AN IMPEDANCE CONTROLLED MANIPULANDUM FOR HUMAN MOVEMENT STUDIES

by

Ian C. Fayé

B.S. M.E., University of California, Davis

(1983)

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

at the

Massachusetts Institute of Technology

June 1986

Signature of Author _____
Department of Mechanical Engineering, May 9, 1986

Certified by _____
Neville J. Hogan
Thesis Supervisor

Accepted by _____
Ain A. Sonin
Chairman, Departmental Committee on Graduate Students

AN IMPEDANCE CONTROLLED MANIPULANDUM

FOR

HUMAN MOVEMENT STUDIES

by

Ian C. Fayé

Submitted to the Department of Mechanical Engineering on May 9, 1986 in partial fulfillment of the requirements for the Degree of Master of Science.

ABSTRACT

The human arm is a highly nonlinear system with dynamic properties that have not yet been well characterised. To facilitate investigation of the dynamic properties of the arm and their role in the control of movement, a powered, two link, impedance controlled manipulandum has been developed. Impedance control is appropriate as the manipulandum must interact stably with a dynamic environment (the human subject). It also permits the environment experienced by the human subject to be modified under computer control.

The practical application of an impedance controller to a two link manipulator is presented. The impedance controller implemented here specifically attempts to control the manipulator's endpoint stiffness and viscosity. Verification of the ability to achieve the desired dynamics is presented for both dynamic and static (stiffness only) cases. The ability to control the endpoint force in the zero impedance case is also presented. Due to the nature of the actuators and the manipulator, the safety of both the subject and the experimenter are of major concern. Safety features incorporated into the apparatus are discussed in detail.

A mathematical model of the manipulator is presented and used with the impedance controller to simulate motion of the manipulator. Comparison of simulation results to actual motion of the apparatus serves to validate the model and to confirm the performance of the impedance controller.

Thesis Supervisor: Dr. Neville J. Hogan

Title: Associate Professor of Mechanical Engineering

ACKNOWLEDGEMENTS

I would first like to thank my parents and my family for the care and support that they have given me for the past twenty-six years. Thanks to my father who has certainly always been my inspiration in engineering. Thanks to my grandfather and my mother; their endless letters always kept my spirits high. I would especially like to thank my mother who has certainly outdone herself countless times on my behalf. It is to her that I dedicate this thesis.

A special thanks to Professor Neville J. Hogan, my advisor, whose insight, advice and guidance have certainly been beneficial to this project and my own growth as an engineer and a researcher. I look forward to further work with Neville. I would also like to thank Dr. Emilio Bizzi for his support and patience with this project.

Thanks to all of the people that are a part of the Biomechanics Lab and Emilio Bizzi's Lab. They have certainly made this work more enjoyable. Thanks to Ted Milner who was always helpful with programming questions and made an effort to get me out riding on the tandem. Certainly no thesis is complete without thanks to Bill Murray and Cary Abul-haj. I appreciate the many hours they spent helping me unravel some of the mysteries of electronics. Ed Colgate also deserves special thanks; he was extremely helpful with the equations and photographs for this thesis. I would also like to thank Tom Buchanan for his help in proofreading my rough drafts. I am especially grateful to Dawei Qi and Crispin Miller. Dawei, for his efforts in writing DYSYS for the Lab's PDP 11/60 and Crispin for his help with editing parts of this document. Thanks Pete, Mike, Greg, Kjirste, Keita, Dov, Qiang, John, Bob, Simon, Joe, Tom, Chris, Eric, Sandro,... you have all been great to work with!

I would also like to thank the administrators of the Whitaker College whose help was instrumental in the production of the figures for this thesis. They have always been very helpful and pleasant to work with.

Thanks to David and Wendy; they always kept me in touch with the real world.

Finally I would like to wish my brother the best of luck as he too pursues a career in the field of engineering.

This work was supported by NIH Grant No. 5 R01 NS09343 -Department of Health and Human Services. This work was performed in Dr. Emilio Bizzi's Lab in the Whitaker College of Brain Sciences and Human Behavior, and in the Eric P. and Evelyn E. Newman Laboratory for Biomechanics and Human Rehabilitation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

