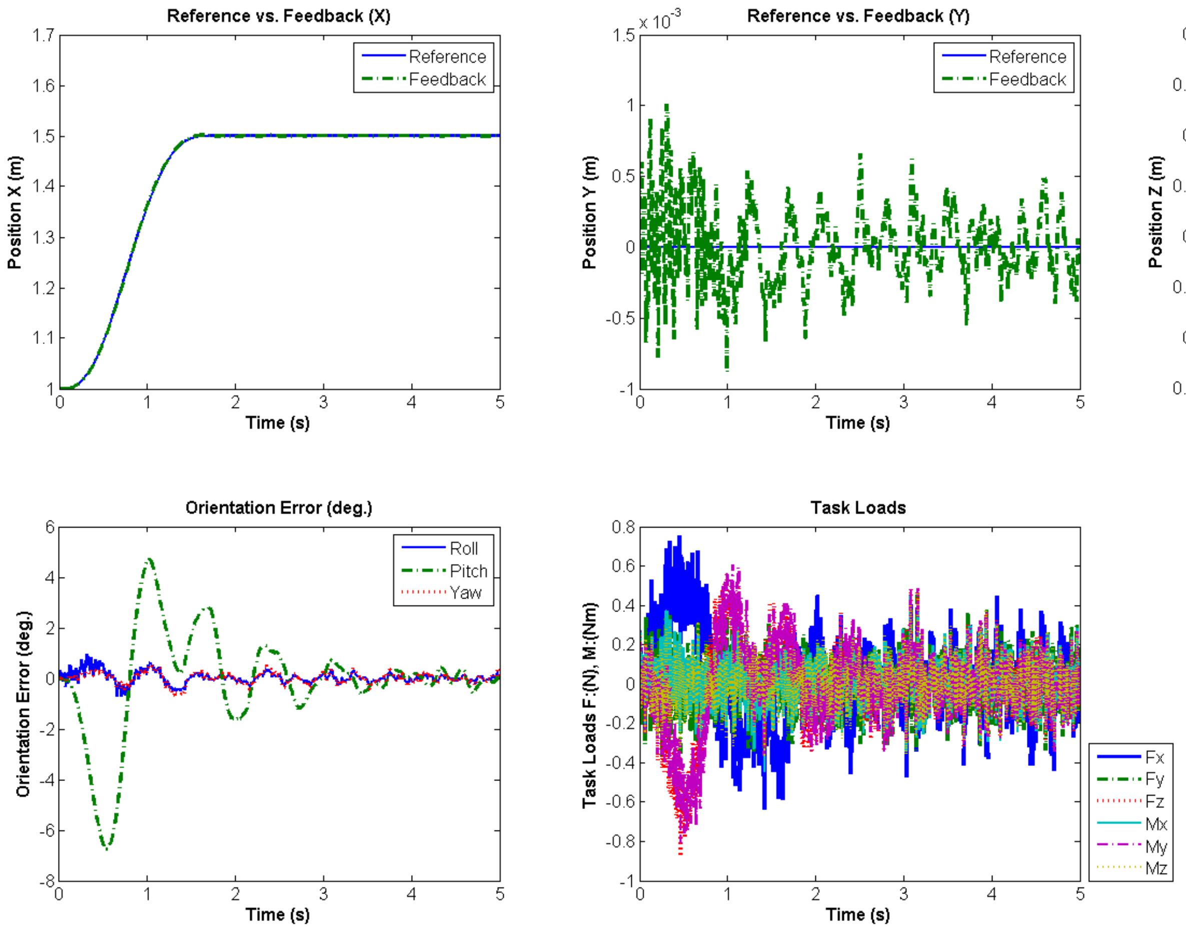


Figure 3.15 : Effect of Load Cell Noise.

3.3. Master-Slave System

Combined block diagram for the Master-Slave system is shown in Figure 3.16. Here, input to the Slave is generated due to positional and orientation error between both end effectors. Again, spring coupling between both subsystems models linear and torsional springs generating forces and moments. No explicit orientation control is performed. Slave manipulator tries to maintain orientation of the master using just force controller acting on the feedback moments generated through spring coupling. In addition, reaction loads experienced by the Master are accounted for, and canceled out.

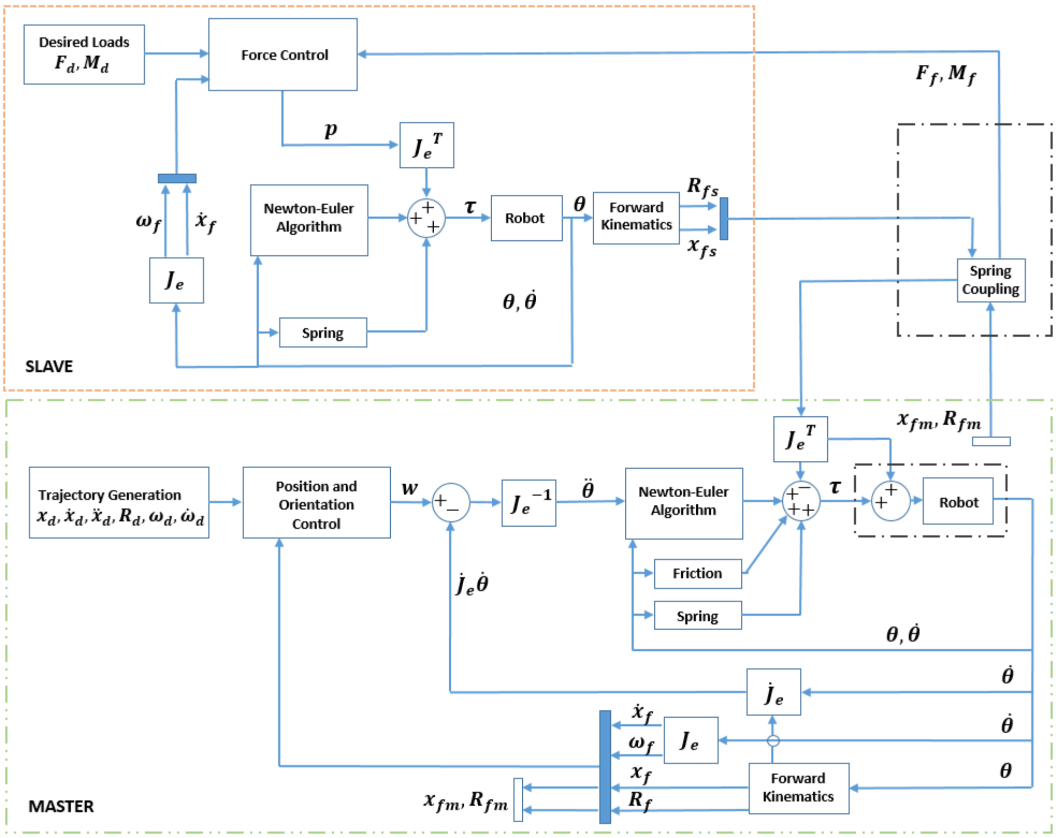


Figure 3.16 : Master-Slave block diagram.

Simulation results for the trajectory specified in Table 3.1 are shown in Figure 3.17. While master manipulator follows reference almost exactly, slave manipulator exhibits some oscillations around followed trajectory. Again, position and orientation gains for the Master as well as the force gains for the slave, are: $k_p = k_{pf} = 100$ and $k_d = k_{df} = 20$. Spring coupling uses same parameter values of $k_e = 200\ N/m$ and $k_t = 5\ Nm/0_e$.

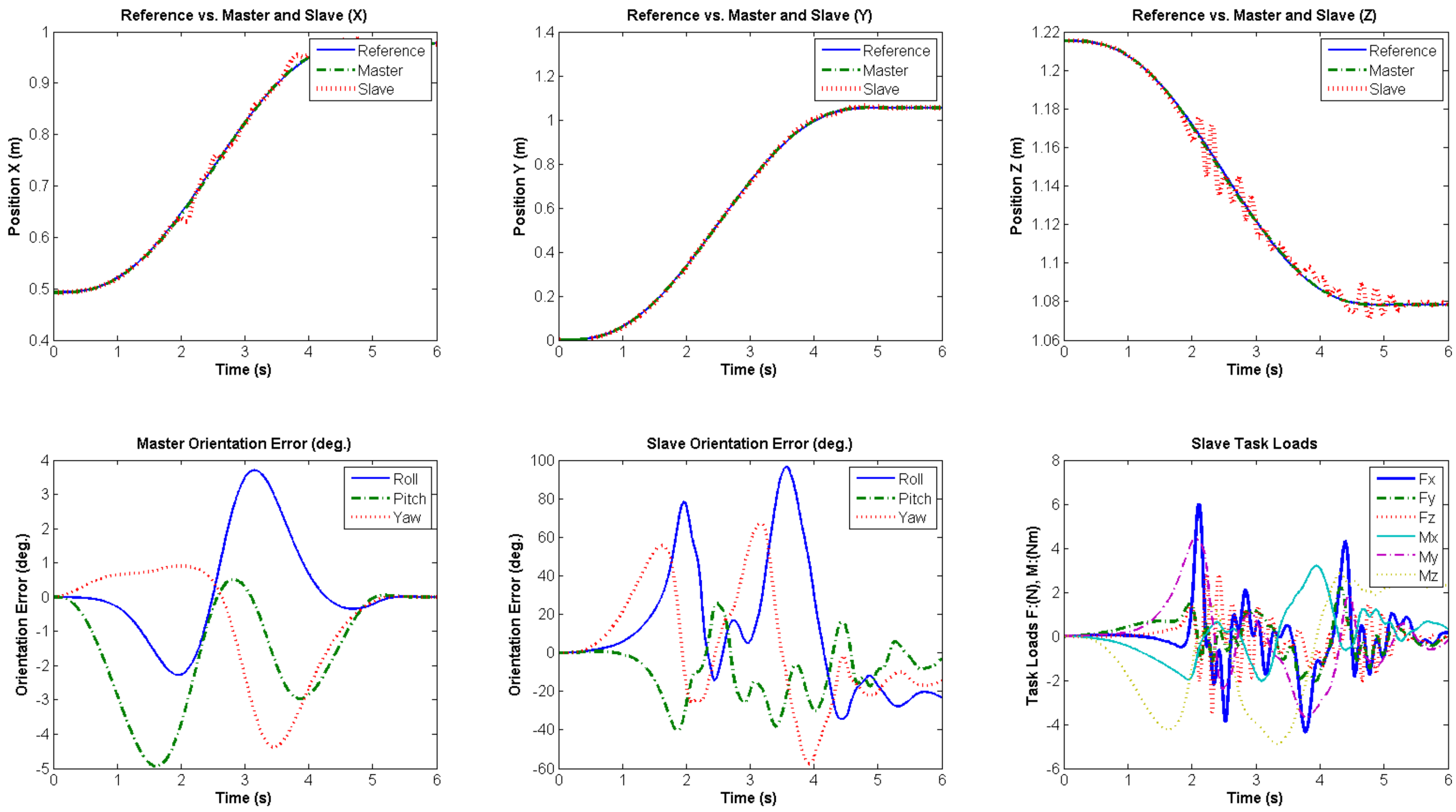


Figure 3.17 : Master-Slave system response for trajectory specified in Table 3.1.

More complicated motion summarized in Table 3.3 is also simulated and presented in Figure 3.18 and in Figure 3.19 using identical simulation parameters. This time a five-point trajectory was constructed. Relatively large time intervals between individual points were set. The results show very good trajectory tracking by the Master as well as the force tracking of the Master by the Slave. Positional error at the terminal point is on the order of a few millimeters. Similarly, orientation error at the trajectory end in only a few degrees. Spikes in the orientation error of the Master, and in a lesser degree those of a Slave, correspond to intermediate trajectory points.

Table 3.3 : Reference trajectory data for Master – Slave system.

Point	Task Position (m) and Orientation (°)						Time (s)
	x	y	z	Roll	Pitch	Yaw	
1	1	0	0.55	0	90	0	0
2	1.5	0	0.55	0	90	0	2
3	1	0.5	1	90	0	0	7
4	0	1	1	90	0	0	12
5	1	0	0.55	180	0	0	15

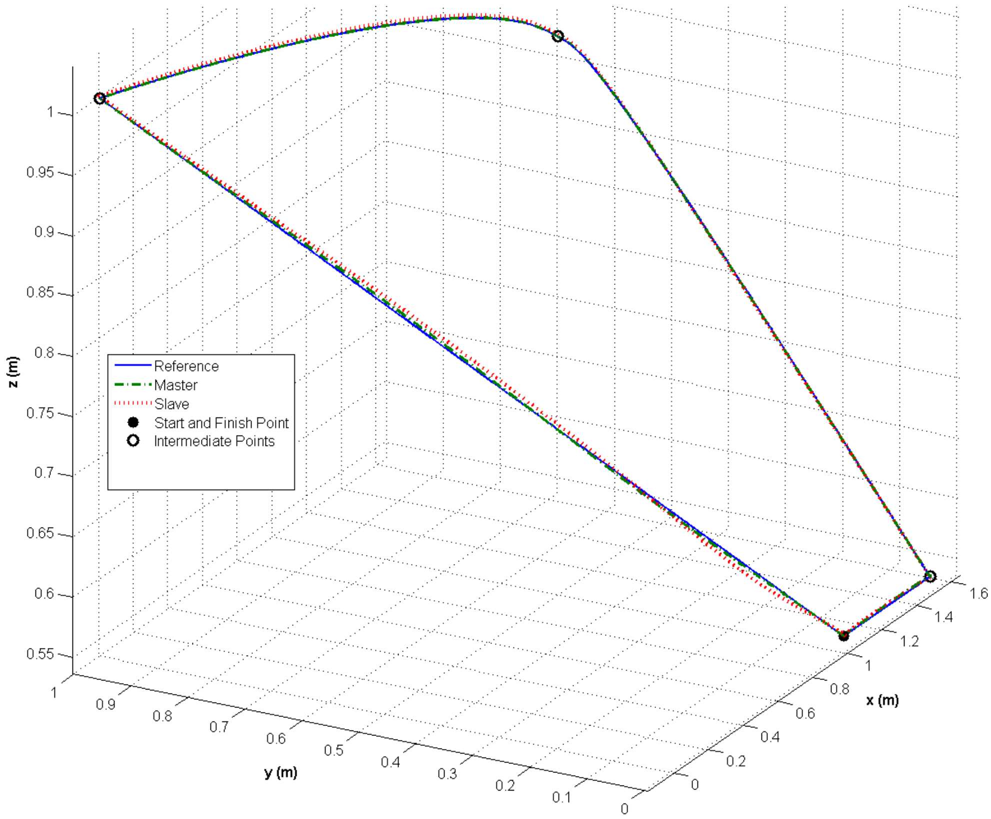


Figure 3.18 : Reference vs Master and Slave manipulator trajectories for Table 3.3.

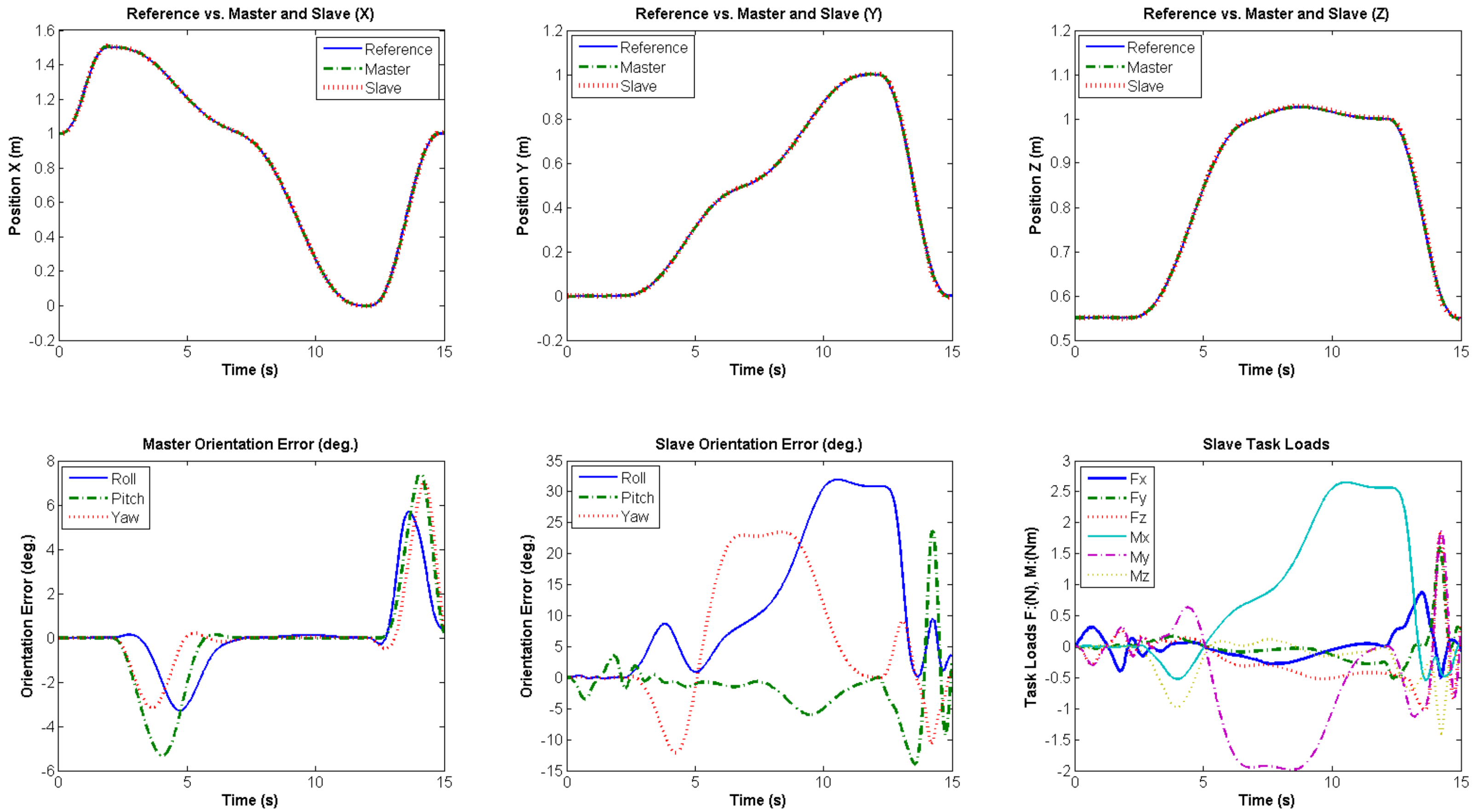


Figure 3.19 : Master-Slave system response for trajectory specified in Table 3.3.

4. EXPERIMENTAL WORK

Modeling phase allowed for the implementation of all major components of the control structure. It also provided guidance. Once complete, it was ported on to the actual hardware. Individual elements were tested and evaluated. Required modifications and solution to problems encountered during the work were performed. Experimental work done during this phase is presented below.

4.1. Torsional Spring

Torque generated by the torsional spring is significant, and is critical in the cancelation of the dynamic terms as require by the feedback linearization. While the spring exhibits linear behavior in the proximity of zero, it is actually a nonlinear term. It is thus important to have a good torque curve characterizing dynamics of this component.

While all three robots are of the same series and are virtually identical, their dynamic response was found to vary between individual units. Torsional spring embedded in joint two of the second manipulator was found to exhibit different behavior than the one found in the first unit. When using torque curve of robot 1, the response of the second robot was not symmetrical across the zero-degree position of joint two. It was shifter toward one side. Characterization of the spring torque of the second unit was required and was performed in the operational range from $-100°$ to $+100°$. Joint two of the manipulator was rotated and every $5°$ static torque feedback required to support the weight of the robot was logged. This was compared with the analytical calculations and the difference between two values was recorded as torque generated by the spring. Figure 4.1 illustrates results of the characterization. Collected raw data, as well as the torque curves of unit 1 and 2 are shown. Equation of the fitted torque curve was found to be:

$$\tau_s(\theta_2) = -31.102 * {\theta_2}^3 + 5.837 * {\theta_2}^2 + 430.568 * \theta_2 - 3.337 \qquad \textbf{(4.1)}$$

This relatively low order polynomial function provides a good fit for the collected data.

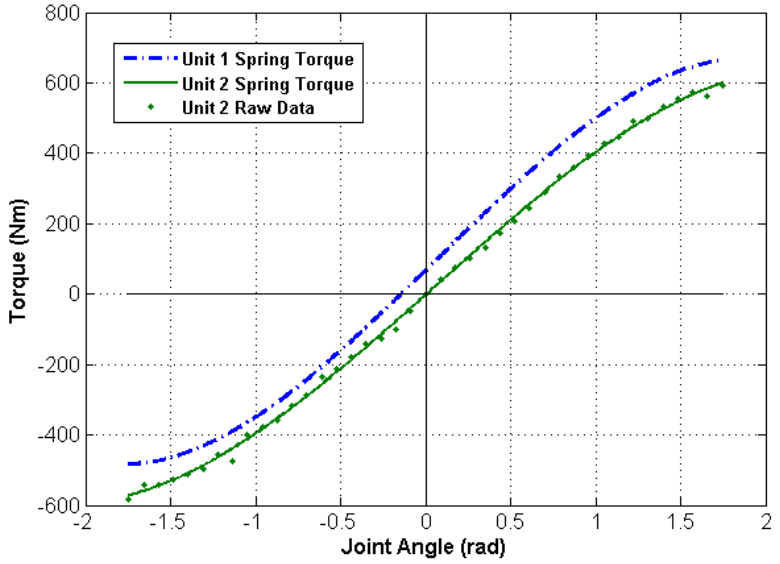


Figure 4.1 : Spring torque curve characterization.

4.2. Feedback Signal Filtering

Introduction of noise into the velocity feedback signal resulted in failure of the simulation to converge. This foretold problem encountered during the experimental phase. Figure 4.2 shows joint velocities of all six joints for a test conducted without any signal processing. This raw signal, especially for the first three joints, is extremely noisy. This proved to be a tremendous problem. At many occasions, robot would simply shutdown as high frequency oscillations were induced in the arm. It became clear that signal filtering of the velocity feedback was required. A simple, one dimensional Kalman filter was chosen and applied individually to all six joints. By controlling process noise coefficient q and sensor noise coefficient r, tradeoff between signal conditioning and resulting signal lag could be achieved. Figure 4.3 and Figure 4.4 illustrate effect of these coefficients on the joint velocity signal using same test parameters as the original test presented in Figure 4.2. Low value of r, or high values of q resulted in high frequency oscillations or chattering. High values of r, or low values of q induced noticeable lag and low frequency oscillations. Calibration resulted in selection of the final values for the coefficients as $q = 0.125$ and $r = 20$. While present, signal lag proved to be of negligible effect of the performance of the control in terms of tracking desired trajectory.

While response of the force control to a noisy force signal proved to be much more robust, same filter was applied to load cell raw output.

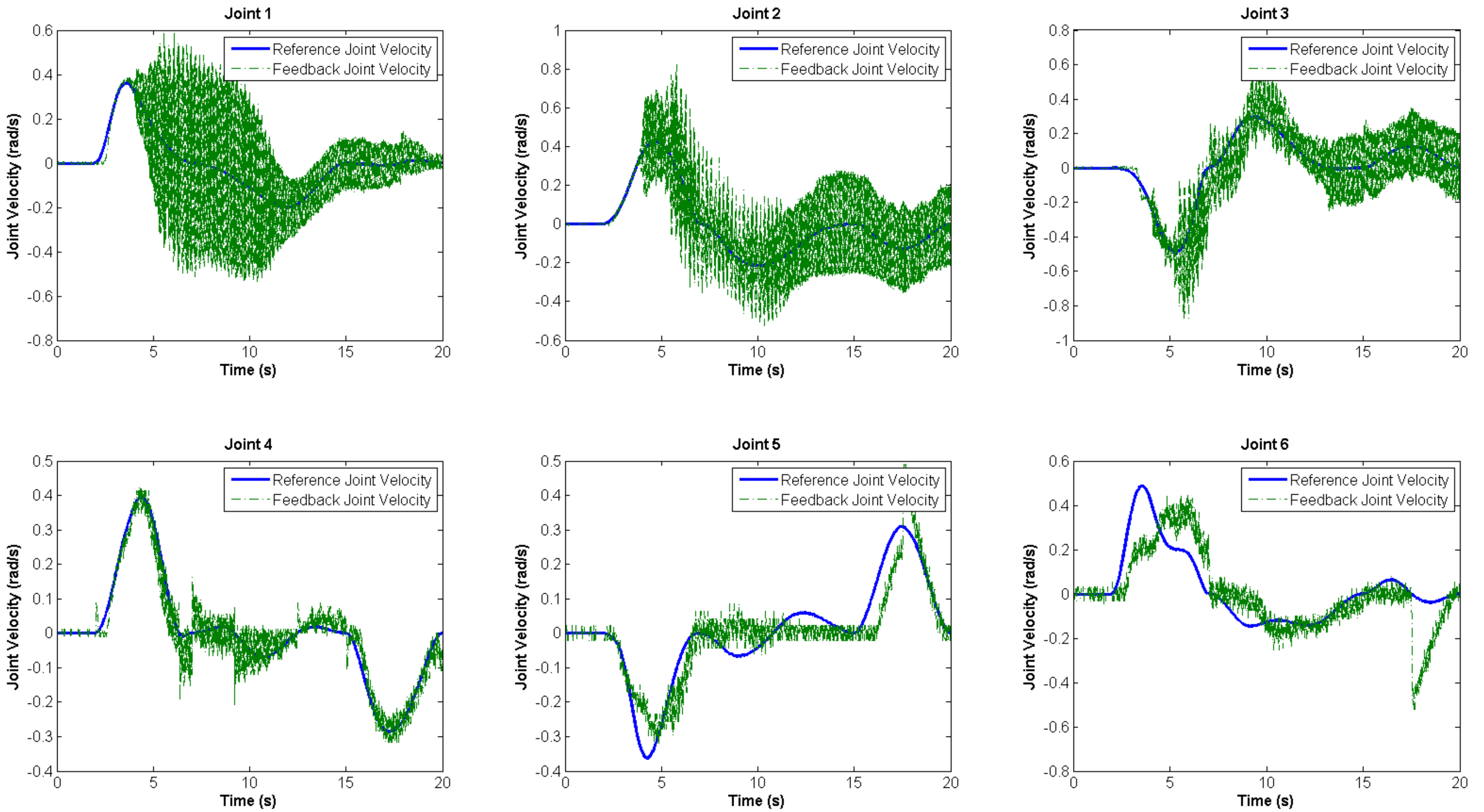


Figure 4.2 : Unfiltered joint velocities.

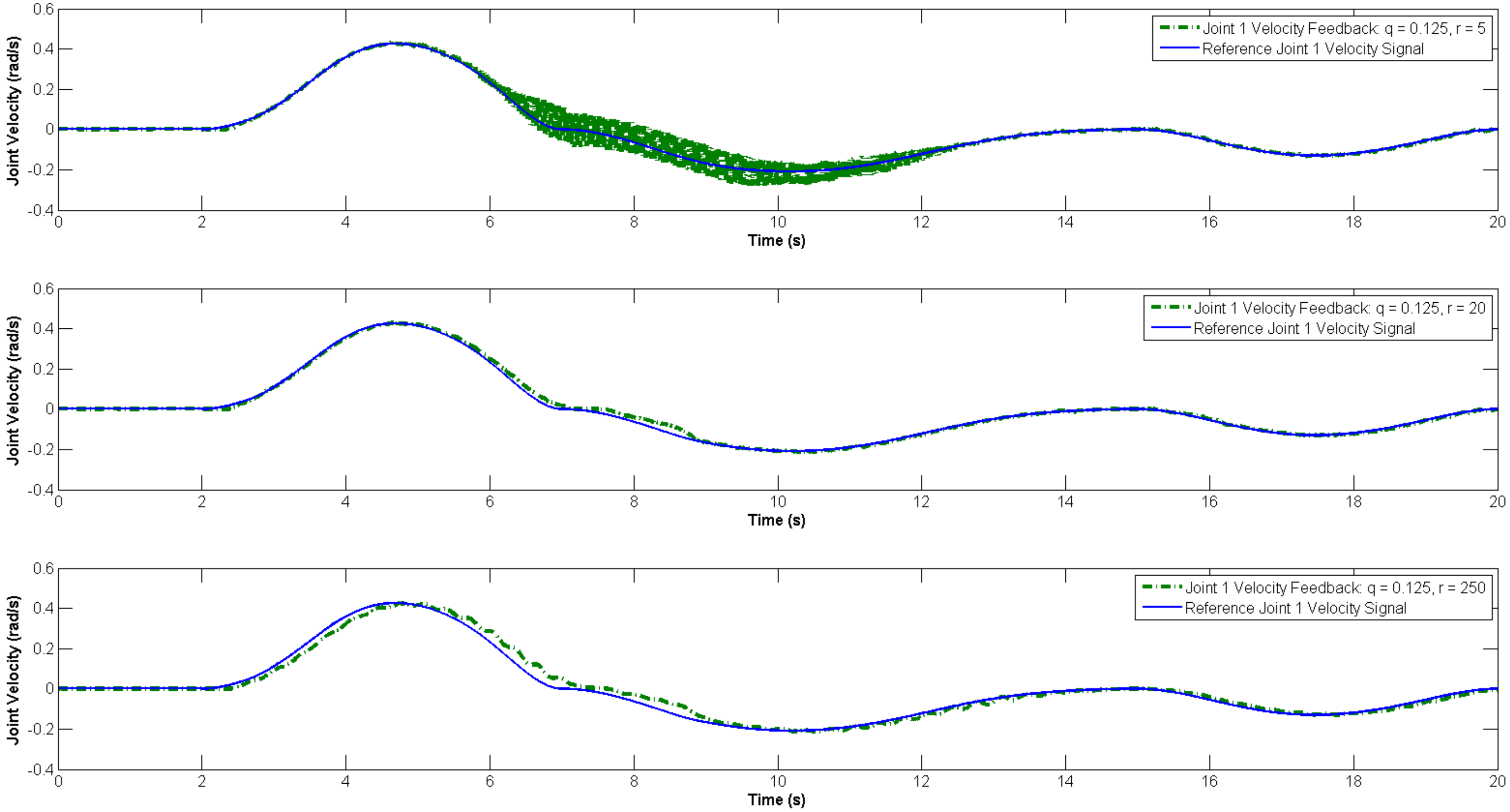


Figure 4.3 : Effects of coefficient r.

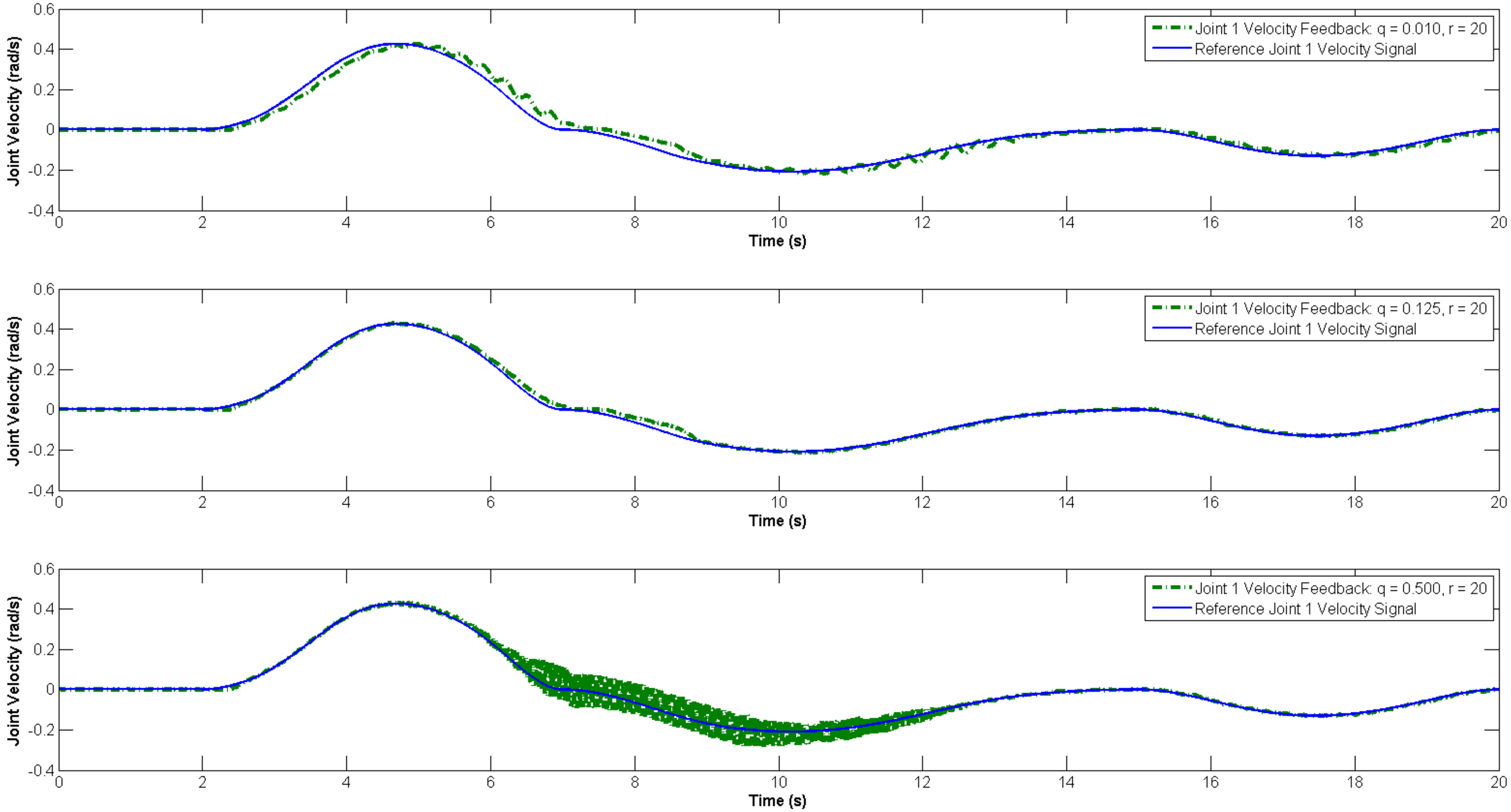


Figure 4.4 : Effects of coefficient q.

4.3. Gripper, Load Cell, and Dead Zone

The attachment of the gripper at the end of the manipulator introduced substantial loads that had to be accounted for in the dynamics of the arm. Moreover, forces and moments generated by the weight of the gripper had to be canceled out from the load cell reading. This offset was dependent on the orientation of the end effector. A simple algorithm was implemented to solve this problem.

Initial tests revealed that the gripper mass compensation was not sufficient. Load cell would still generate substantial and anomalous readings. Upon closer inspection, custom tool frame was found programmed on the controller. Once deleted, load cell proved to produce much better feedback data. Hoverer, some additional erroneous effects still persisted. While relatively small, an attempt was made to eliminate this effect. Dead zone around origin was implemented to suppress undesired force and moment readings. Based on the power law and characterized by the three inputs: $a, b,$ and c, as per equation 4.2, it was integrated with the control architecture and accessible online through LLI interface. Dead zone is shown in Figure 4.5. While useful, this feature proved to be a blunt tool in terms of dealing with undesired effects.

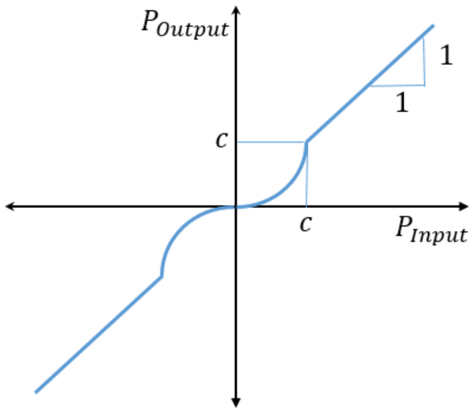


Figure 4.5 : Dead zone.

$$P_{Output} = a * {P_{Input}}^{b} \quad \forall \quad -c < P_{Input} < c \qquad \textbf{(4.2)}$$

Ultimately, tests revealed an internal mass associated with the load cell itself. With nothing mounded, load cell would generate a reading that, as in the case of the gripper, changed with the orientation. Load cell data were collected at various orientations to estimate this mass. Moment output due to this effect was negligible and the mass was assumed to be located at the base of the load cell-gripper interface. Identification of this effect for both of the manipulators was done successfully. Once characterized, the internal mass was combined with the mass of the gripper and the center of mass of the gripper was adjusted. Subsequent tests with the gripper revealed negligible load cell readings verifying implementation of this approach.

4.4. Master Manipulator

Final architecture and control of the Master manipulator is shown in Figure 4.6 and Figure 4.7. The block diagram incorporates some of the features discussed previously. These include signal filtering, gripper compensation, and dead zone. Likewise, a new element is included. Inverse kinematic engine is incorporated as an optional source of signal to the Newton-Euler algorithm.

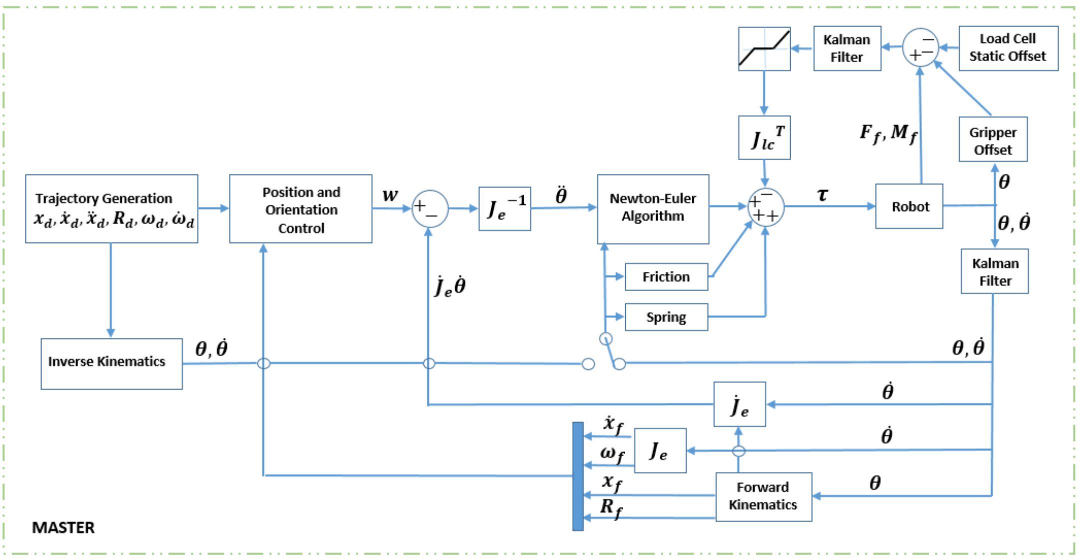


Figure 4.6 : Final Master manipulator architecture.

Control schematic has been updated and includes integral control for both position and orientation, as well as an alternative selection for the orientation error. This option is accessible from the gain submenu in the main user interface and can be readily applied during experiments. Quaternion formulation has been employed as a possible solution for the persistent orientation error encountered during testing. Some of these features are discussed in more detail on subsequent pages.

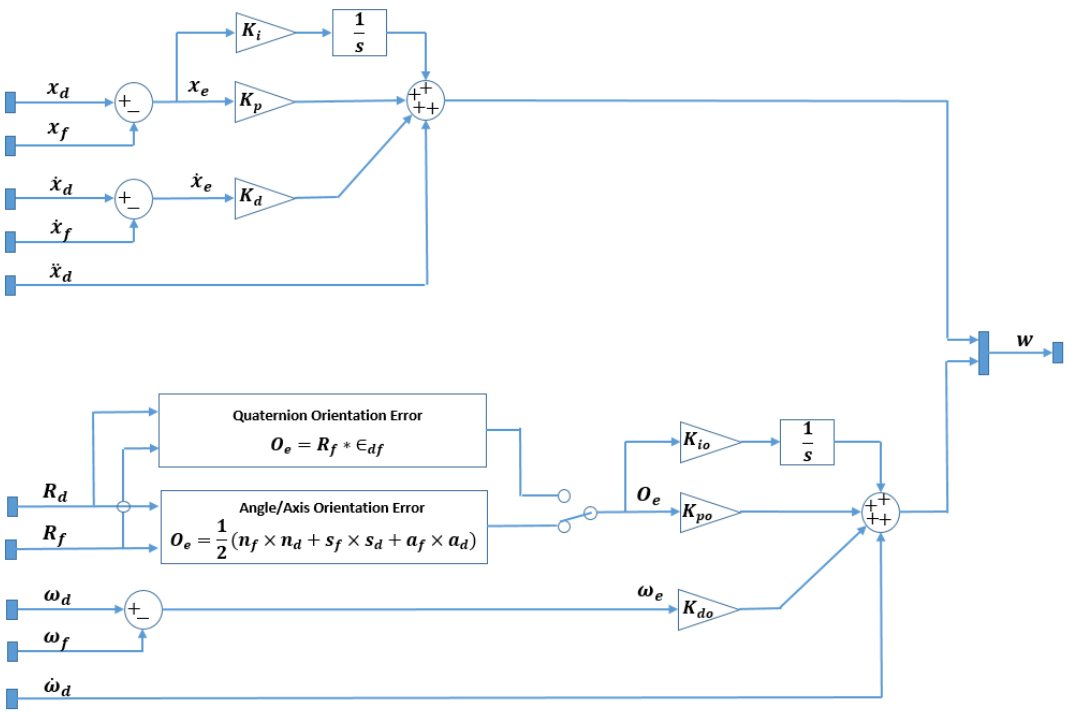


Figure 4.7 : Master manipulator position and orientation control diagram.

4.4.1. Friction

Friction model comes from Waiboer (2007) and friction parameter identification from work of Zengin (2013). This was later modified by Akbaş (2015). When implemented, a significant problem was identified. Static friction on joint 4 was too large. With no explicit command, joint 4 would move once the robot was enabled. While no significant work was done to verify borrowed friction model, a small modification was employed. Asperity friction torque coefficient was multiplied by a constant to bring magnitude of the initial torque down to a level where non-commanded motion would no longer take place. Value of this constant was found to be approximately equal to 0.6. Figure 4.8 illustrates torque required to overcome friction for each joint as per original parameters found in Akbaş (2015) and modified version.

Another interesting observation arising during initial testing concerned performance of the fifth and the sixth joint. While testing without friction model, basic positional tracking could still be obtained. Large inertia of the first four joints allowed for effective motion. Orientation tracking however was practically nonexistent. Joint five and six would not move at all. With such small inertias, generated dynamic torque was insufficient to actuate these two links. Finally, when the friction model has been incorporated, orientation control was achieved. While sufficient for the present work, orientation tracking still remains an open problem. Further work may be necessary for better control. Verification of the friction model may be necessary, especially for the last two joints and the complication arising from their coupling.

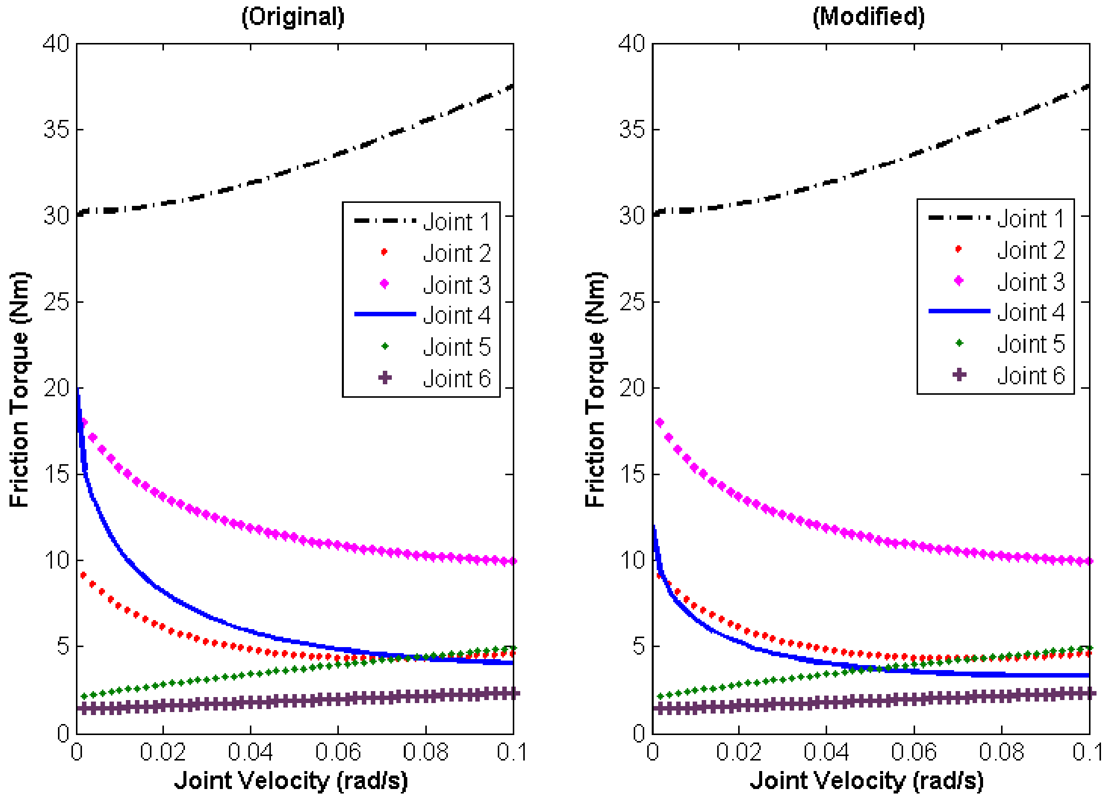


Figure 4.8 : Original and modified friction torque curves.