B_a	Effective damping of the apparatus at the handle (scalar).
B_c	Interface friction between inner and outer links.
$\underline{B}_e$	Desired endpoint viscosity matrix.
B_h	Effective damping of the human at the hand (scalar).
B	Effective damping
$\underline{B}$	Desired endpoint viscosity matrix.
$\underline{B}_\theta$	Joint viscosity matrix.
B_{tot}	Combined damping of subject and manipulator
BW	Bandwidth
δ	Logarithmic decrement
$\underline{F}$	Vector of endpoint forces.
h_n	Distance to the center of mass of link n.
I_n	Rotational inertia of link n about its center of gravity.
$\underline{I}_e$	Endpoint inertia matrix of the manipulator
$\underline{I}_\theta$	Absolute joint angle inertia matrix of the manipulator
$\underline{J}$	Jacobian transformation matrix.
J_n	Rotational inertia of link n about its axis of rotation.
K_a	Effective stiffness of the apparatus at the handle (scalar).
$\underline{K}_e$	Desired endpoint stiffness matrix.
K_h	Effective stiffness of the subject at the hand (scalar).
$\underline{K}$	Desired endpoint stiffness matrix.
K_{tot}	Combined stiffness of subject and manipulator.
K_t	Torque motor sensitivity
ℓ_n	Length of link n.
M_a	Effective mass of apparatus at the handle (scalar).
M_h	Effective mass of human at the hand (scalar).
M_{tot}	Combined mass of subject and Manipulator.
m	Mass
m_n	Mass of link n.
q_1	Joint coordinates
q_2	Endpoint coordinates
SF	Sampling frequency
$\underline{T}_c$	Vector of commanded actuator torques.
T_s	Sampling period
$\underline{\theta}$	Vector of absolute joint angles
θ_1, θ_2	Absolute joint angles
δW	Incremental work.
$\underline{V}$	Vector of endpoint velocities.
V_d	Digitally differentiated velocity approximation.
$\underline{\omega}$	Vector of absolute joint angular velocities.
ω_1, ω_2	Absolute joint angular velocities.
ω_n	Natural frequency
$\underline{X}$	Vector of endpoint positions.
$\underline{X}_v$	Vector of virtual endpoint positions.
X,Y	Absolute coordinate axes.
x,y	Manipulator endpoint position.
u,v	Manipulator endpoint velocity.
ζ	Damping ratio
ζ_h	Damping ratio of human subject

CHAPTER 1: INTRODUCTION

This thesis is concerned with the development of control strategies for a two-degree-of-freedom serial-link manipulandum. This manipulandum is designed to be used as an experimental apparatus for investigating human arm movements. Studying the dynamic parameters of these arm movements requires coupling between the apparatus and the human subject. This dynamic interaction has a profound effect on the stability and performance of the apparatus, therefore the control strategy used is impedance control [10,11]. The results of this work will also be relevant to the control of manipulators (1) in general.

The research community concerned with the physiology of motor control has become increasingly interested in two-link manipulators for several reasons. One reason is that the two-link manipulator can mimic the primate upper extremity: its inertial properties are inherently similar (as opposed to mechanisms such as cartesian robots) because of its serial link segmented structure; and with appropriate feedback control, it can also display dynamic behavior (such as stiffness and viscosity) (2) similar to the primate neuromuscular system.

---

(1) The apparatus described in this thesis is a type of mechanism commonly referred to as a "two-link manipulator" by extension from robotics terminology (the configuration is similar to that of a SCARA robot). While this particular application is more accurately referred to as a manipulandum (since the subject grasps it and may attempt to control its motion), the generic mechanism will also be referred to, in this work, as a manipulator.

(2) The manipulator is comprised of two rigid links, held together by pin joints that have no associated inherent stiffness. The only damping in this mechanism is small, as low friction bearings are used at the pin joints.

Thus it is possible to postulate movement control strategies adopted by the human brain and then evaluate the performance of these strategies with a physical piece of hardware. This will, in general, help in understanding issues involved in movement control both in manipulators and in the biological system. Note that while there is no guarantee that the biological computation is analogous to machine computation, the mechanics that apply to the control of the two-link manipulator must also be addressed by the brain in the control of movement [1].

However, the primary interest of the physiological research group for whom this apparatus was built is to develop a manipulandum that can provide disturbances to a human arm in order to study its dynamic properties. These disturbances can be in the form of either forces or displacements imposed on the hand of a human subject. In this work, the control strategy used to generate these disturbances is one known as impedance control.

1.1 Impedance Control

During the the last decade, the use of multi-link manipulators in industrial manufacturing has grown tremendously. As the use of manipulators in industry has become so extensive, the need for practical real-time control strategies has increased. One approach to controlling manipulators is the impedance control strategy developed by Hogan [11]. Impedance control is designed to deal with dynamic interaction (i.e., the coupling of the controlled system to a dynamic environment).