4.4.2. Gain tuning

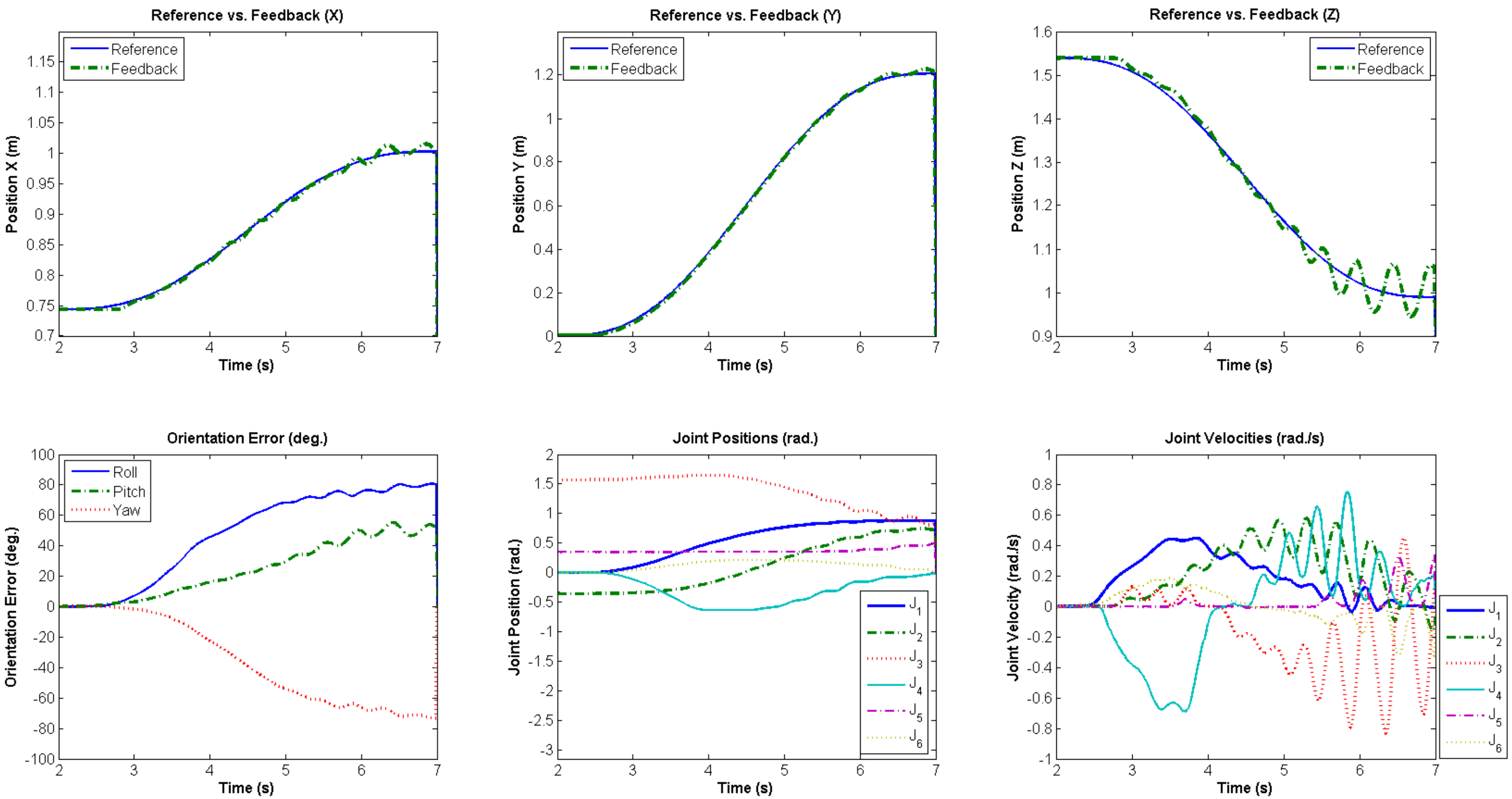
While simulation studies allowed for a crude estimation of gains, proper testing provided reliable numerical values. Ziegler – Nichols method was used during evaluation. Beginning with the proportional gain and orientation gains set at zero, the value was raised until substantial oscillations in the response were present. This value was recorded as k_u and the proportional gain k_p was set at $0.8 * k_u$. Oscillation period T_u was used to evaluate required derivative gain as equal to $k_p * T_u/8$. Calculations resulted in values of $k_p = 400$, and $k_d = 25$. Figure 4.9 presents response of the system at the gain value of k_u and induced oscillations with period of T_u.

The above procedure was then amended with additional fine tuning. Staring with base values of k_p and k_d indicated above, gains were varied and results recorded using two metrics. These were the magnitude of the final task position error vector as well as the summation of the error vector over the whole trajectory. Table A.1 indicates trajectory, and Table A.2 lists all recoded values for different k_p and k_d combinations. Experiments with noticeable oscillations and, or chattering, were discarded. Ultimately, pair $k_p = 500$ and $k_d = 35$ was selected as the best overall performer. Interestingly, presence of k_d allowed for proportion gain to reach k_u without previously seen oscillations.

Once positional gains were set, evaluation of orientation gains k_{po} and k_{do} took place. Again, gains were changed and the resulting evaluation metrics were recorded as per Table A.3. It has been found that increasing k_{po} above value of about 1000 resulted in unstable behavior.

In general, when the orientation gains are increased, positional accuracy suffers. Whereas pure positional control can approach final position within a few millimeters, addition of orientation results in accuracy on the order of a few centimeters. Depending on the particular application, orientation accuracy could be prioritized and the accuracy increased at the expense of the position error. Eventually, ability to effectively reach designated position was chosen as the primary objective. Selection of the orientation gains was completed and the default gains were set to $k_{po} = 200$ and $k_{do} = 70$.

To further improve trajectory tracking, integral control was added to both position and orientation loops. Selected results are given in Table A.4. Different values of integral gain were tested while the total integral error was limited to unity to prevent integral windup. Addition of integral gain k_i increased positional tracking accuracy from baseline performance, in terms of final error, from 2.8cm down to about 4mm. This was achieved without noticeable loss of orientation accuracy. Introduction of the orientation integral gain decreased total orientation error about 29% going from about 281 down to 199. This however was done at the expense of positional accuracy. Furthermore, the results here are somewhat ambiguous. While the total orientation error over the whole trajectory decreased, final error, with the exception of the last test using $k_{io} = 1000$, did not. When both of the integral gains were finally tried simultaneously, a noticeable improvement in all metrics was recorded when using high values in the range of 500-1000.



.

Figure 4.9 : System response at $k_u = 500$ with $T_u \approx 0.5s$.

The last comprehensive test involved more complicated trajectory outlined in Table 4.1. In addition, gain values estimated during simulations as well as the Ziegler – Nichols gains obtained previously, were tested.

Table 4.1 : Five point task trajectory expressed in terms of joint angles.

Point	Joint Position (°)						Time (s)
	Joint _1	Joint _2	Joint _3	Joint _4	Joint _5	Joint _6	
1	0	-20	90	0	20	0	0
2	0	-20	90	0	20	0	2
3	45	45	45	45	45	45	7
4	0	0	90	45	0	45	15
5	0	-20	90	0	20	0	20

Test data values are captured in Table 4.2. Not all tests proved to be successful. Experimental tests using high integral gains, which initially provided good results, failed. Manipulator would try to attain orientation loosing positional tracking. As errors increased so did speed and required torque. This was further amplified by the torque safety limits. Arm would automatically shut down as some of the joints exceeded their envelopes. In some cases, manual e-stop was pressed once it was realized manipulator went off track. Graphs of position and orientation errors for completed tests are presented in and Figure 4.10.

In general, baseline results of fine PD tuning provided good positional and fair orientation tracking. Addition of integral gain in the position control loop appears to have decreased total positional error. The final position as well as the orientation error remained comparable. While the final orientation error of the Ziegler – Nichols tuning was low, other metrics were much higher than the baseline test. Gains estimated during simulation phase display worst overall performance. This is the case of incomplete modeling of the system. Friction is primarily responsible for this result.

Additional tuning of integral control for the orientation remains to be done. Effects of integral limit, proportional, derivative and integral terms need to be explored in greater detail. Friction and coupling of the fifth and sixth joint has to considered.

The selection of gains presented above is limited to a relatively low velocity range used during experiments. Different applications may require re-tuning and optimization.

Table 4.2 : Position and orientation errors for different gain values.

kp	kd	ki	kpo	kdo	kio	Task Position Total Error (m)	Task Position End Error (m)	Orientation Total Error	Orientation End Error	Notes
100	20	0	100	20	0	296.2609	0.0094	1353.70	0.3306	Simulation Estimate
400	25	0	400	25	0	227.4238	0.013	1064.90	0.096	Ziegler – Nichols Tuning
500	35	0	200	70	0	97.0997	0.0032	1083.20	0.2043	PD Fine Tuning
500	35	1000	200	70	0	29.4048	0.0056	1086.60	0.1764	Integral Tuning
500	35	0	200	70	1000	Failed Test				Integral Tuning
500	35	1000	200	70	1000	Failed Test				Integral Tuning
500	35	1000	200	70	500	Failed Test				Integral Tuning

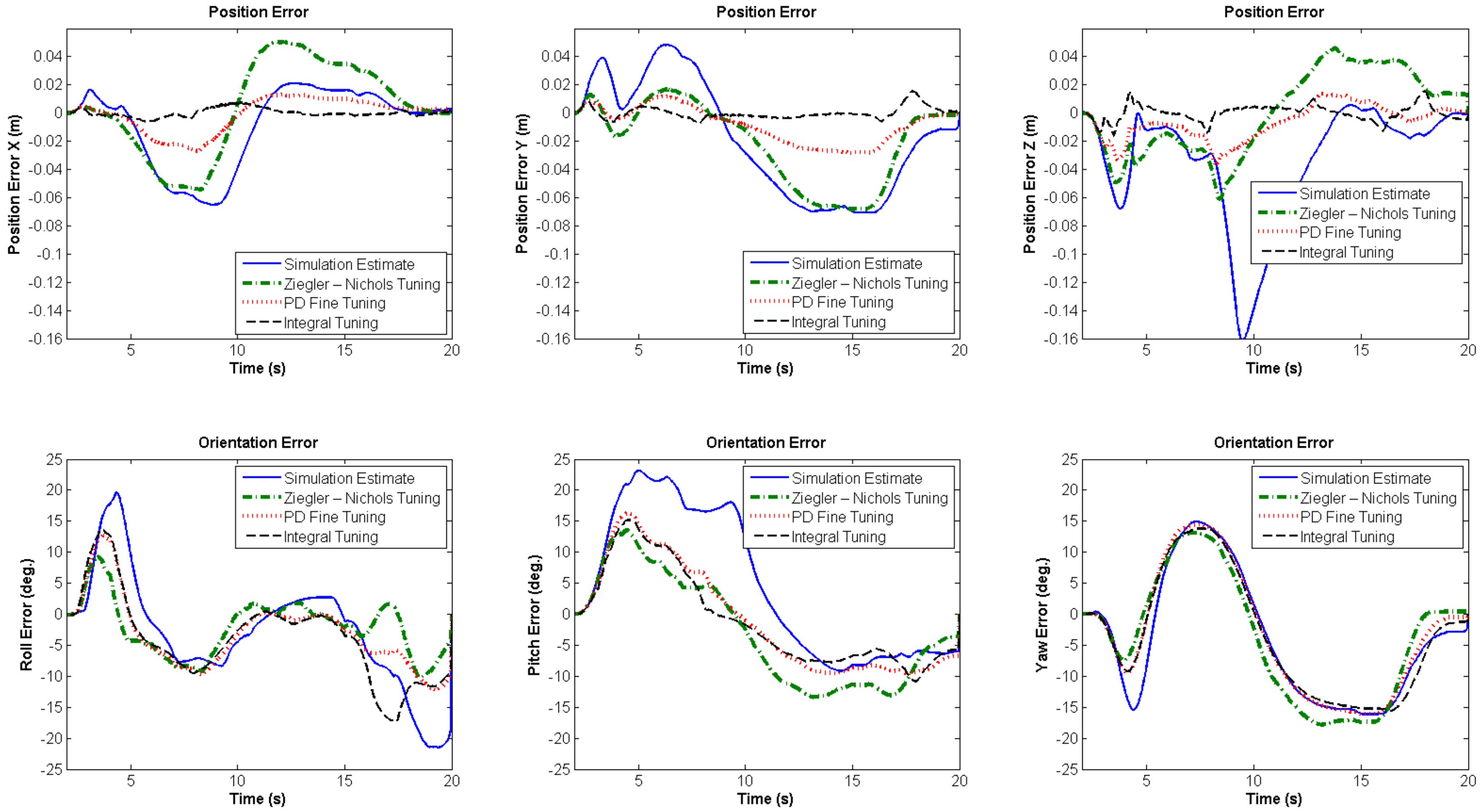


Figure 4.10 : Position and orientation errors for different gain sets.

4.4.3. Inverse kinematics

LLI inverse kinematic engine has been included in the master manipulator control architecture as per Figure 4.6. This low level function was used to replace feedback going into Euler-Newton algorithm. It was hopped that the cleaner signal would improve tracking performance. Testing has been conducted to verify this hypothesis. Table 4.3 illustrates results while tracking trajectory provided in Table 4.1 and using previous gain values.

Table 4.3 : Comparison of error between regular and inverse kinematics feedback.

Task Position Total Error (m)	Task Position End Error (m)	Orientation Total Error	Orientation End Error	Notes
97.0997	0.0032	1083.20	0.2043	PD Fine Tuning
97.2124	0.0035	1079.80	0.2057	PD Fine Tuning + Inverse Kinematics
29.4048	0.0056	1086.60	0.1764	Integral Tuning
29.4446	0.0048	1081.50	0.1795	Integral Tuning + Inverse Kinematics

Clearly, results are almost identical for both tested configurations. This may be the result of Kalman filtering which already proved to be well suited in conditioning joint feedback velocity.

4.4.4. Quaternion based orientation error

Implementation of quaternion based orientation error was a method designed to improve accuracy of the orientation control. It has been suggested in the literature, (Siciliano & Villani, 1999), that the performance of the axis/angle error is worse than that of quaternion formulation in the case of large orientation errors. Both expressions for the error were tested and compared. Figure 4.11 captures results of the Axis/Angle and Quaternion based orientation errors, in terms of Pitch, Roll and Yaw, with manipulator following trajectory found in Table 4.1. As clearly seen form the experimental data, using identical position and orientation gains $k_p = 500, k_d = 35, k_i = 1000, k_{po} = 200, k_{do} = 70, k_{io} = 0$, performance with quaternion based orientation error was inferior to that of the Axis/Angle.

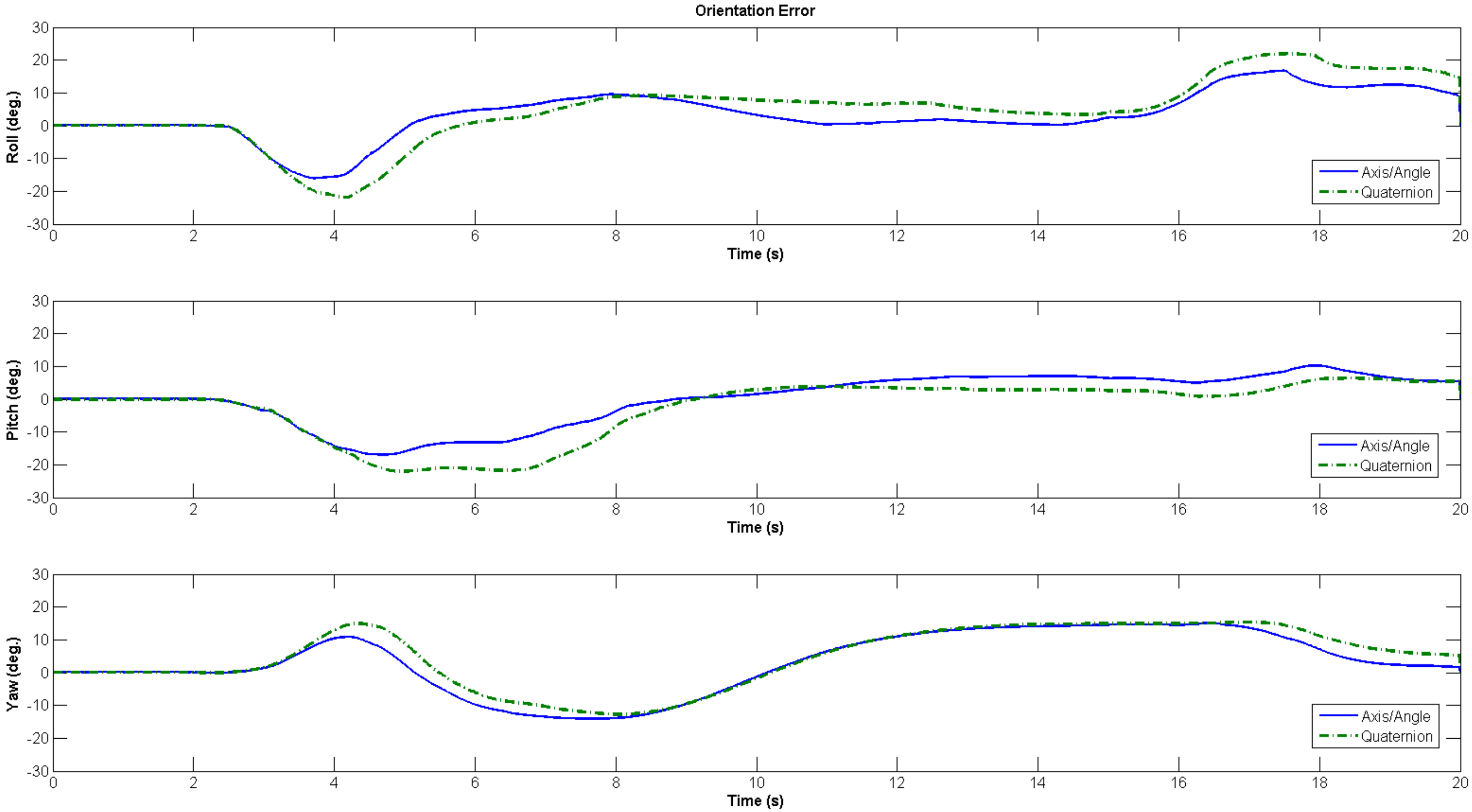


Figure 4.11 : Axis/Angle vs. Quaternion orientation error.

4.4.5. External load compensation

Master manipulator experiences external loads due to reaction forces arising during cooperation. Proper compensation is accomplished by the last term of the equation 2.20. Task loads are transformed directly into necessary joint torques through Jacobian transpose operator. Experimental data were collected for manipulator with, and without external load compensation. Results are shown in Figure 4.12. First column, comprised of three graphs, illustrates external loads acting on the end effector, resulting joint torques, and task position for the case with no external load compensation. Torque delivered to individual joints is that required for gravity compensation. Under relatively light loads, manipulator changes position. Clear correspondence between applied loads and manipulator task position can be seen.

Enabling compensation results in minimal motion of the manipulator. There is minimal shift in position of the end effector along X_0 and Z_0 directions. High magnitude task loads applied along various directions are readily compensated for as seen from the graphs of the second column of Figure 4.12. This time, joint torque increases significantly to oppose the external loads.

However, whenever large loads are suddenly removed, the delay in the response of the compensation results in the manipulator moving in the opposite direction. Residual joint torques that were applied to compensate for the external loads persist for some time after the loads are removed and actuate the joints. Third column of the Figure 4.12 illustrates this effect. System cannot respond sufficiently fast to a high rate of load change.

The above mentioned effect results in deterioration of the position and orientation tracking of the master manipulator. This was first observed during initial testing of the cooperation between master and the slave with both robots coupled using a compression spring. Before obtaining suitable gains, large oscillations were induced in both manipulators. External load compensation of the master was unable to compensate for these rapidly changing loads giving rise to large tracking errors. This was solved later by a proper selection of test gains and filter parameters which improved cooperative performance and eliminated undesired oscillations. In particular, increasing filter r parameter introduced artificial damping that resulted in smoothing out of the load feedback signal. This proved to be the most effective method of dealing with the above problem. Caution had to be exercised however, as not to increase the delay in the system response. This was most critical in the master – slave cooperation where relatively large gains were used.

Similar effect has been observed during slave to slave cooperation while handling a rigid object. Coupling between two manipulators resulted in an over constrained system and gave rise to high frequency oscillations. Without proper signal conditioning, adverse effect on the robot performance were experienced.

4.4.6. Master manipulator sliding mode control

Sliding mode controller has been implemented for the master manipulator. It has been applied in the control of orientation only, retaining classical PID for the position control. Basic functionality has been established and initial testing revealed improvements in the orientation tracking. Position tracking however, suffered. Oscillations can be readily seen in the Figure 4.13 which illustrates performance of original, and sliding mode controller, following trajectory given in the Table A.1. Both controllers used identical positional gains of $k_p = 500, k_d = 35, k_i = 1000,$ and integral error limit of 1. The orientation gains in the original controller were set at $k_{po} = 200, k_{d0} = 70,$ and $k_i = 0$. Sliding mode controller, as per equation 2.48, was utilized with gains of $\Lambda = 100, K_1 = 10, K_2 = 1.$ It should be noted that a classical PID control scheme can easily attain similar orientation accuracy. By properly selecting orientation gains, orientation error can be decreased to a level seen in the results presented below.

Sliding mode control gains have not been optimized. At this point, more work is required in order to validate this control strategy. Moreover, smoothing of the control discontinuity may be required to achieve better performance. This can be done by introducing boundary layer around sliding surface and interpolating control signal whenever inside. In addition, sliding mode controller can be applied in the control of position in addition to orientation. This configuration should offer increased robustness to parametric uncertainties and outside disturbances.

While the preliminary results are optimistic, experimental testing of the robot to robot and human to robot cooperation, presented in the following sections, is based strictly on the classical PID control strategy.

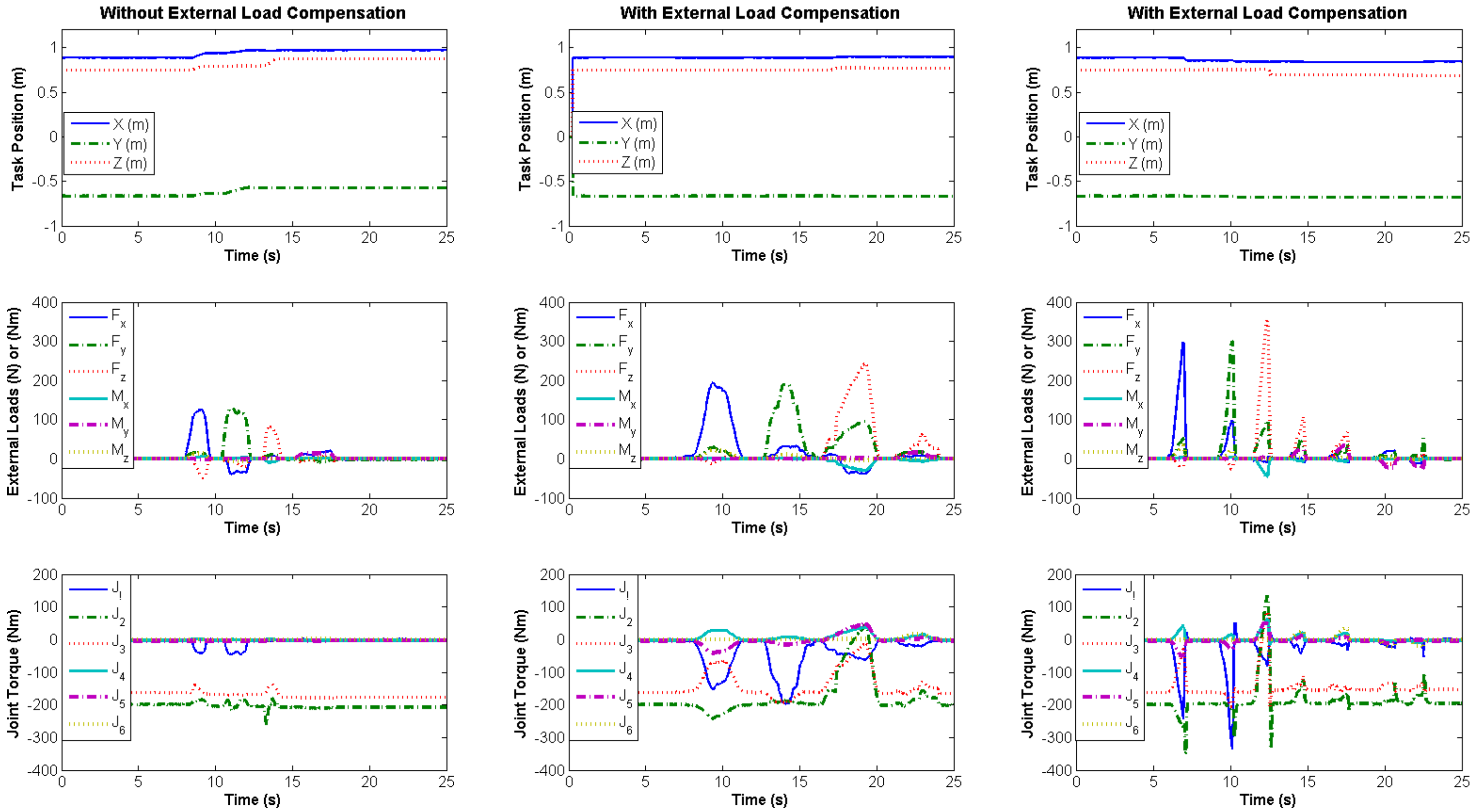


Figure 4.12 : External load compensation.