Nearly all other control strategies attempt to control some set of system variables (e.g., endpoint postion, joint angles, etc.). These variables form a vector that is generally some function of the system state variables. In contrast, the primary objective of an impedance controller is to control the relationship between conjugate power variables (such as force and velocity) at an interaction port. An interaction port is defined as the point where the controlled system (e.g., the two-link manipulator) is coupled to the environment (e.g., the human subject).

In its most general form, the relation between conjugate power variables is a set of nonlinear time-varying functions. Thus the impedance controller is specifying the mapping between one vector of variables and its power conjugate vector of variables. The two-link manipulator, in this work, is controlled using one particular form of an impedance controller. The goal of this impedance controller is to control the apparent stiffness and viscosity of the manipulator as seen by the human

subject holding the handle at the distal end of the manipulator. The modification of apparent endpoint inertia would be possible with force feedback, but will not be addressed in this document.

1.2 Ideal Manipulandum

Ideally an apparatus used to study human movement should be capable of providing any force at the endpoint, while simultaneously maintaining any impedance from zero to infinte. This ideal apparatus should also be able to generate the required force instantaneously. Thus it would be possible to perturb the subject and make measurements before the neuromuscular system could respond to the perturbation.

Physical limitations of mechanical actuators and sensors restrict the ability to implement the ideal device. The torque motors that drive the system are limited by their ability to generate torque, and also by their mechanical time constant. Thus the torque motors have some finite bandwidth that will limit the performance of the practical manipulandum. Sensors that provide kinematic information to the impedance controller are also limited by bandwidth, and by finite resolution as well. The implementation of an infinite range of impedances would require sensors with infinite dynamic range (the ratio of the largest and smallest measurable quantities). With these limitations of the actual apparatus, a portion of this work involves quantifying the actual range of achievable impedances and endpoint forces.

1.3 Related Work and Comparison to Current Project

1.3.1 Physiological

In recent years several studies of human movement have been done utilizing a two-link manipulandum as the experimental apparatus. [1,6,13,14,18]. The device has been used in various ways, in both powered and unpowered modes. A brief overview of some of this previous work will be made in this section.

As a passive (unpowered) device, the apparatus is used to record the position of the subject's hand during movements. Regardless of the functional mode of the device (passive or active), the apparatus restricts the subject's hand to a single plane of motion during the movement. This serves to reduce the complexity of the experiment, by reducing the degrees of freedom of the motion of the hand and eliminating the effect of gravity.

Active elements (such as D.C. torque motors and/or magnetic clutches) coupled to the manipulandum enable the experimenter to provide disturbances to the hand of the subject. These disturbances may consist either of constraining the hand to a particular path (as with a magnetic clutch applied to one axis) or applying forces to the hand using the D.C. torque motors.

First, consider the passive manipulandum. For these experiments, subjects held the handle and made completely free, undisturbed arm move-

ments (except for the inherent constraint to a horizontal plane) [1,6,14]. The passive two-link manipulandum was used as a kinematic data acquisition device to measure the location of the subject's hand. Flash [6] used this apparatus to collect data on human movements between designated target locations as shown in figure 1.1.