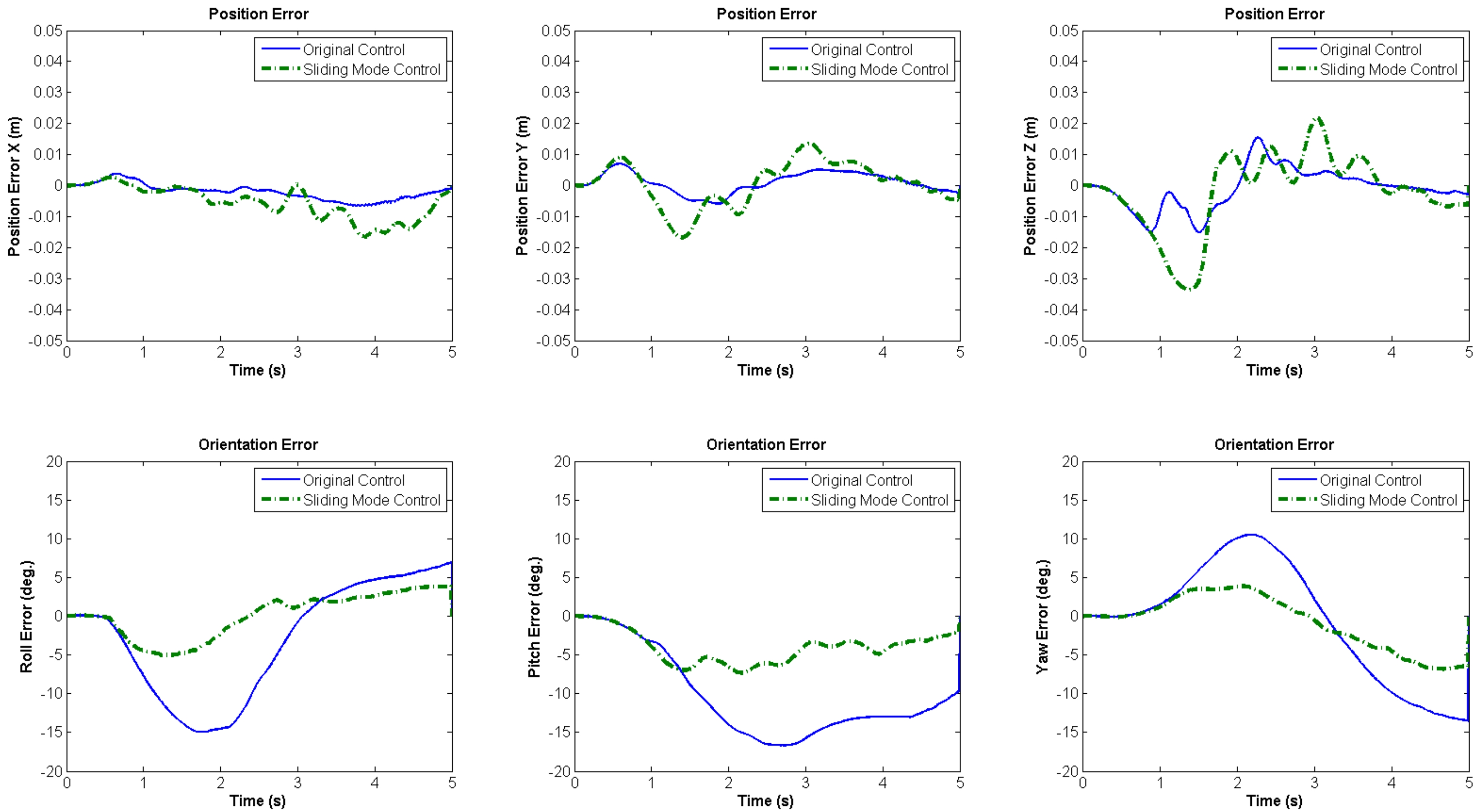


Figure 4.13 : Original vs. Sliding mode control; position and orientation errors.

4.5. Slave Manipulator

Slave manipulator architecture, force and orientation control diagrams are show in Figure 4.14, Figure 4.15, and Figure 4.16 respectively. Number of elements considered previously, such as signal filtering, gripper compensation, and dead zone, have been incorporated.

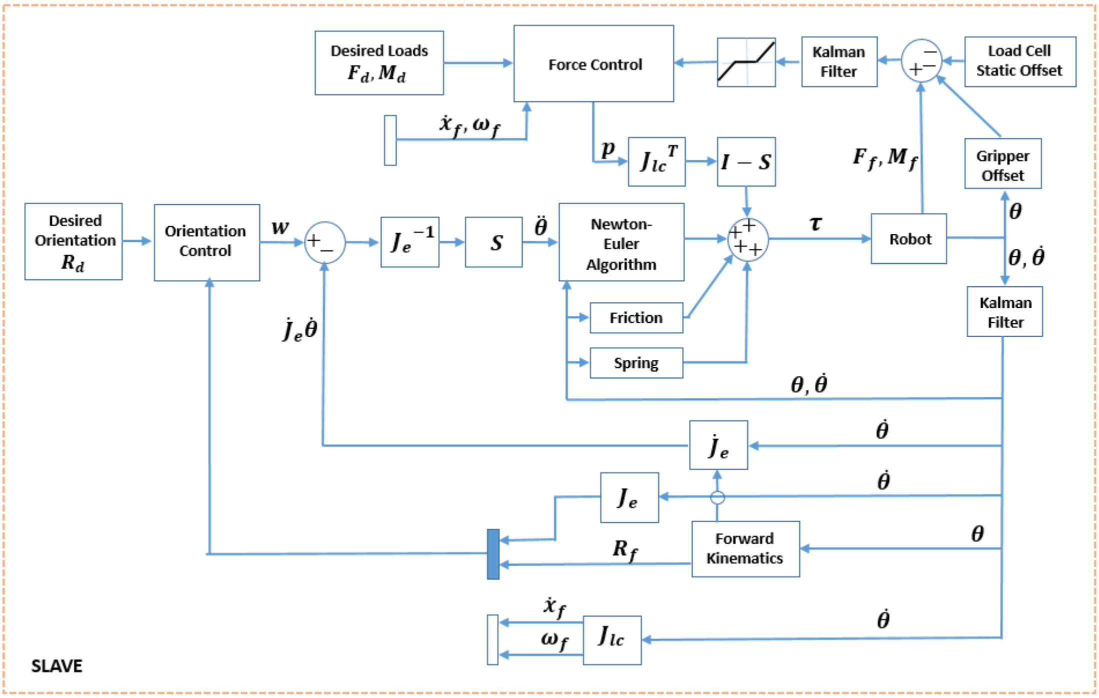


Figure 4.14 : Final Slave manipulator architecture.

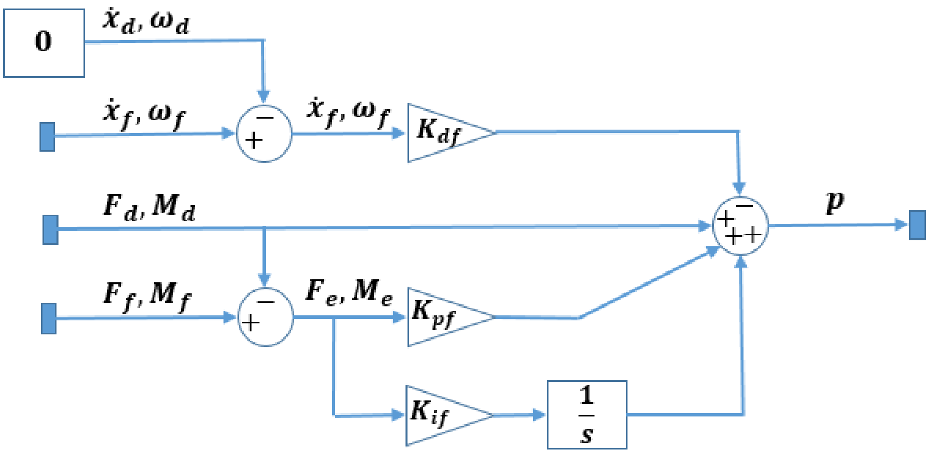


Figure 4.15 : Slave manipulator force control diagram.

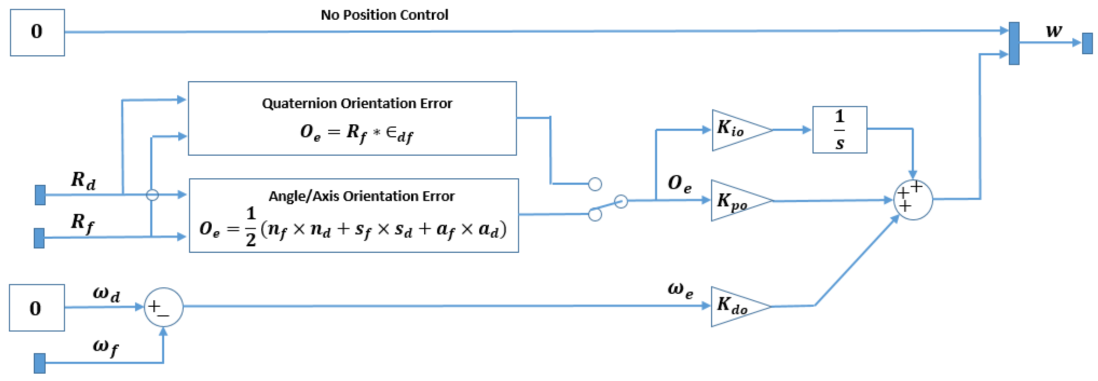


Figure 4.16 : Slave manipulator orientation control diagram.

In addition to control employed during simulation studies, full friction compensation was incorporated into the slave controller. It has been used with the orientation mode of the slave manipulator. Furthermore, choice of directions spanned by the orientation and force controllers was implemented through compliance selection matrices S, and $I - S$ as indicated in the original hybrid controller found in Figure 2.6. This however, has not produced expected results.

The resultant inferior performance of the original formulation might be related to instability coming from the inverse kinematic model, and associated with the mapping of the reduced task subspace, due to application of the compliance matrix S, to joint space. This result has been shown by Fisher and Mutjaba (1992). The correct formulation utilizes general solution of the form:

$$\boldsymbol{dq} = (SJ)^{+}\boldsymbol{dx} + (I - J^{+}J)\boldsymbol{z} \qquad \textbf{(4.3)}$$

where $\boldsymbol{z}$ is an arbitrary vector in joint space and superscript $^{+}$ indicates pseudoinverse. Moreover, it is often sufficient to implement the minimal norm solution, first term of the equation 4.3, which is always stable.

The above formulation has not been verified experimentally. Further work might be necessary to accomplish this task. Instead, In contrast to hybrid controller of Figure 2.6, matrices S and $I - S$ haves been moved after corresponding $J_e{}^{-1}$and J^T operators, effectively moving compliance selection from task to joint space of the manipulator. This modification was intuitively based on the separation of the position and orientation subspaces, as found in the formulation of the inverse kinematics, where first three joints control position and the remaining three joints control orientation.

This design choice resulted in improved overall performance and allowed effective implementation of the orientation control for the slave manipulator.

4.5.1. Gain tuning

Subjective evaluation of force control gains was performed. Force and moment loads were applied to the manipulator while grasping the end effector with a hand. Motion of the manipulator was observed and the force gains adjusted to obtain desired behavior. Smooth response of the manipulator was the evaluation criterion. In has been noted that when operating in the full compliance mode, with orientation control disabled, friction compensation was no longer essential. Force controller, with properly selected gains, was more than adequate to compensate for the lack of friction. Oftentimes, disabling friction proved to be desirable. It made slave manipulator more robust to noise of the force/torque sensor while still delivering optimal performance.

4.5.2. Force tracking

Basic functionality of the slave manipulator included full compliance to external load input. Manipulator position and orientation could be adjusted by exerting forces and moments on the end effector. Correspondence between global forces and task position along global directions X_0, Y_0, and Z_0 is illustrated in figure 4.17. Similarly, relationship between local moments and joint positions of the orientation joints $4-6$, is shown in second column of the same figure. Small force control gains, $k_{pf}=10$ and $k_{df}=20$, and resulting large delay, permitted safe interaction between human operator and the robot.

4.5.3. Force tracking with orientation control

Enabling orientation control allows for manipulation of the slave white retaining prerecorded orientation. Force trancing us used to control first three joints. This allows for the manipulator to change position in response to external force guidance. At the same time, moment loads are compensated through orientation controller acting on the last three joints of the robot. Figure 4.18 presents position and orientation of the slave manipulator with orientation control switched on. Human operator guides the manipulator through direct input at the end effector. Again, force control gains were set at $k_{pf}=10$ and $k_{df}=20$. Orientation control gains were selected as $k_{po}=600$, $k_{do}=50$, and $k_{io}=100$, with integral error limit of 3.

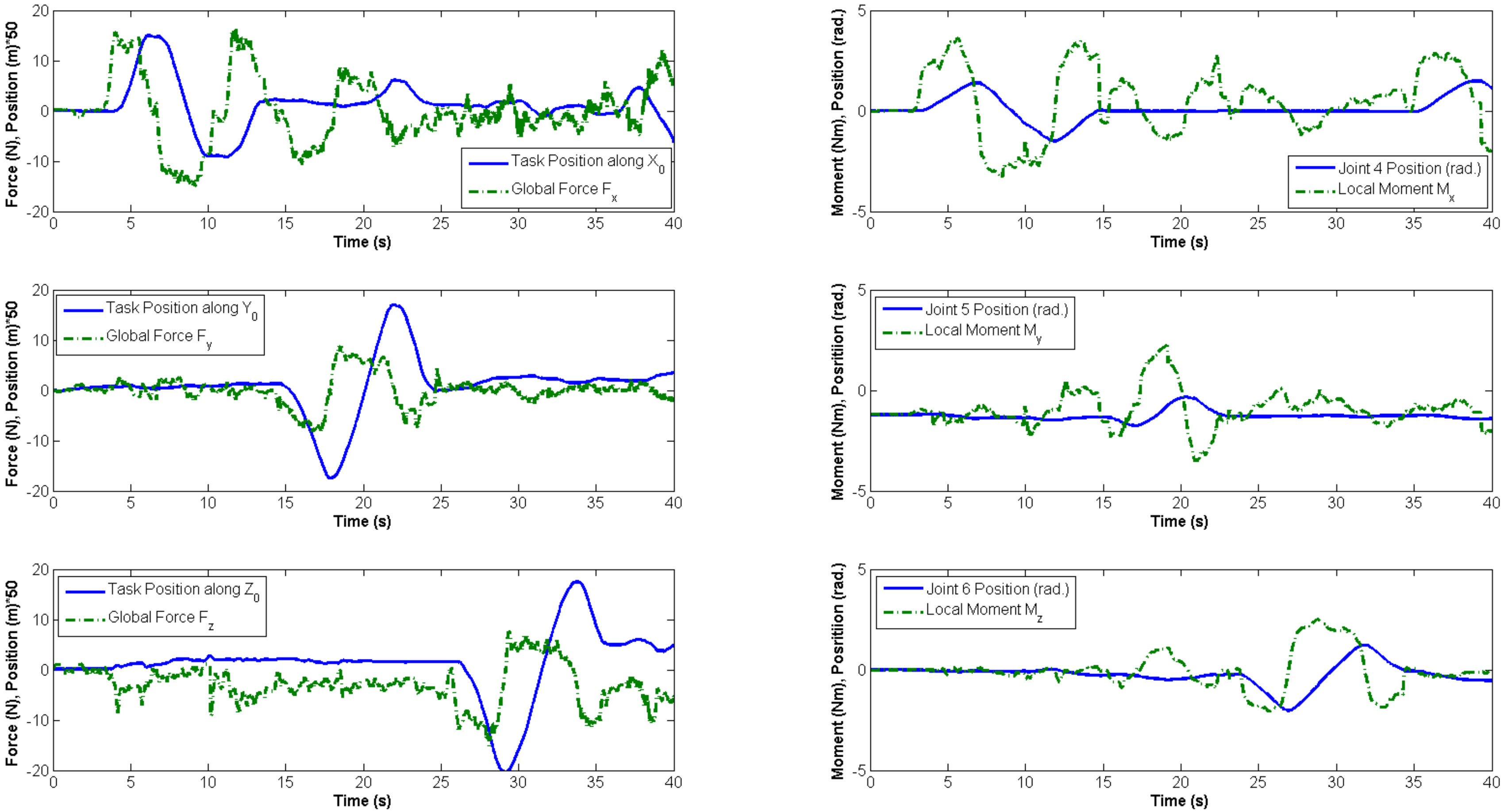


Figure 4.17 : Slave manipulator load tracking.

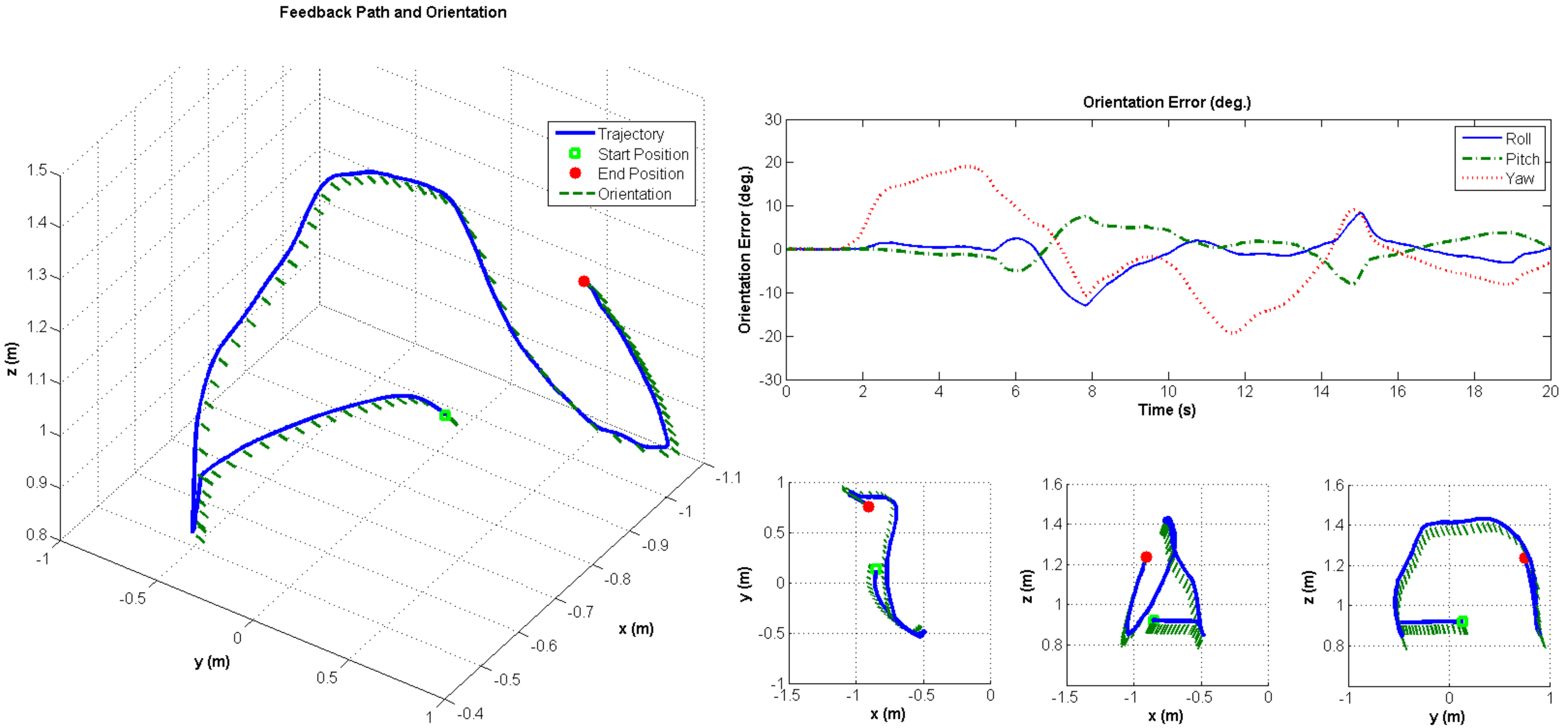


Figure 4.18 : Slave manipulator force tracking with orientation control.

4.5.4. Unknown object compensation with orientation control

A simple task of handling and transporting an unknown object has been scripted into the control. Whenever enabled, and in orientation control mode, slave manipulator would compensate for the weight of the object allowing it to be repositioned with the outside guidance. Experimental results are given in Figures 4.19 and 4.20. In this particular example, slave manipulator is enabled and awaits command to close the jaws. This takes place at about $t = 4.4$ seconds. At this point, load data are used to compensate for the object weight by providing required joint torques as per:

$$\boldsymbol{\Gamma} = J^T \boldsymbol{f} \tag{4.4}$$

All load data to the force controller between $t = 4.4$ and $t = 7.4$ seconds are zeroed. Feedback load data are collected and average offset loads are calculated. Direct joint torque compensation is canceled at $t = 7.4$ seconds into the experiment as seen by a sudden jump in the Z force of about $-19N$. At this moment, offset loads are used to compensated for the weight of the object. Mass of the object is estimated at $2.122\ kg$. Gripper is now firmly holding the object and is ready for transport. By applying outside loads, gripper is repositioned number of times during the test. Orientation of the gripper during each motion segment remains relatively constant. Orientation error never exceeds more that $\pm 20^\circ$ in pitch, roll or yaw. Final position is reached at about $t = 24$ seconds. At $t = 26.9$ seconds into the experiment, gripper jaws are disengaged and the object is removed. This triggers offset load values to be zeroed. Final repositioning takes place between $t = 29$ and $t = 32.5$ seconds. Some end effector motion remains afterwards as the manipulator regains its orientation.

The above experiment has been conducted with force and orientation gains of $K_{pf} = 10I, K_{df} = 20I, K_{po} = 600I, K_{do} = 50I,$ and $K_{io} = 100I$ with integral limit of 3.0.

Data in the above scheme are recorded only at the initial pose and is thus insufficient to completely calculate center of mass of the unknown object. This however is not necessary in the above application. As the orientation remains close to the original one, constant force and moment offsets are sufficient to compensate for the unknown object, and to reposition the item. More robust method where the known mass and its center of gravity is evaluated fully, is presented in the following section.