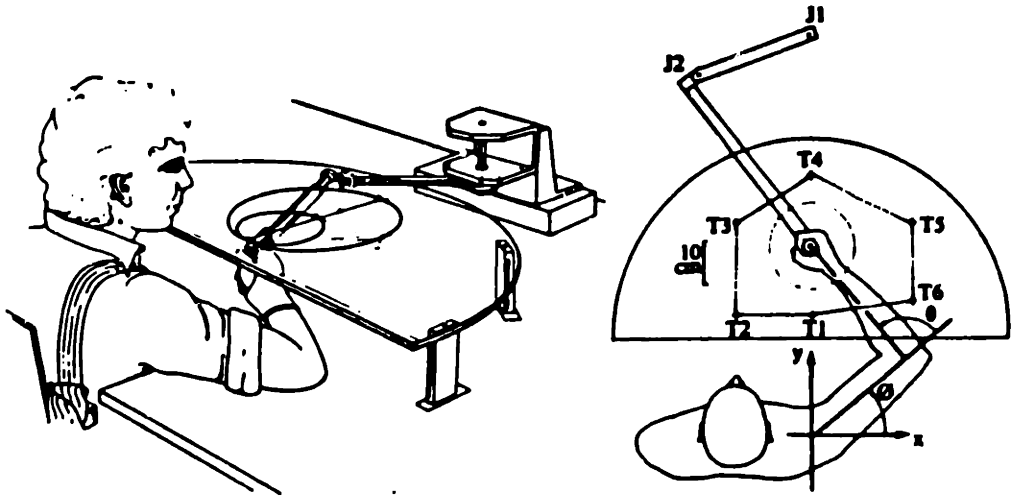


Figure 1.1 Experimental Apparatus with Human Subjects

The next experiment in this discussion used a two-link manipulator to constrain the motion of the subject's hand during a movement [13]. By using a magnetic clutch at the axis of rotation of the inner link, it was possible to kinematically constrain the subject's movements. Activating the clutch would constrain the motion of the handle to a circle with a radius equal to the length of the outer link. In this set of experiments, the interaction force at the handle was measured as well as the endpoint position. This interaction force is the force that the subject applies to the handle.

The final experiment in this discussion used two active torque inputs (D.C. torque motors) coupled to a two-link manipulandum. With torque motors it is possible to drive the manipulandum around its workspace. The workspace is defined as the complete set of possible linkage configurations, given the physical limits of the hardware. Using appropriate torque commands, the manipulandum was used to apply perturbations to the hand of a subject in the form of small displacements of the hand. Recording the force associated with these displacements allowed the computation of an apparent endpoint stiffness of the subject's arm, associated with the direction of the applied displacement. In this way it was possible to determine the tensor of apparent endpoint stiffnesses for a number of different arm configurations.

Successful and informative as these experiments were, they would benefit from an improved apparatus. The case where a manipulator was used passively to study the kinematics of movement would have benefited from an active manipulator control algorithm that made the effective endpoint inertance (or mass) zero or at least small. Thus the subject would not have experienced any load during a movement, and hence the device would have provided no dynamic encumbrance.

Conversely, the experiments done with the magnetic clutch on the inner link are a good example of a device with infinite (or at least large) impedance. By design, the apparatus in the magnetic clutch experiments was restricted to infinite impedance only for the degree of freedom associated with a single link. An ideal apparatus, however, would be capable of generating infinite impedance in any chosen direc-

tion (or in all directions) at the endpoint. This capability, combined with the ability to generate any desired force at the endpoint, would result in experimental hardware that could produce a variety of different disturbances to the subject. For example, with an infinite impedance device it would be possible to displace the subject's hand a precisely controlled amount for the stiffness experiments described above.

1.3.2 Impedance Control

Work done by Cotter [4] showed the feasibility of implementing a nonlinear impedance controller on a two-degree-of-freedom manipulator. While this work demonstrates the practical application of a nonlinear impedance controller, it does not address the use of force feedback in the controller.

Force feedback was incorporated into the impedance controller for a pneumatic/hydraulic system built by Kleidon [17]. This system, however, is limited to one degree of freedom. The goal of Kleidon's work was to implement a system whose inherent dynamics could be modified with pneumatics and hydraulics and then fine tuned with electronic feedback.

The first successful demonstration of a complete nonlinear impedance controller [11] for a two-link manipulator with force feedback was done by Wlassich [25]. With this work it has been shown that force feedback facilitates the modification of the apparent endpoint inertia. The manipulator used in Wlassich's work is very similar to the manipula-

tor used in this work. However, that manipulator was only intended to act as test bed for demonstrating the nonlinear impedance controller and could not be used with human subjects.

As the stiffness experiments by Mussa-Ivaldi might suggest, the concept of impedance control can be applied to biological systems as well as to machinery. There is reason to suspect that primates modify their dynamics (or effective stiffness) by co-activation of agonist and antagonist pairs of muscles. Interest in this area has motivated investigations into the role of co-activation and maintaining postural stability [8,9] and the construction and control of an above elbow prosthesis simulator aimed at investigating the use of co-activation by amputees [2].

Work done by Colgate [3] involved the design and construction of a dynamics measuring device. Using air jets attached to the subject's wrist, this device is intended to apply forces to the human arm in order to measure the dynamic properties of the limb. As a force generator this device can be considered a near-zero-impedance device. The addition of position feedback to this device would make it an impedance device as well, but accurate real-time measurement of position could be quite difficult.

The work that is presented here focuses on the control of an apparatus that can be used as both a high- and low-impedance device. This apparatus will be capable of generating both forces at the endpoint and desired endpoint impedances. Each of these desired endpoint charac-

teristics will require its own specific algorithm. For the impedance controller it will be necessary to use closed-loop feedback control, but for force generation it will be possible to use open loop control. However, due to the nonlinear relation between motor torque and endpoint force, the orientation of the linkage must be constantly monitored. Although the force generation is closed-loop, it is important to point out that there is no measurement of force in this control, either direct or inferred (i.e., an observer based on an internal model of the dynamic system).