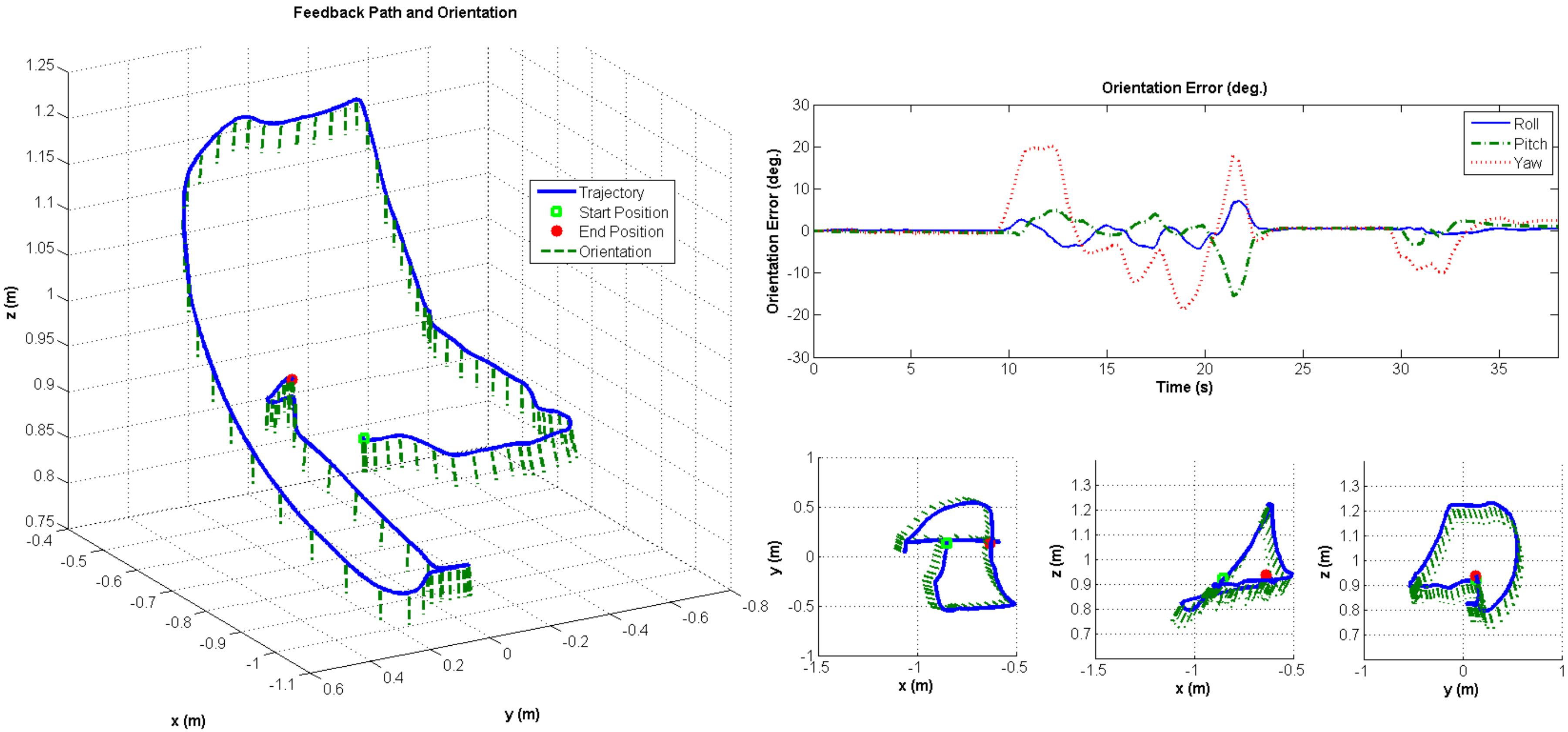


Figure 4.19 : Unknown object compensation; trajectory and orientation.

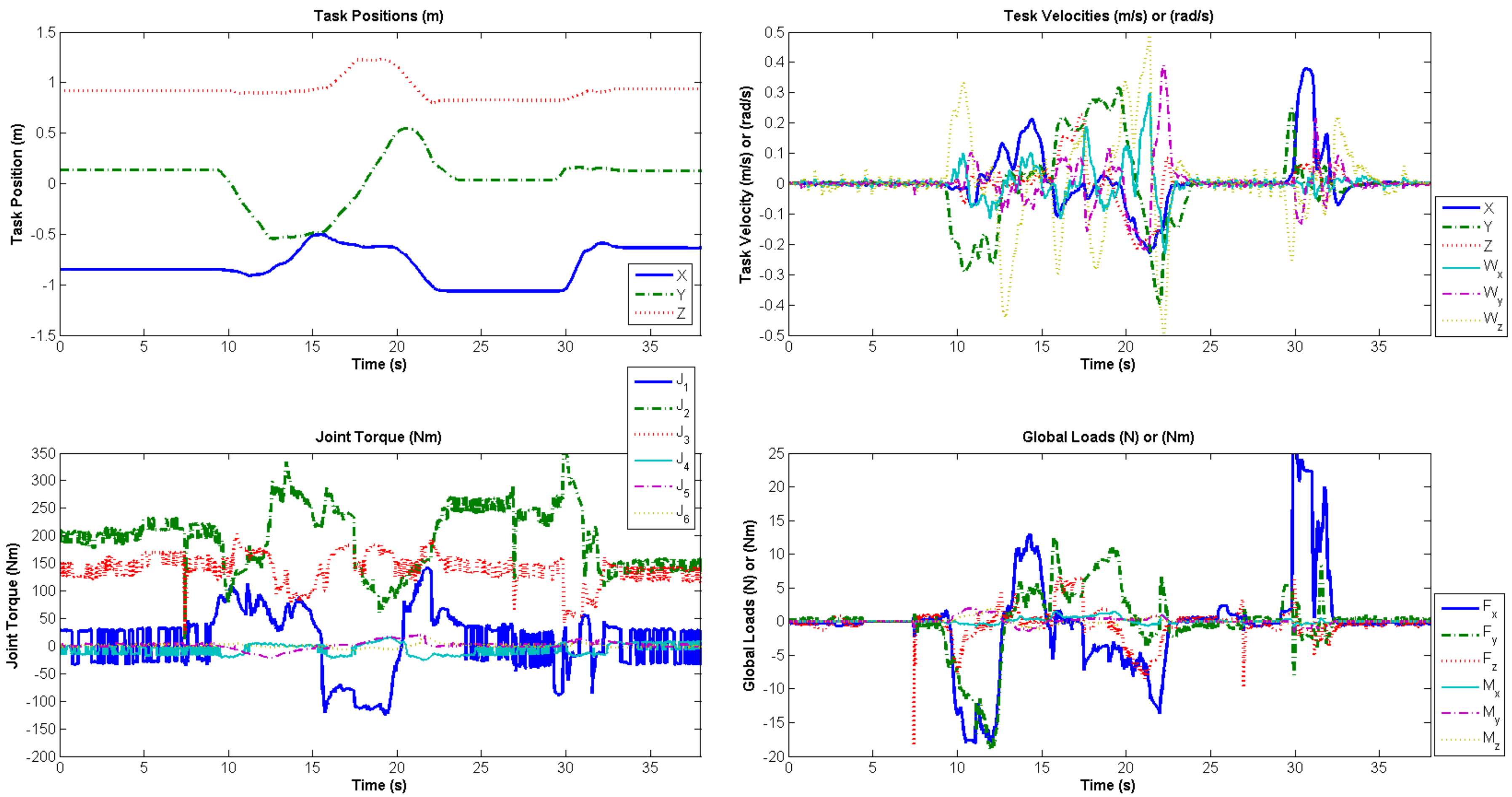


Figure 4.20 : Unknown object compensation; position, velocities, loads and joint torques.

4.5.5. Unknown object compensation in full compliance mode.

Proper calculation of center of mass of an unknown object, held by the end effector, requires two sets of load data collected at different orientations. Vector $\boldsymbol{r_{cg}}$ can be extracted form

$$\boldsymbol{M}_i = \boldsymbol{r_{cg}} \times \boldsymbol{F}_i \tag{4.5}$$

for $i = 1$ and 2. This provides six equations and six solutions, two for each component of the $\boldsymbol{r_{cg}}$ vector**.** Once obtained, physically meaningful solutions have to be selected. Simplest method used in following experiment, is based on the evaluation, and comparison of the denominators. Solutions with the higher value of the denominator is selected. Moreover, mass of the object is obtained using:

$$m = \frac{\sqrt{F_x{}^2 + F_x{}^2 + F_x{}^2}}{9.804} \quad (kg) \tag{4.6}$$

Parameter estimation is illustrated in Figures 4.21 and 4.22. Procedure is initiated once gripper is closed. This takes place at $t = 4.1\ s$. Loads are recorded and used to calculate joint torques required to compensate for the weight of the object. Once complete, force feedback is zeroed. Next, a triangular moment signal is generated to reorient the manipulator. This takes place between $t = 6.3$ and $t = 7.7\ s$ into the experiment. Resulting change in joint torques, velocities, and task position, can be clearly seen. After motion is complete, a second set of load data are recorded. Again, weight of the object is compensated for through direct load to joint torque transformation of equation 4.4. Two sets of load data are then used to evaluate center of gravity (CoG) of the unknown object as per equation 4.5. Acquired parameters are used to update mass and CoG of the gripper. The unknown object becomes part of the manipulator itself. Direct torque compensation is terminated at $t = 10\ s$ indicated by a sharp, instantaneous z load. Afterwards, manipulator is repositioned/reoriented four times. After each motion, manipulator retains orientation and position indicating successful parameter evaluation. Mass and CoG in local frame $R6$ were found to be: $m = 2.002\ kg$, $r_x = 0.0286$, $r_y = -0.0052$, and $r_z = 0.1087\ m$.

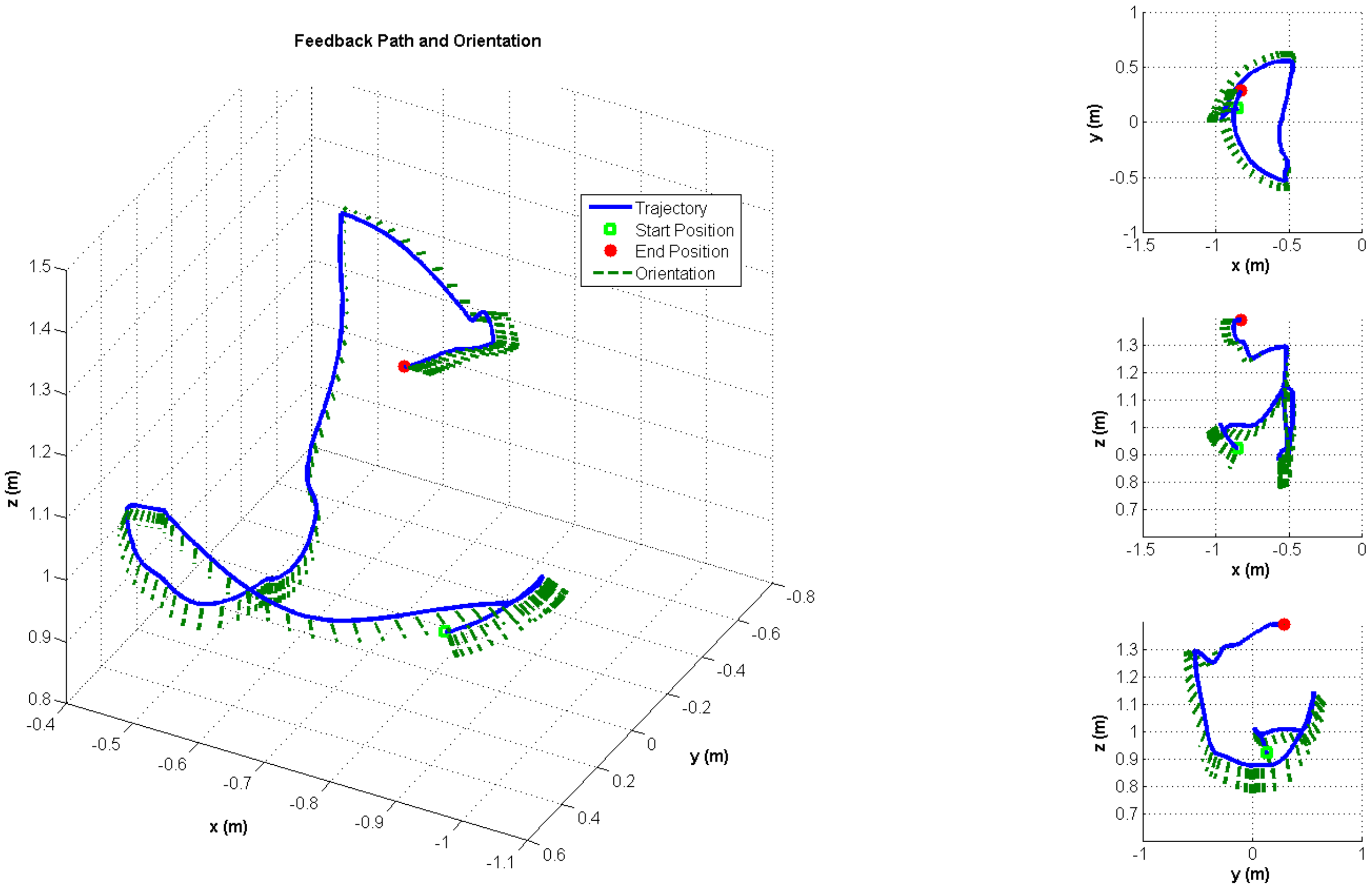


Figure 4.21 : Unknown object mass and center of gravity estimation; trajectory and orientation.

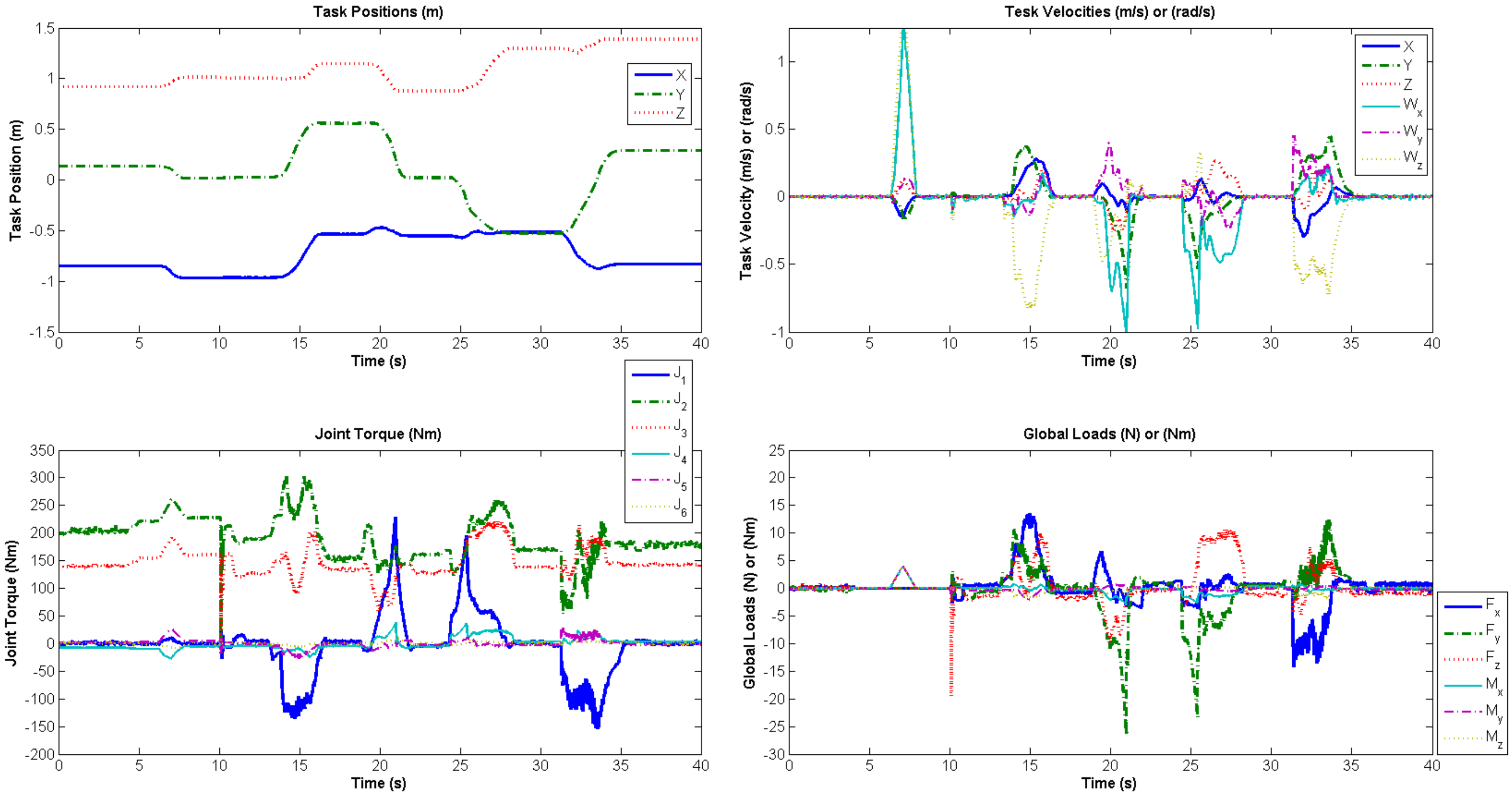


Figure 4.22 : Unknown object mass and center of gravity estimation; position, velocities, loads and joint torques.

4.6. Human-Robot Cooperation.

Initial human to robot cooperation took place during evaluation of the slave manipulator gains. Manipulator was handled and positioned by the human operator. Subsequent experiments were performed where cooperative tasks of lifting and transporting an object were executed.

4.6.1. Lift and transport of an unknown object.

Scripted task of lifting an unknown object was implemented in the control loop. Experimental data for one such task are given in Figures 4.23 and 4.24. First, manipulator is repositioned so that the gripper could properly engage the object $(t\colon 2.4 - 9s)$. Once gripper closes, $(t = 15s)$, force feedback signal to the controller is zeroed. Instead, artificial force control signal in the Z_0 direction is introduced and increased linearly at initially designated rate. Triangular load profile is clearly seen between $t = 18$ and $t = 20.7$ seconds. Joint torque of joint 2 and 3 increases in the similar matter. During all this time, load feedback is used to compensate for the weight of the object through direct joint torque as per equation 4.4.

Position along Z_0 direction is monitored and used as a trigger. When feedback position exceeded a threshold, lift signal is terminated. Some motion however still continues due to inertial effects. Once arm stabilizes at $t = 22.8s,$ load feedback is recorded to be used later. At this point, object is held by the gripper and awaiting input form the user. Another trigger is used to enable repositioned by the operator. A threshold force and moment about local Z_{lc} direction disables direct joint torque compensation and enables force feedback to the force controller. This is indicated by a sharp, instantaneous Z_0 load at $t = 25$ seconds. Load offset in the gripper compensation is now used to compensate for the weight of the object and the manipulator can be now repositioned. This takes place between $t = 25$ and $t = 31.5s$. Again, orientation control permits compensation without fully evaluating unknown object`s CoG.

Human operator does not bear any weight of the object during transport phase but delivers necessary force to repositioned the arm to a desired location. Once in place, gripper jaws are disengaged $(t = 34.4s)$. With the opening of the gripper, load offsets are removed form gripper compensation algorithm. Lift and transport task is complete, and the manipulator arm is removed. Images of the process are shown in Figure 4.25.

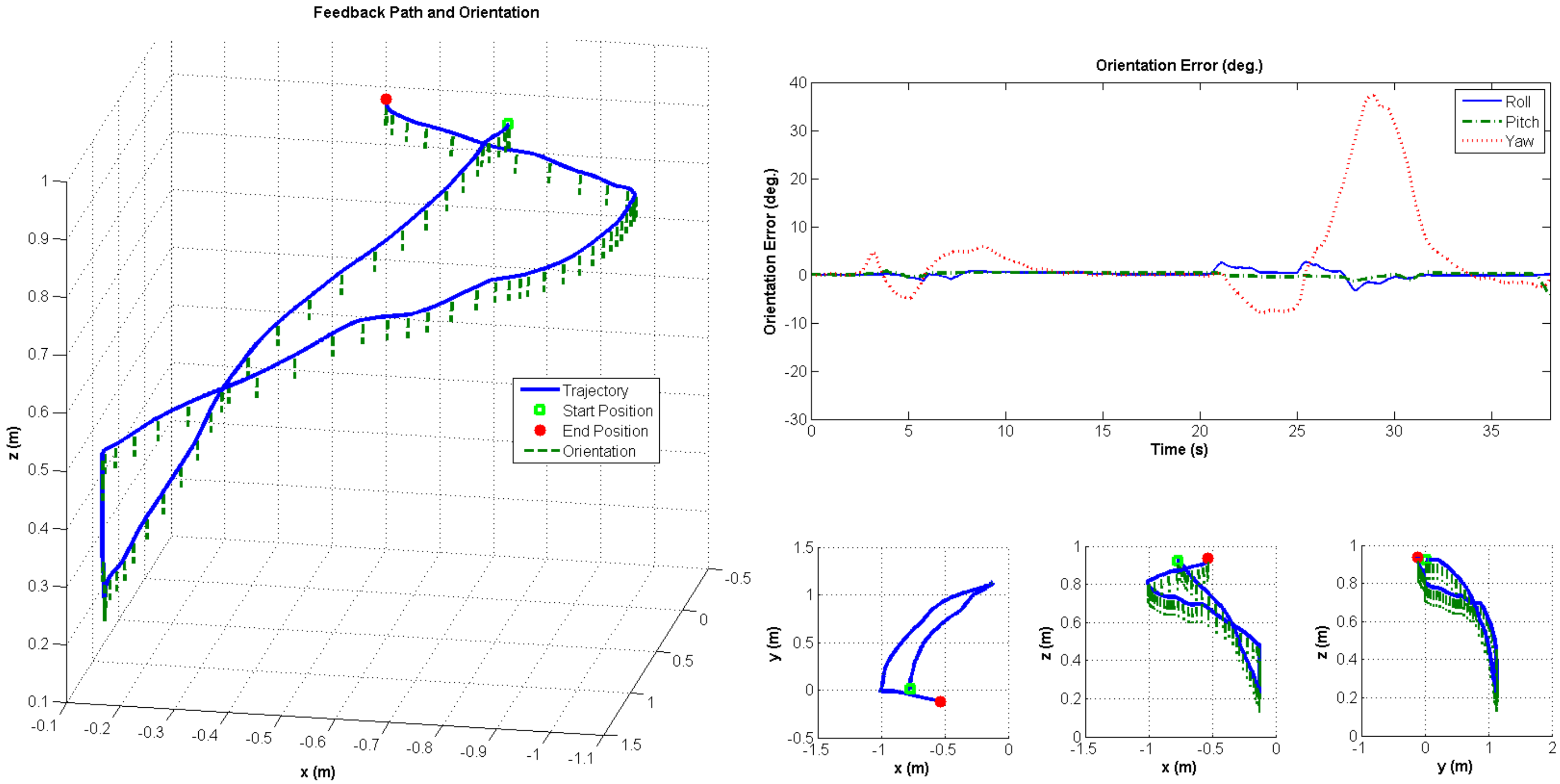


Figure 4.23 : Unknown object lift and transport; trajectory and orientation.

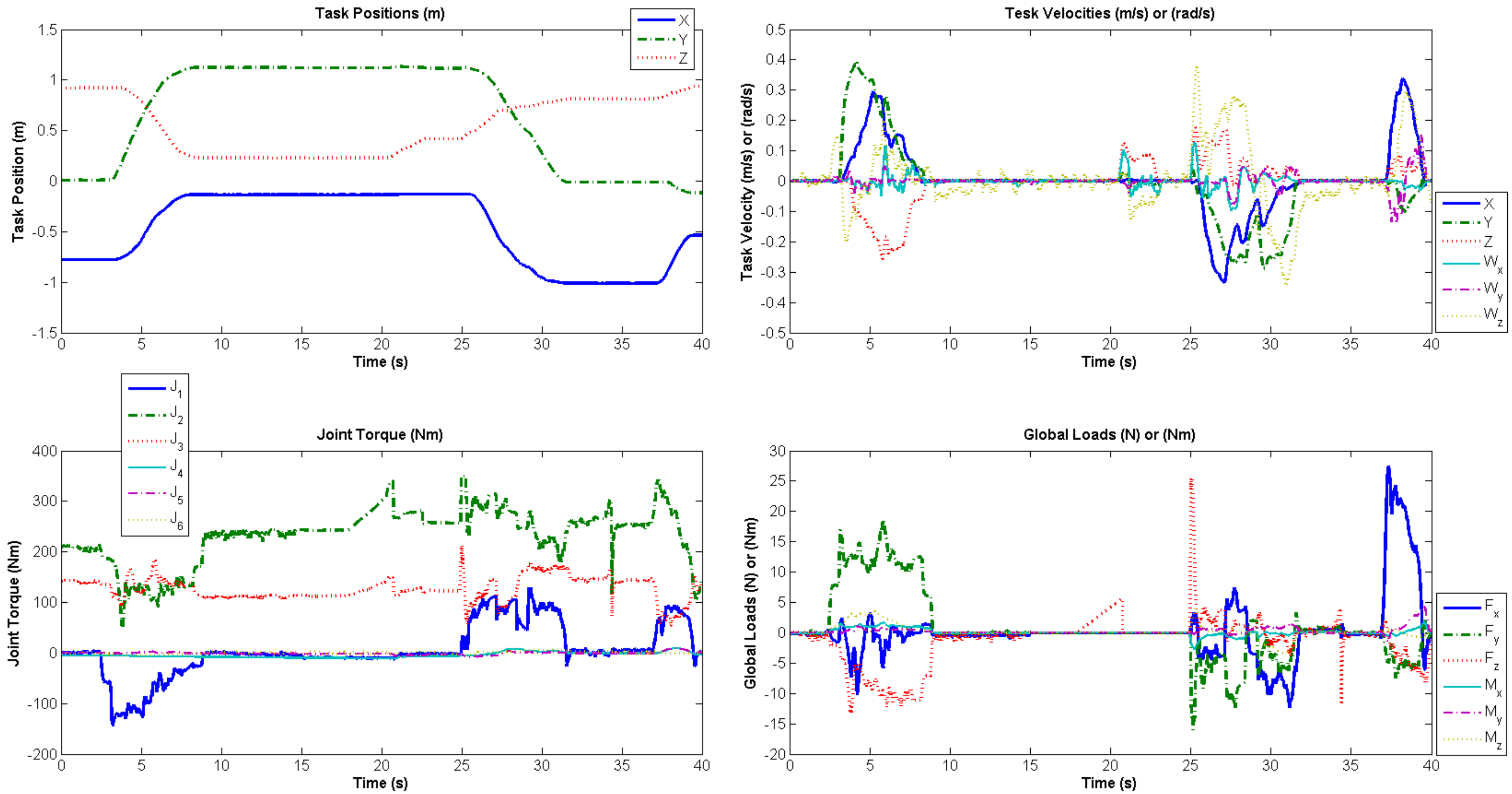


Figure 4.24 : Unknown object lift and transport; position, velocities, loads and joint torques.

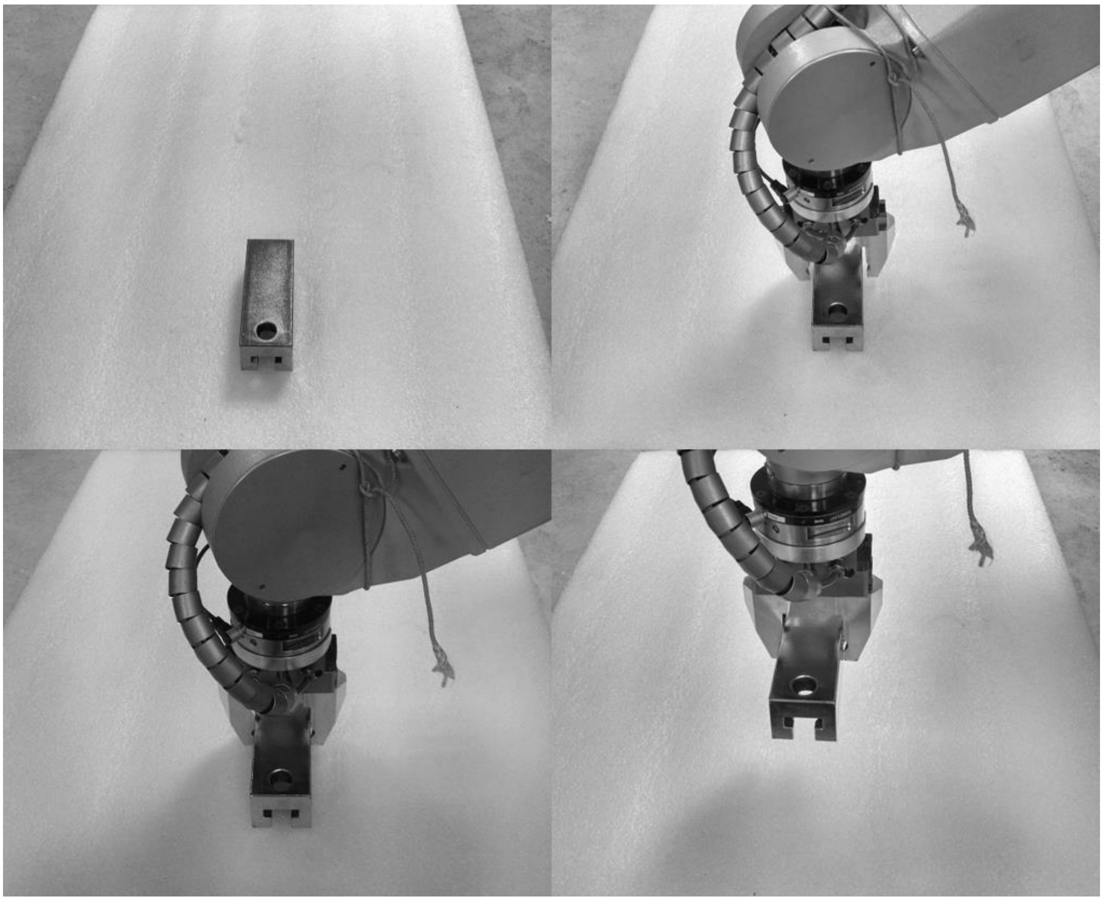


Figure 4.25 : Unknown object lift and transport.

4.6.2. Transport of an unknown object with two slave manipulators.

Two slave manipulators were used to assist in transportation of an unknown object. This time, no lift operation was performed. Instead, an object of an unknown mass was placed between, and clamped onto by the grippers of two slave robots. Once enabled, each unit evaluated loads necessary to hold the object in place. Total mass of the item was found to be $5.2kg$. At this time, human operator grasped the object with two hands and proceeded to reposition the item with the assistance form the robots, see Figure 4.26 and Figure 4.27. Time, $t = 0$ seconds, indicates start of motion. Trajectory and orientation of both slave units is captured in Figure 4.28. Data, including task velocities and global loads, are shown in Figure 4.29.

In terms of gains and parameters, see Table 4.4, it is interesting to note relatively low orientation gains used during this experiment. Furthermore, even though orientation control was engaged, moment feedback was enabled by removing compliance selection matrix $I - S$ in the force control loop of Figure 4.14 . This was done to test effect of moments in terms of repositioning the robotic arm. With human holding the object, orientation could be adjusted manually, and the generated moments helped to reconfigure the arm in such a way as to help in the motion of two coupled manipulators. Again, low orientation gains and the usage of moments allowed for some compliance in this closed kinematic chain. This indeed has worked and was generally more successful than the case of high orientation control gains and classical force feedback with the compliance matrix in place. However, human operator had to endure some moments during the task. While the weight of the object was shared among two manipulators, loads necessary to reposition the system were delivered by the human. These loads were generally higher than those of the previous experiment utilizing a single slave robot.

Initially, rigid coupling of two manipulators resulted in infighting between both arms resulting in dangerous and unstable behavior. This was a result of high frequency oscillations in the load feedback. The signals had to be conditioned to resolve this issue. Parameters of the Kalman filter for force and velocity feedback were adjusted. High value of 5000 was chosen for the r parameter, or noise covariance. This effectively smooth out the feedback signal and attenuated the response of the manipulators. Raw and filtered load data, for manipulator A, are given in Figure 4.30. The induced delay between original and filtered signals can be clearly observed. This effect however was not detrimental in any way because of the interaction with a human whose response is generally much slower.

Table 4.4 : Gains and parameters.

Gain/Parameter		Slave Manipulator
Orientation Control	kpo	200
	kdo	70
	kio	0
	e_limit	
Force Control	kpf	9
	kdf	20
	kfi	0
	e_limit	
Kalman Filter (Velocity)	q	0.125
	r	5000
Kalman Filter (Force)	q	0.125
	r	5000

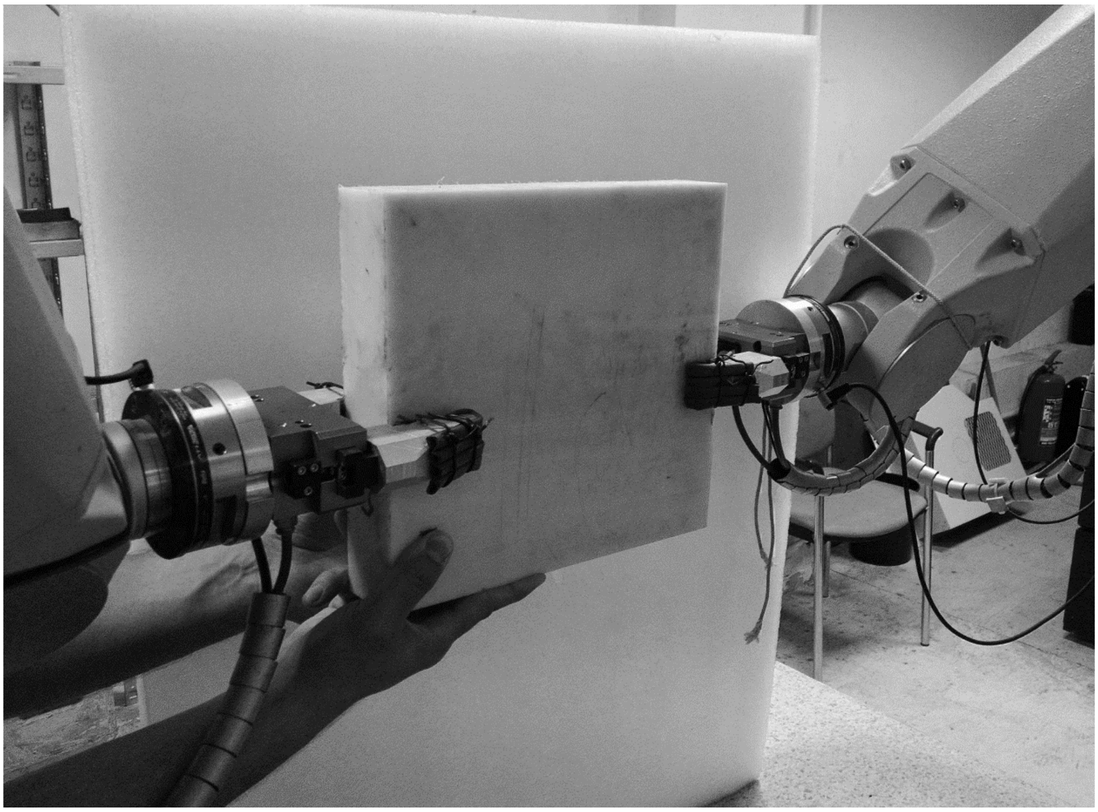


Figure 4.26 : Object being grasped by two slave manipulators

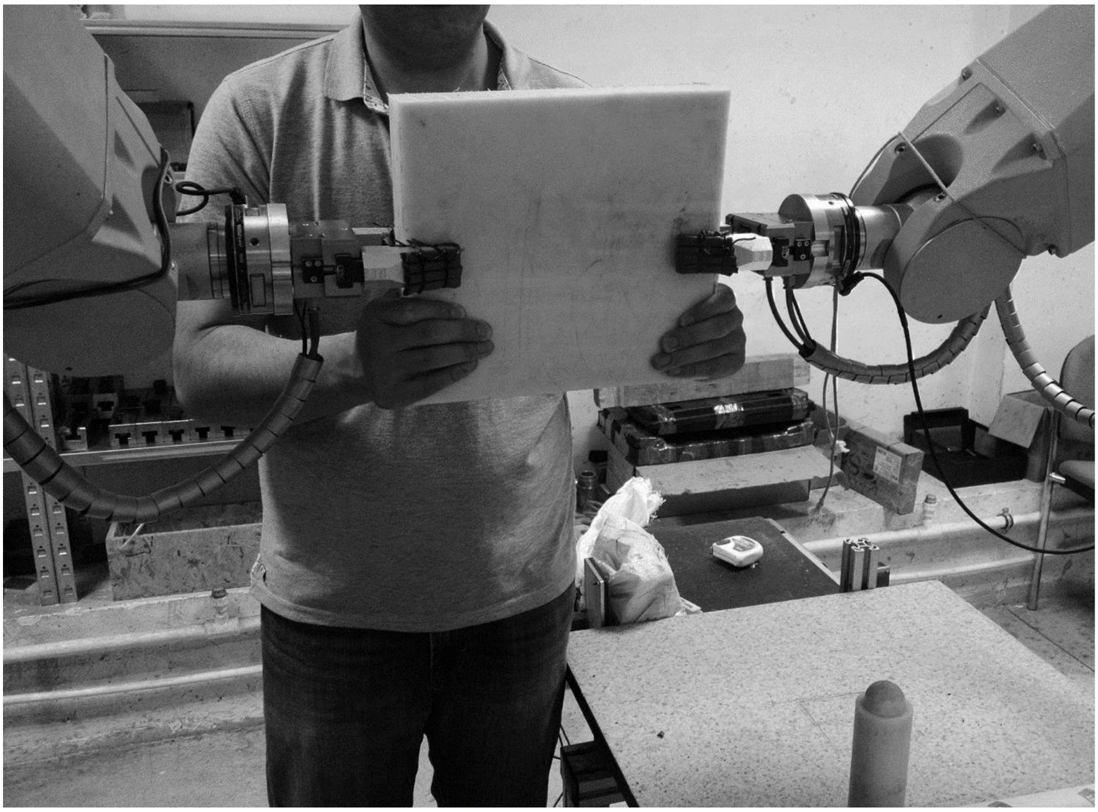


Figure 4.27 : Human-robot cooperation with two slave manipulators.

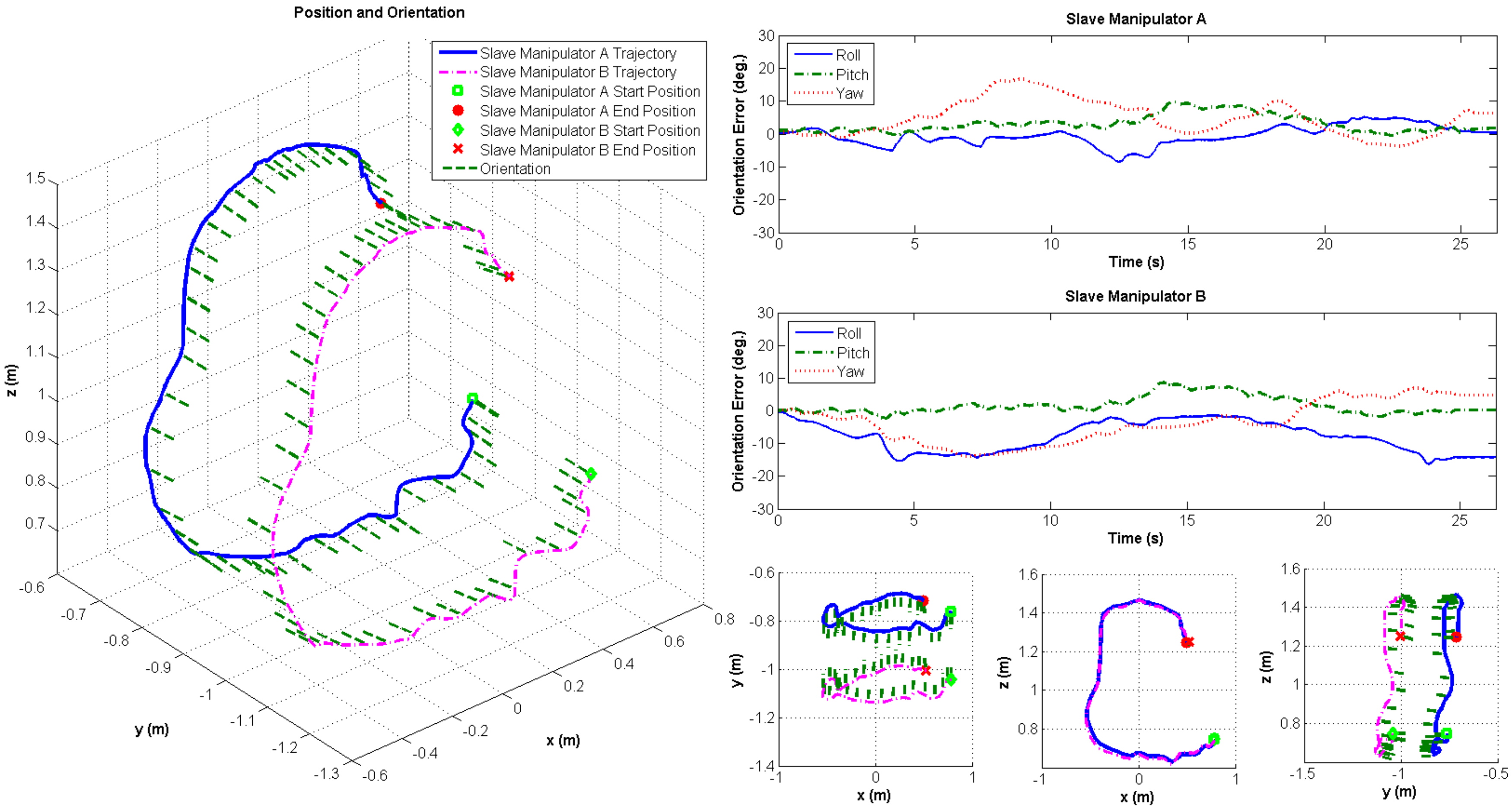


Figure 4.28 : Human-robot cooperation with two slave manipulators; trajectory and orientation.

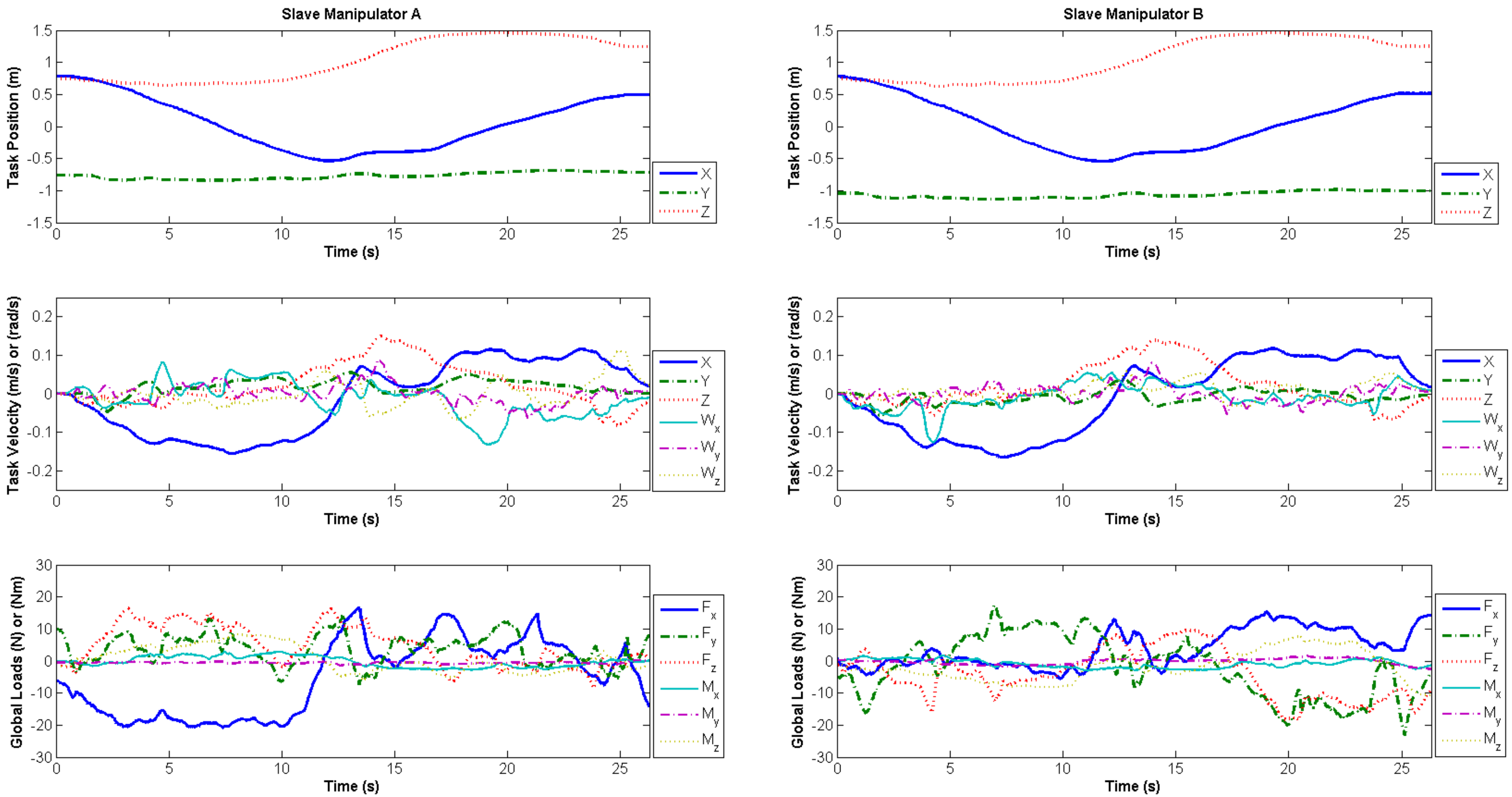


Figure 4.29 : Human-robot cooperation with two slave manipulators; task position, and velocity, global loads.