1.4 Goals

The primary goal of this research is to implement an impedance controller that will allow an experimeter to modify the set of endpoint dynamics that the subject experiences at the handle of the manipulandum. In terms of impedance control topics, a secondary goal of this work is to develop an apparatus that can be used not only for physiological experiments, but also for further investigation of some of the various control topics discussed above (such as force feedback).

Both the apparent stiffness and viscosity will be modified. Modification of apparent endpoint inertance will not be implemented, however, since force feedback is not yet provided. The performance of the impedance controller will be evaluated, and the range of achievable stiffnesses and viscosities, given the limitations of the mechanical hardware and control circuitry, will be qualitatively evaluated. Finally, simple experiments will be performed with subjects to demon-

strate how this apparatus can be used in the study of the human upper extremity.

1.5 Summary Of Remaining Chapters

Chapter 2 includes a discussion of impedance control and a presentation of the impedance control algorithm used in this work. This chapter addresses computational and control issues concerning both real-time control and the rates of feedback gain changes in the workspace. Chapter 2 also includes a discussion of force production at the manipulator endpoint.

Chapter 3 is a detailed description of the mechanical hardware and safety precautions used.

Chapter 4 covers the sensors and control circuitry needed to implement the impedance controller. This chapter includes a discussion of the problems associated with structural resonance of the support holding the apparatus, and a solution is presented that minimizes its effects.

A nonlinear model of the mechanical hardware is developed and presented in chapter 5. A method of simulation is discussed and the mathematical model is adapted for the simulation.

Chapter 6 evaluates the ability of the impedance controller to produce the desired dynamics in both static and dynamic cases. This chapter also evaluates the production of forces at the endpoint, but

only in the static case. In this chapter an evaluation of the system controller is made, for large amplitude motion, by comparing experimental data with simulations. The final section of this chapter contains the data taken in the human movement experiments.

Chapter 7 contains a final discussion and recommendations for further work with this particular system.

CHAPTER 2: CONTROLLER THEORY AND IMPLEMENTATION

In the introduction it was stated that the control of a two-link manipulator used to study arm movements would present a practical implementation of an impedance controller. This chapter discusses the reasons why the impedance control strategy was adopted in this work. The implementation of a particular impedance control algorithm as well as some of the computational issues facing practical implementation will also be discussed. In the later part of this chapter the topic of endpoint force generation is addressed.

2.1 Impedance Control

The impedance control approach was adopted here because it provides a straightforward approach to controlling a physical piece of hardware that is designed to interact with a dynamic environment. The human arm is certainly a highly nonlinear dynamic system. It is comprised of limbs with nonlinear inertial properties and muscles that act as both nonlinear force actuators and as nonlinear springs [12,18]. By definition, the impedance controller is designed to work with systems that are coupled to dynamic environments (in this case, the human arm). It is these points of coupling that are considered the interaction ports between the systems. In this system, the subject (the environment) is coupled to the manipulator (the controlled system) at a single port of interaction, the handle at the manipulator's endpoint.

Given energetic physical systems, a fundamental restriction is

imposed on their interaction: it is not possible for one of two interacting systems to prescibe both the effort (e.g. force) and the flow (e.g. velocity) at the point of interaction [21]. At a specific point of interaction, these variables (effort and flow) are known as power conjugate variables. Their product is the amount of power that is transfered across that interaction point.

A control strategy that precisly controls the position (intergrated flow variable) of a physical system, cannot also control an interaction force (effort) that would arise upon contact with another physical system. Similarly, a control strategy that focuses on the control of the interaction force has no control of the position in that direction. These ideal controllers (position or force) are restricted in this way because they attempt to control only a subset of system variables.

The impedance control approach does not attempt to control any of the system variables, but rather the relationship between these variables and a "virtual trajectory", [10]. The impedance controller generates a set of output forces as a function of the measured kinematic state of the system (position, velocity, etc.). In steady state, the force is only a function of the position. This function is restricted such that zero force defines some unique position. This position (where the force is zero) is defined as the "virtual position"; a time history of virtual positions define a "virtual trajectory". The impedance controller can now be redefined so that it generates a set of output forces as a function of the measured kinematic state of the system and the virtual trajectory. Thus forces are generated to drive the actual position

to the virtual position. The forces only go to zero at equilibrium when the commanded virtual position and the actual position are the same. In this way control of one's position in the absence of interaction forces is achieved.

The term "impedance" is used because the environment is assumed to be an admittance (e.g. a mass, or a kinematic constraint). That is, the environment