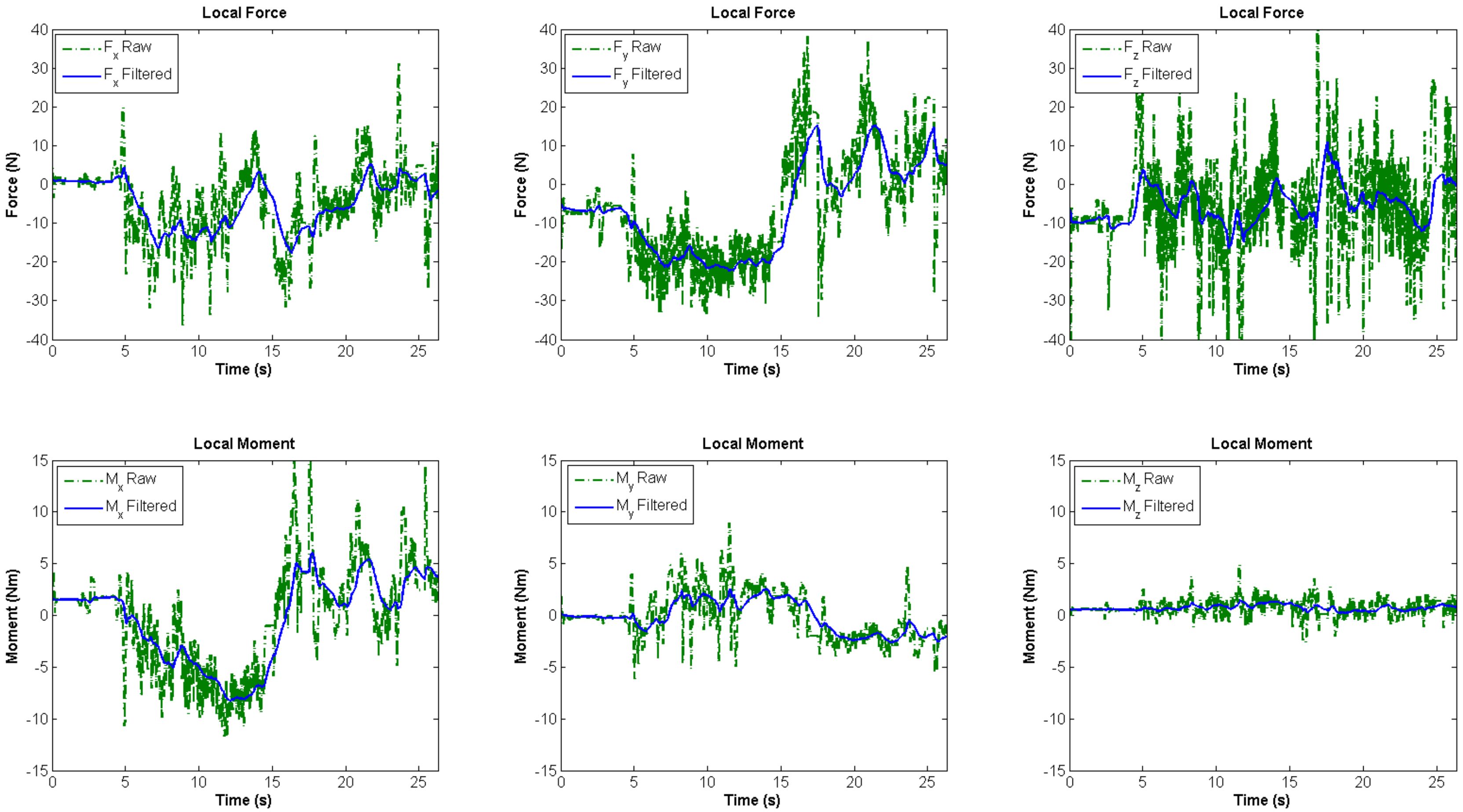


Figure 4.30 : Human-robot cooperation with two slave manipulators; Manipulator A local loads, raw and filtered.

4.7. Master-Slave Manipulator Coupling and Cooperation

Cooperation between master and slave manipulators have been experimentally tested by coupling both robots using a compression spring shown in Figure 4.31. A simple, two segment trajectory, with constant orientation, and facing the other robot, was given to the master unit. Slave manipulator was tasked with following the master based on force feedback. Orientation control and unknown object estimation function, described previously, have been enabled for the slave. Load compensation has been switch on for the master robot. Parameters and gains used during the experiment are given in Table 4.5. Orientation gains, including integral term, have been increased for both manipulators. Whereas force gains for human to robot cooperation were in the range of $5-10$ for k_{pf} and $15-30$ for k_{df}, robot to trobot cooperation necessitated much higher values. Higher k_{pf} was used to increased sensitivity of the slave. Much higher value of k_{df} was found to effectively damp oscillations arising during cooperation with master manipulator.

Table 4.5 : Cooperation gains and parameters.

Parameter/Gain		Master Manipulator	Slave Manipulator
Position Control	kp	500	
	kd	35	
	ki	0	
	e_limit		
Orientation Control	kpo	600	600
	kdo	100	50
	kio	100	100
	e_limit	2	3
Force Control	kpf		35
	kdf		1250
	kif		0
	e_limit		
Kalman Filter (Velocity)	q	0.125	0.125
	r	20	20
Kalman Filter (Force)	q	0.125	0.125
	r	20	500

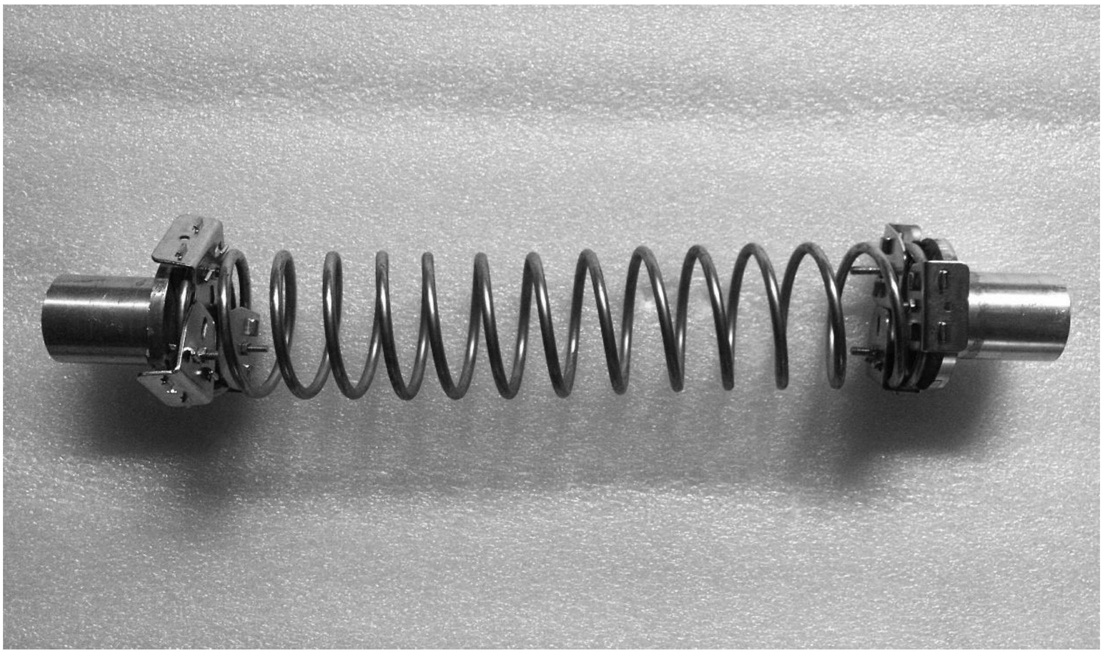


Figure 4.31 : Coupling spring.

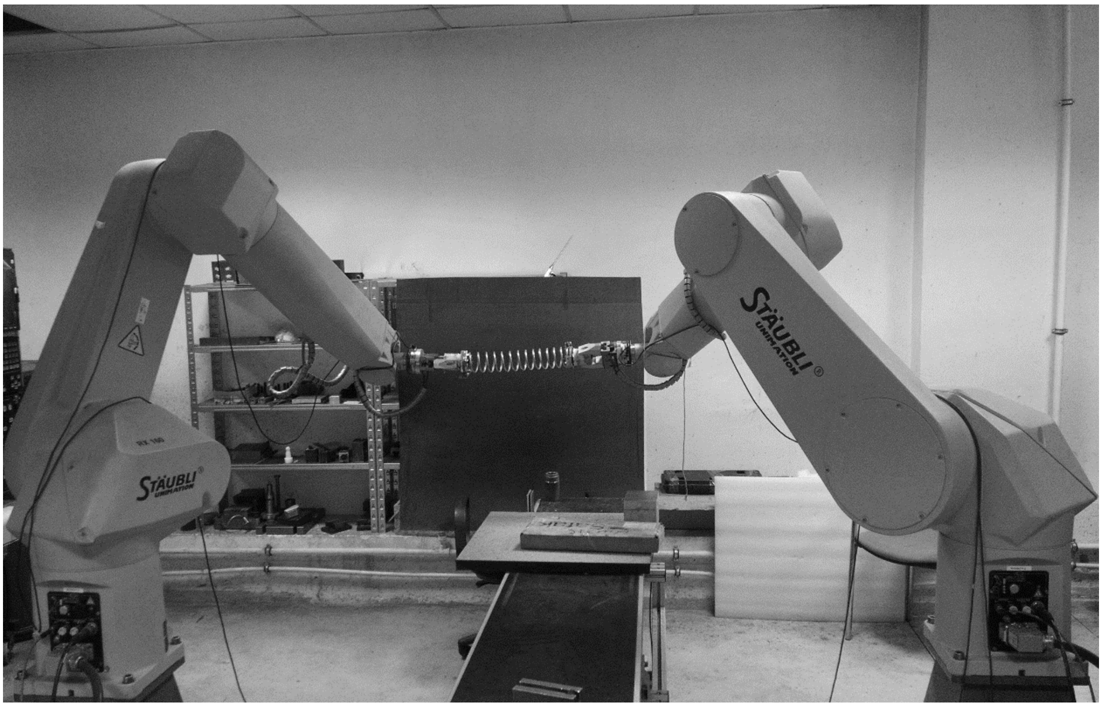


Figure 4.32 : Master-Slave cooperation.

Reference, and feedback trajectories, as well as the orientation, and orientation error, for the master and slave manipulators, are shown in Figure 4.33. Orientation errors for both units are bellow $10°$ in pitch, roll, and yaw. Conversely, position tracking of the master has been significantly affected. It deviates as much at $0.05\ m$ from the desired path. This result is a direct consequence of the higher orientation gains and external disturbances due to cooperation experienced by the master robot. Task position, along $X_{0,}$ $Y_{0,}$ and Z_0 directions, are clearly shown in Figure 4.34. For the slave manipulator, shape of the position curve along Z_0 is related to shape of the F_z force experienced by the unit. The relationship can be seen as that of a function, force, and its integral, position.

It has been found that simply increasing force gains was not sufficient to stabilize the system. Kalman force parameters had to be adjusted to further damp the response and to eliminate undesired oscillations. This unfortunately increased the delay to approximately $0.15 - 0.2$ seconds. Local, raw and filtered, load signals for the slave robot are shown in Figure 4.35. Furthermore, the discrepancy in the filter parameters for velocity and force signals made it difficult to extrapolate any conclusions about the response. Whereas k_{pf} acts on the force error dependent on the force Kalman filter, k_{df} acts on task velocity, as per equation 2.46, which is governed by the velocity Kalman signal. During the experiment filter parameters were different for velocity and force signals. As such, normal relationship between k_{pf} and k_{df} through oscillation period T_u may not be invoked. It is desired to conduct the experiment using same parameters values or to switch from velocity to force derivative in the control of equation 2.46. More testing is required.

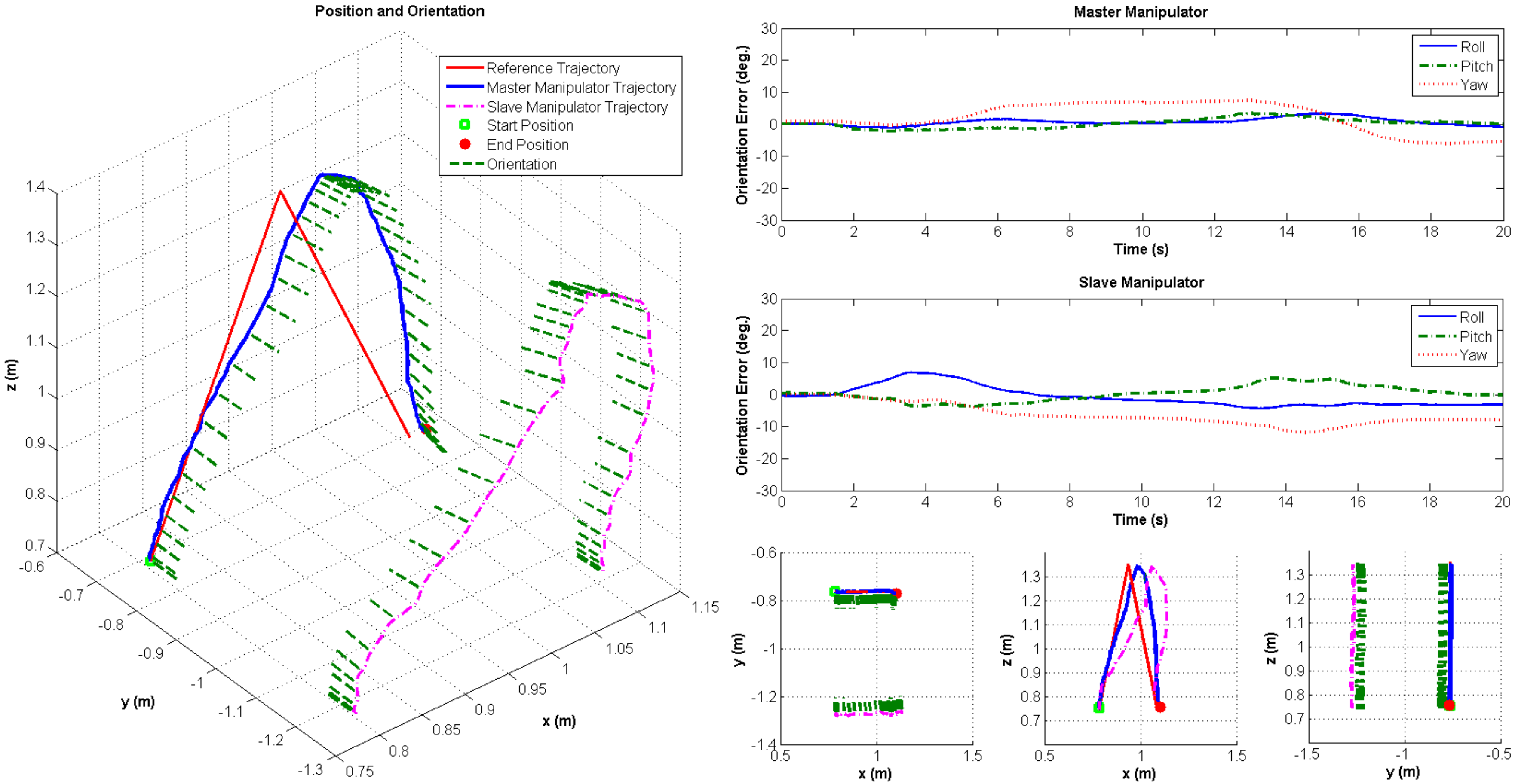


Figure 4.33 : Cooperation; trajectory and orientation.

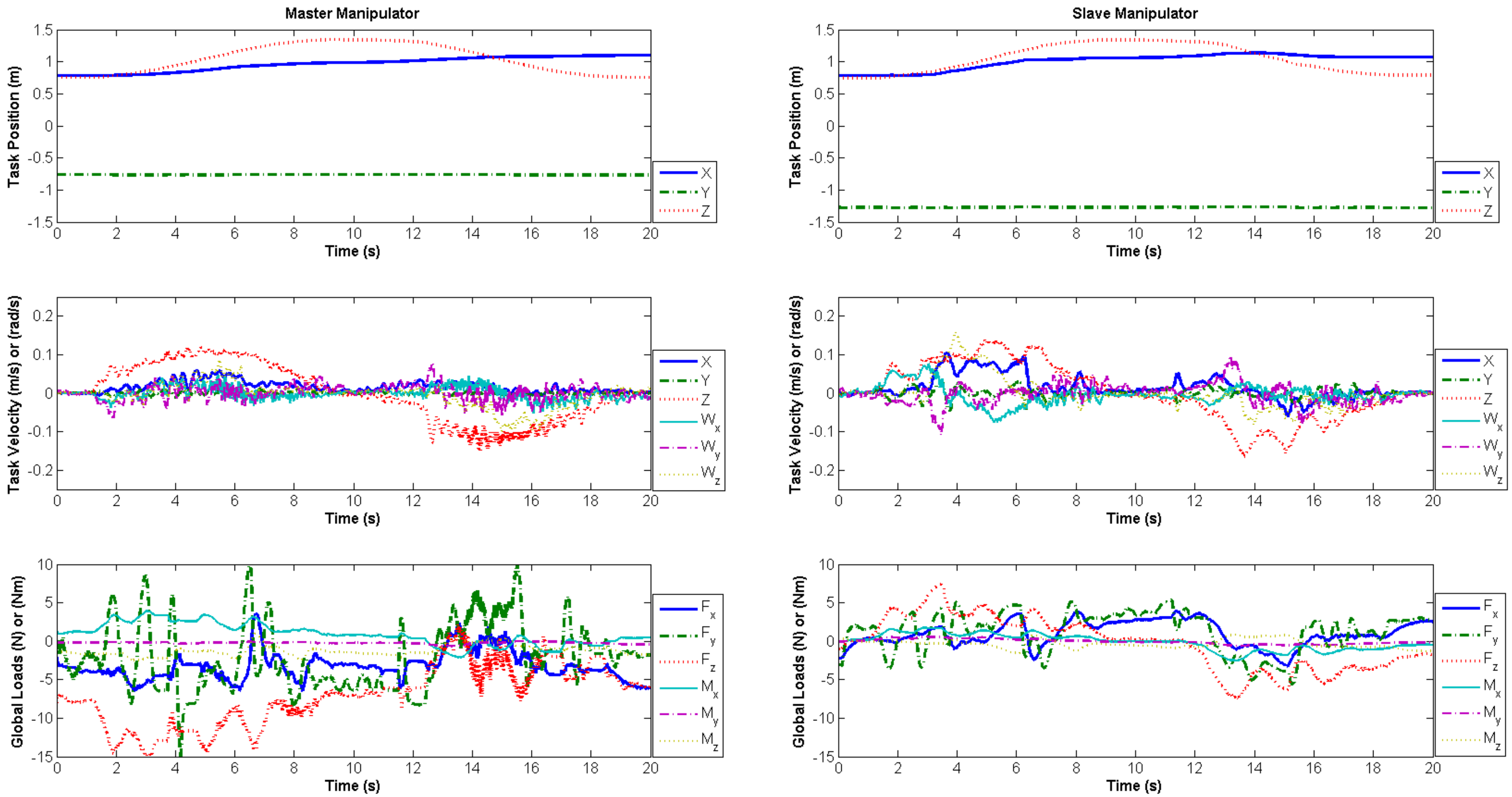


Figure 4.34 : Cooperation; task position, and velocity, global loads.

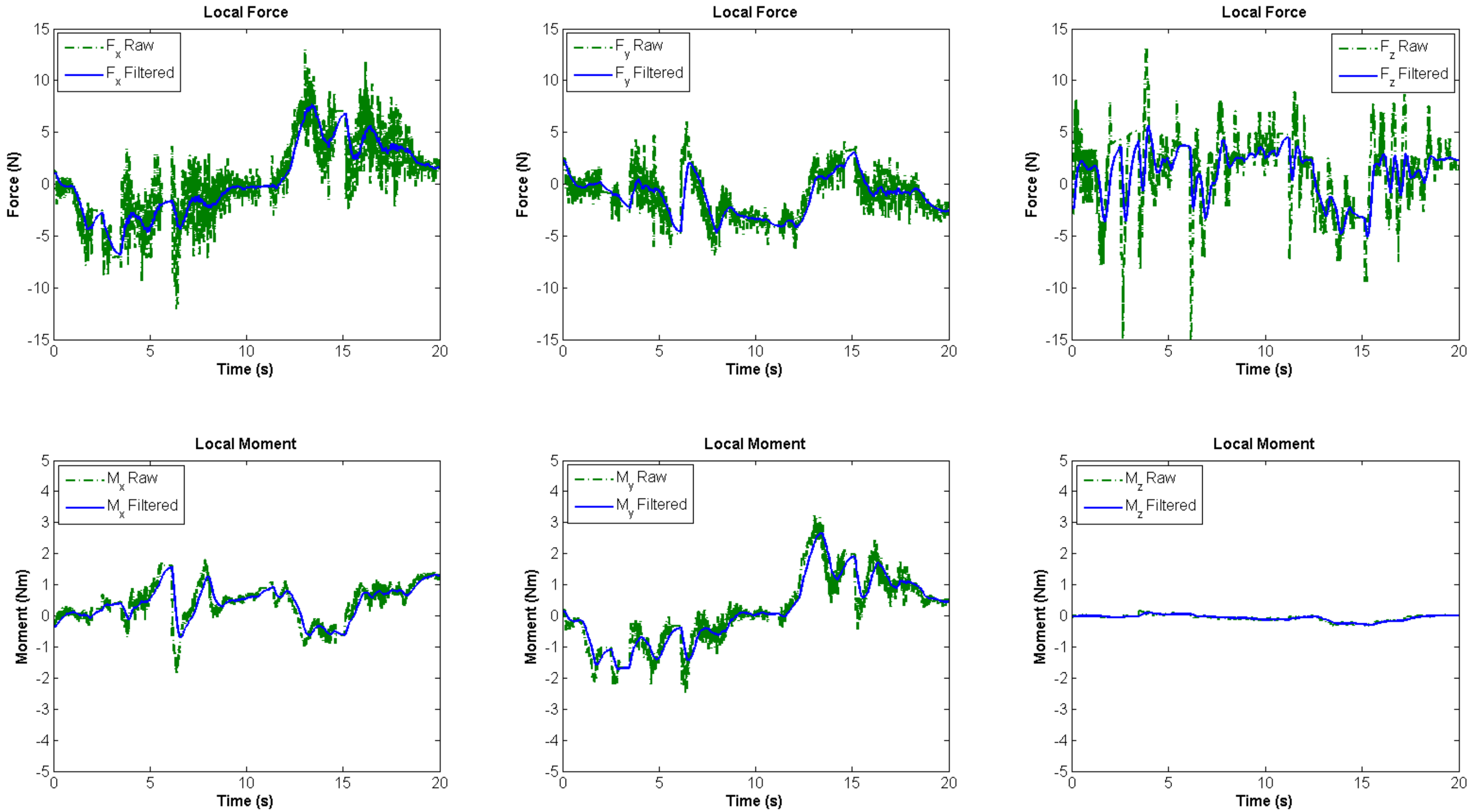


Figure 4.35 : Cooperation; local loads, raw and filtered.

5. CONCLUSIONS AND RECOMMENDATIONS

5.1. Practical Application of This Study

In addition to direct force control, inverse dynamic control has been used quite extensively in the academia as well as in the industry. Experimental work validated theory and simulations. Furthermore, control architecture has been implement and executed using industrial grade robot manipulators. Completed experiments accomplished tasks commonly required in various field operations. Therefore, research presented above has direct practical application.

5.2. Conclusions

The aim of this work was to investigate, learn, and to apply control engineering methods to a practical problem. This has been realized with the application of computed torque and force controls in robotics. All elements of the subject were investigated and successfully applied in the area of robotic cooperation, topic with high practical and research value. Effective position, orientation and force controls was implemented. Successful cooperation between two manipulators was achieved. It has been further shown that the study has a direct application in the area of human-robot cooperation.

5.3. Recommendations

Improving performance of the Master-Slave robot cooperation, and extending its applicability is an open topic. Among many, good characterization of the friction model, and friction parameter identification, would be one of the top objectives. This would benefit accuracy of the orientation control. Another interesting extension would involve addition of gain tuning algorithms or supervisory control. Moreover, other control structures could be investigated. Most promising methods include impedance, adaptive, fuzzy and passivity base controls. Extending reliability and robustness would be welcomed as well. While sliding mode control has been implemented, it has not been sufficiently tested to provide adequate evaluation. Additional work is required. Lastly, method dealing with singular configurations would further extend applicability of the presented work.

REFERENCES

Akbaş, S. (2015). *Force Control of Staubli Rx160 Manipulator.* İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

ATI Industrial Automation. (n.d.). F/T Sensor Specifications: Delta. Retrieved 4 18, 2016, from http://www.ati-ia.com/products/ft/ft_models.aspx?id=Delta

Cheng, W., Wang, X., & Ma, H. (2015). Master-Slave Force Control Based on Grey GM(1,1) Model of Robot Gripper. *2015 International Conference on Advanced Mechatronic Systems (ICAMechS)* (pp. 380 - 383). Beijing: IEEE.

Cui, J., Tosunoglu, S., Roberts, R., Moore, C., & Repperger, D. W. (2003). A Review of Teleoperation System Control. *Proceedings of the Florida Conference on Recent Advances in Robotics.* Boca Raton.

Fisher, W., & Mujtaba, M. (1992). Hybrid Position/Force Control: A Correct Formulation. *The International Journal of Robotics Research, 11*(4), 299-311.

Gueaieb, W., Karray, F., & Al-Sharhan, S. (2004). A robust adaptive fuzzy positionforce control scheme for cooperative manipulators. *IEEE Transactions on Control Systems Technology, 11*(4), pp. 516 - 528.

Hogan, N. (1985). Impedance Control: An Approach to Manipulation: Part 1-Theory. *Journal of Dynamic Systems, Measurement and Control, 107*, 1-24.

Lee, S. J., & Jung, S. (2011). Experimental Studies of an Object Handling Task by Force Control between Two Balancing Robots. *Control, Automation and Systems (ICCAS)* (pp. 197 - 201). Gyeonggi-do: IEEE.

Liu, C., Zhao, D., & Xiang, X. (2014). Force Synchronization of Multiple Robot Manipulators: A first study. *Control Conference (CCC)* (pp. 2212 - 2217). Nanjing: IEEE.

Luh, J., & Zheng, Y. (1987). Constrained Relations Between Two Coordinated Industrial Robots for Motion Control. *Int. J. Robot, 6*, 6-70.

Matlab Simulink. (R2014a). [Computer Software] MathWorks.

Nakano, E., Ozaki, S., Ishida, T., & Kata, I. (1974). Cooperational Control of the Anthropomorphous Manipulator 'MELARM'. *Proc. 4th Int. Symp. Ind. Robots*, (pp. 251-260). Tokyo.

Raibert, M. H., & Craig, J. J. (1981). Hybrid Position/Force Control of Manipulators. *ASME Journal of Dynamics Systems, Measurement, and Control, 102*, 126-133.

Schneider, S. A., & Cannon Jr., R. H. (1990). Object Impedance Control for Cooperative Manipulators: Theory and Experimental Results. *IEEE Trans. Robot. Autom.*, 383-394.

Siciliano, B., & Villani, L. (1999). *Robot Force Control.* New York: Kluwer Academic Publishers.

Stäubli. (2012, 06 14). Low Level Robot Control Interface. (LLI S7.4). Faverges, France.

Stäubli. (n.d.). Product Leaflet. Retrieved 4 8, 2016, from http://www.staubli.com/index.php?eID=tx_nawsecuredl&u=0&g=0&t=1461092878&hash=f6cf63e806177a708b2502456ae66a9b168162e0&file=fileadm

in/user_upload/staublicom/robotics/PDF-files/RX160_Robot_EN_092013.pdf

Stückler, J., & Behnke, S. (2011). Following Human Guidance to Cooperatively Carry a Large Object. *IEEE-RAS International Conference on Humanoid Robots* (pp. 218 - 223). Bled: IEEE.

Sun, D., & Mills, J. (2002, August). Adaptive Synchronized Control for Coordination of Multirobot Assembly Tasks. *IEEE Transactions on Robotics and Automation, 18*(4), pp. 498 - 510.

Tarn, T. J., Bejczy, A. K., & Yun, X. (1988). New Nonlinear Control Algorithms for multiple Robot Arms. *IEEE Trans. Aerosp. Electron. Syst*, 24-25.

Waiboer, R. (2007). *Dynamic Modeling, Identification and Simulation of Industrial Robots for Off-Line Programming of Robotised Laser Welding.* Veldhoven.

Whitsell, B., & Artemiadis, P. (2015). On the role duality and switching in human-robot cooperation: An adaptive approach. *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3770 - 3775). Seattle: IEEE.

Zengin, E. (2013). *Staubli RX160 Manipulatörün Modellenmesi, Taninmasi ve Kontrolü.* İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

APPENDICES

APPENDIX A: Master Manipulator Gain Tuning
APPENDIX B: Spatial Operator Algebra derivation of term $\dot{J}\dot{\boldsymbol{q}}$
APPENDIX C: Lyapunov stability for orientation error dynamics

APPENDIX A

Table A.1 : Task trajectory for gain tuning expressed in terms of joint angles.

Point	Joint Position (°)						Time (s)
	Joint _1	Joint _2	Joint _3	Joint _4	Joint _5	Joint _6	
1	0	-20	90	0	20	0	0
2	45	45	45	45	45	45	5

Table A.2 : Master manipulator position gain tuning.

kp	kd	Task Position Total Error (m)	Task Position End Error (m)	Orientation Total Error	Orientation End Error	Notes
400	25	17.3277	0.0044	920.6034	0.996	
450	25	14.5327	0.0036	863.1952	1.0028	
500	25	13.1527	0.0035	862.2198	1.003	
600	25	10.9733	0.0023	862.041	1.003	Chattering
800	25	9.6478	0.0027	851.855	1.0025	Oscillations
400	35	15.7391	0.0045	854.4157	1.0022	
400	45	15.3695	0.0045	832.9657	1.001	
400	55	15.1882	0.0045	823.0712	0.9994	Oscillations
400	70	14.8392	0.0042	835.9098	0.9995	Chattering
450	35	14.0599	0.0039	860.5577	1.0028	
450	45	13.6848	0.004	845.6057	1.0022	Oscillations
450	55	13.5614	0.0033	814.8306	0.9966	Oscillations
450	70	13.1477	0.0047	819.096	0.9959	Chattering
500	**35**	**12.5739**	**0.0035**	**851.0151**	**1.0023**	
500	45	12.3785	0.0036	851.9463	1.0023	Oscillations
500	55	12.1793	0.0029	845.6223	1.0007	Oscillations
500	70	12.0472	0.0037	834.59	0.9982	Chattering
550	35	11.3684	0.0032	841.1842	1.0012	Oscillations
550	45	11.142	0.0034	837.8359	1.0012	Oscillations
550	55	11.0745	0.0024	842.6255	1.0006	Oscillations
550	70	10.8303	0.0025	819.2001	0.9962	Chattering

Table A.3 : Master manipulator orientation gain tuning at $k_p = 500$ and $k_d = 35$.

kpo	kdo	Task Position Total Error (m)	Task Position End Error (m)	Orientation Total Error	Orientation End Error	Notes
200	0	24.6035	0.0311	346.7391	0.3362	
400	0	42.2783	0.0556	265.0961	0.2895	
800	0	54.1142	0.0773	177.7183	0.2051	
1200	0	59.4874	0.1091	135.5432	0.1949	Unstable
1600	0	73.8259	0.1369	119.402	0.1701	Unstable
300	48	33.1908	0.0422	272.342	0.2907	
450	48	42.3428	0.0559	231.7743	0.2578	
640	48	50.5306	0.0696	200.9936	0.2274	
850	48	54.1817	0.0794	165.507	0.1966	
1050	48	56.548	0.0899	141.7074	0.1775	
300	30	33.2384	0.042	275.7581	0.2922	
300	70	33.1755	0.0412	264.3575	0.2827	
450	30	46.7367	0.0581	255.8022	0.2675	
450	70	42.1109	0.055	227.4402	0.2543	
640	30	50.0335	0.0691	200.1226	0.2274	
640	70	50.3178	0.0699	199.2274	0.2277	
850	30	53.651	0.0794	164.8625	0.1967	
850	70	53.759	0.0795	163.3231	0.1969	
1050	30	55.8038	0.0909	141.1015	0.1798	
1050	70	56.0999	0.0842	139.5064	0.1661	
200	30	25.1686	0.0305	321.3037	0.3195	
200	48	25.0246	0.0287	311.3767	0.3055	
200	**70**	**25.4065**	**0.0297**	**302.3862**	**0.304**	

Table A.4 : Integral gain tuning.

kp	kd	ki	kpo	kdo	kio	Task Position Total Error (m)	Task Position End Error (m)	Orientation Total Error	Orientation End Error	Notes:
500	35	0	200	70	0	24.2264	0.028	300.2301	0.3195	Baseline
500	35	100	200	70	0	18.7937	0.0181	291.0705	0.3065	Position Control, Kio=0
500	35	250	200	70	0	15.5498	0.0109	283.5091	0.2965	
500	35	500	200	70	0	12.7649	0.006	283.3721	0.2929	
500	35	1000	200	70	0	8.7202	0.0039	284.894	0.2946	
500	35	0	200	70	100	33.0076	0.054	280.8368	0.3755	Orientation Control, Ki=0
500	35	0	200	70	250	45.9362	0.0865	272.4347	0.41	
500	35	0	200	70	500	61.3363	0.1049	244.1072	0.3414	
500	35	0	200	70	1000	47.2782	0.0052	198.8951	0.1853	
500	35	500	200	70	250	19.5131	0.0266	247.3662	0.3593	Position and Orientation Integral Gains
500	35	500	200	70	500	24.0331	0.0251	214.5629	0.2986	
500	35	500	200	70	1000	27.1436	0.0089	161.7886	0.1705	
500	35	1000	200	70	250	12.9135	0.0151	241.7597	0.3542	
500	35	1000	200	70	500	14.8089	0.0128	205.8338	0.2892	
500	35	1000	200	70	1000	16.1595	0.0054	151.8348	0.1954	

APPENDIX B

Using Equation 2.16 for all links and writing these in a matrix form leads to equation B.1

$$\boldsymbol{V} = \Phi H\dot{\boldsymbol{\theta}} \quad where$$
$$\Phi = \begin{pmatrix} I_{6x6} & O_{6x6} & O_{6x6}. & \cdots & O_{6x6} \\ \phi_{2,1} & I_{6x6} & O_{6x6} & \cdots & O_{6x6} \\ \phi_{3,1} & \phi_{3,2} & I_{6x6} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_{n,1} & \phi_{n,2} & \phi_{n,3} & \cdots & I_{6x6} \end{pmatrix}$$
$$\phi_{k,k-1} = \begin{pmatrix} I_{3x3} & O_{3x3} \\ -\widehat{\boldsymbol{l}_{k-1,k}} & I_{3x3} \end{pmatrix} \tag{B.1}$$
$$H = \begin{pmatrix} \boldsymbol{H_1} & \boldsymbol{O_{6x1}} & \boldsymbol{O_{6x1}}. & \cdots & \boldsymbol{O_{6x1}} \\ \boldsymbol{O_{6x1}} & \boldsymbol{H_2} & \boldsymbol{O_{6x1}} & \cdots & \boldsymbol{O_{6x1}} \\ \boldsymbol{O_{6x1}} & \boldsymbol{O_{6x1}} & \boldsymbol{H_3} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \boldsymbol{O_{6x1}} & \boldsymbol{O_{6x1}} & \boldsymbol{O_{6x1}} & \cdots & \boldsymbol{H_n} \end{pmatrix}$$
$$\boldsymbol{H_k} = \begin{pmatrix} \boldsymbol{h_k} \\ \boldsymbol{O_{3x1}} \end{pmatrix}$$

Similarly, if the last link is the end effector, we have equation B.2.

$$\boldsymbol{V_e} = \Phi_e \Phi H\dot{\boldsymbol{\theta}} \quad where$$
$$\Phi_e = (O_{6x6} \quad \cdots \quad O_{6x6} \quad \phi_{e,n}) \tag{B.2}$$

Jacobian can then be identified as follows.

$$J = \Phi_e \Phi H \tag{B.3}$$

Equation 2.26 can be similarly rewritten in terms of a matrix as per equation B.4

$$\begin{aligned} \begin{pmatrix} \dot{\boldsymbol{\omega}}_k \\ \dot{\boldsymbol{v}}_k \end{pmatrix} &= \begin{pmatrix} I & 0 \\ -\widehat{\boldsymbol{l}_{k-1,k}} & I \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{\omega}}_{k-1} \\ \dot{\boldsymbol{v}}_{k-1} \end{pmatrix} + \begin{pmatrix} \boldsymbol{h_k} \\ 0 \end{pmatrix} \ddot{\theta}_k + \begin{pmatrix} \widehat{\boldsymbol{\omega}_{k-1}}\, \boldsymbol{\omega}_k \\ \widehat{\boldsymbol{\omega}_{k-1}}^{\,2}\, \boldsymbol{l}_{k-1,k} \end{pmatrix} \\ \dot{\boldsymbol{V}}_{\boldsymbol{k}} &= \qquad \phi_k \qquad\quad \dot{\boldsymbol{V}}_{\boldsymbol{k-1}} + \quad \boldsymbol{H_k}\, \ddot{\theta}_k + \qquad \boldsymbol{a_k} \end{aligned} \tag{B.4}$$

Collecting terms for all links leads to a matrix equation B.5

$$\dot{\boldsymbol{V}} = \Phi \mathrm{H} \ddot{\theta} + \Phi \boldsymbol{a} \tag{B.5}$$

From B.2 and B.3 we have the following relation

$$\boldsymbol{V_e} = \Phi_e \boldsymbol{V} = \Phi_e \Phi H \dot{\boldsymbol{\theta}} \quad \textbf{(B.6)}$$

Differentiating and using B.5 gives B.7

$$\dot{\boldsymbol{V}}_{\boldsymbol{e}} = \dot{\Phi}_e \boldsymbol{V} + \Phi_e \dot{\boldsymbol{V}} = \dot{\Phi}_e \boldsymbol{V} + \Phi_e (\Phi \mathrm{H} \ddot{\theta} + \Phi \boldsymbol{a}) \quad \textbf{(B.7)}$$

Similarly, differentiating B.8 we arrive at the equation B.9

$$\boldsymbol{V_e} = J \dot{\boldsymbol{\theta}} \quad \textbf{(B.8)}$$

$$\dot{\boldsymbol{V}}_{\boldsymbol{e}} = \dot{J} \dot{\boldsymbol{\theta}} + J \ddot{\boldsymbol{\theta}} \quad \textbf{(B.9)}$$

Finally equating equations B.7 and B.9 allows us to arrive with the expression for $\dot{J}\dot{\boldsymbol{\theta}}$.

$$\dot{J} \dot{\boldsymbol{\theta}} = \dot{\Phi}_e \Phi \mathrm{H} \dot{\boldsymbol{\theta}} + \Phi_e \Phi \boldsymbol{a} \quad \textbf{(B.10)}$$

Right hand side of the equation B.10 has been evaluated and reduced to a single 6x1 vector. The validity of the above derivation has been confirmed by direct comparison with the approach using analytical derivative of the Jacobian matrix. Results of the equation B.10, while still formidable in terms of required operations, are much more computationally friendly than the alternative. Furthermore, number of parameters required for the above calculation is already available from the kinematic engine.

APPENDIX C

Derivation presented bellow follows directly work by Siciliano and Villani (1999).

Axis/angle orientation error dynamics.

Equation describing orientation error dynamics is given.

$$\dot{\boldsymbol{\omega}}_{\boldsymbol{e}} + k_{do}\boldsymbol{\omega}_{\boldsymbol{e}} + k_{po}\boldsymbol{O}_{\boldsymbol{e}} = \mathbf{0} \qquad \textbf{(C.1)}$$

Extracting quaternion from relative rotation matrix R_{df} and expressing orientation error in the end effector frame as:

$$\boldsymbol{O}_{\boldsymbol{e}} = 2\eta_{df}\boldsymbol{\epsilon}_{\boldsymbol{df}} \qquad \textbf{(C.2)}$$

Selecting Lyapunov function of the following form:

$$V = 2k_{po}{\boldsymbol{\epsilon}_{\boldsymbol{df}}}^T\boldsymbol{\epsilon}_{\boldsymbol{df}} + \frac{1}{2}{\boldsymbol{\omega}_{\boldsymbol{e}}}^T\boldsymbol{\omega}_{\boldsymbol{e}} \qquad \textbf{(C.3)}$$

Leads to Lyapunov derivative:

$$\begin{aligned} \dot{V} &= 4k_{po}{\boldsymbol{\epsilon}_{\boldsymbol{df}}}^T\dot{\boldsymbol{\epsilon}}_{\boldsymbol{df}} + {\boldsymbol{\omega}_{\boldsymbol{e}}}^T\dot{\boldsymbol{\omega}}_{\boldsymbol{e}} \\ &= 2k_{po}{\boldsymbol{\epsilon}_{\boldsymbol{df}}}^T\boldsymbol{E}(\eta_{df},\boldsymbol{\epsilon}_{\boldsymbol{df}})\boldsymbol{\omega}_{\boldsymbol{e}} - k_{do}{\boldsymbol{\omega}_{\boldsymbol{e}}}^T\boldsymbol{\omega}_{\boldsymbol{e}} - 2k_{po}\eta_{df}{\boldsymbol{\omega}_{\boldsymbol{e}}}^T\boldsymbol{\epsilon}_{\boldsymbol{df}} \\ &= -k_{do}{\boldsymbol{\omega}_{\boldsymbol{e}}}^T\boldsymbol{\omega}_{\boldsymbol{e}} \end{aligned} \qquad \textbf{(C.4)}$$

Using following:

$$\dot{\eta} = -\frac{1}{2}\boldsymbol{\epsilon}^{\boldsymbol{T}}\boldsymbol{\omega} \qquad \textbf{(C.5)}$$

$$\dot{\boldsymbol{\epsilon}} = \frac{1}{2}\boldsymbol{E}(\eta,\boldsymbol{\epsilon})\boldsymbol{\omega} \qquad \textbf{(C.6)}$$

with:

$$\boldsymbol{E}(\eta,\boldsymbol{\epsilon}) = \eta I - \hat{\boldsymbol{\epsilon}} \qquad \textbf{(C.7)}$$

Since Lyapunov derivative is negative semi-definite, LaSalle theorem is invoked to prove asymptotic stability. System is characterized by three invariant sets.

$$\begin{aligned}
\mathcal{E}_1 &= \{\eta_{df} = 1, \boldsymbol{\epsilon_{df}} = \mathbf{0}, \boldsymbol{\omega_e} = \mathbf{0}\} \\
\mathcal{E}_2 &= \{\eta_{df} = -1, \boldsymbol{\epsilon_{df}} = \mathbf{0}, \boldsymbol{\omega_e} = \mathbf{0}\} \\
\mathcal{E}_3 &= \{\eta_{df} = 0, \boldsymbol{\epsilon_{df}}: \|\boldsymbol{\epsilon_{df}}\| = 1, \boldsymbol{\omega_e} = \mathbf{0}\}
\end{aligned} \qquad \textbf{(C.8)}$$

Invariant set $\mathcal{E}_3$ is unstable in light of C.9, and fact that V is a decreasing function as seen from equation C.4.

$$V_\infty = 2k_{po} \qquad \textbf{(C.9)}$$

This can be seen by taking small perturbation $\eta_{df} = \sigma$ about the equilibrium resulting in ${\boldsymbol{\epsilon_{df}}}^T \boldsymbol{\epsilon_{df}} = 1 - \sigma^2$, and perturbed Lyapunov function of:

$$V_\sigma = 2k_{po} - 2\sigma^2 k_{po} < V_\infty \qquad \textbf{(C.10)}$$

Consequently, system converges asymptotically towards $\mathcal{E}_1$ or $\mathcal{E}_2$. These invariant sets however, represent same orientation. The error equation in C.1 is therefore stable and converges to zero, achieving desired R_d and $\boldsymbol{\omega}_d$.

Quaternion orientation error dynamics.

Starting with the error dynamics equation:

$$\dot{\boldsymbol{\omega}}_{\boldsymbol{e}} + k_{do}\boldsymbol{\omega_e} + k_{po}R_f\boldsymbol{\epsilon_{df}} = \mathbf{0} \qquad \textbf{(C.11)}$$

Letting Lyapunov function to be

$$V = k_{po}((\eta_{df} - 1)^2 + {\boldsymbol{\epsilon_{df}}}^T\boldsymbol{\epsilon_{df}}) + \frac{1}{2}{\boldsymbol{\omega_e}}^T\boldsymbol{\omega_e} \qquad \textbf{(C.12)}$$

Time derivative, using C.5 through C.7, then is

$$\begin{aligned}
\dot{V} &= 2k_{po}(\,(\eta_{df} - 1)\dot{\eta}_{df} + {\boldsymbol{\epsilon_{df}}}^T\dot{\boldsymbol{\epsilon}}_{\boldsymbol{df}}\,) + {\boldsymbol{\omega_e}}^T\dot{\boldsymbol{\omega}}_{\boldsymbol{e}} \\
&= k_{po}\big(-(\eta_{df} - 1){\boldsymbol{\epsilon_{df}}}^T\boldsymbol{\omega_e} + {\boldsymbol{\epsilon_{df}}}^T\boldsymbol{E}(\eta_{df}, \boldsymbol{\epsilon_{df}})\boldsymbol{\omega_e}\big) \\
&\qquad - k_{do}{\boldsymbol{\omega_e}}^T\boldsymbol{\omega_e} - k_{po}{\boldsymbol{\omega_e}}^T\boldsymbol{\epsilon_{df}} \\
&= -k_{do}{\boldsymbol{\omega_e}}^T\boldsymbol{\omega_e}
\end{aligned} \qquad \textbf{(C.13)}$$

Again, LaSalle theorem is used to prove asymptotic stability. Invariant sets of the system are the same as for axis/angle problem, equation set C.8. Similarly, set $\mathcal{E}_3$ is unstable, as seen from C.13 and the fact that the Lyapunov function is a decreasing one.

$$V_\infty = 2k_{po} \tag{C.14}$$

Small perturbation $\eta_{df} = -1 + \sigma$ about the equilibrium, with $\sigma > 0$, results in $\boldsymbol{\epsilon_{df}}^T \boldsymbol{\epsilon_{df}} = 2\sigma - \sigma^2$, and perturbed Lyapunov function of:

$$V_\sigma = 4k_{po} - 2\sigma^2 k_{po} < V_\infty \tag{C.15}$$

Sets $\mathcal{E}_1$ and $\mathcal{E}_2$ represent same orientations and are stable. The error equation in C.11 is therefore stable and guarantees asymptotic convergence to zero. Desired orientation can be achieved.

CURRICULUM VITAE

Name Surname : Mateusz Szczesiak

Place and Date of Birth : Konin, Poland, September 24th 1980

E-Mail : matsz09@yahoo.com

EDUCATION

- **B.E.** : 2005, The City College of New York, Grove School of Engineering, Mechanical Engineering
- **M.A.** : 2007, The City College of New York, Division of Sciences, Physics

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2006-2014 Project Engineer at Honeybee Robotics Spacecraft Mechanism Corporation.

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS: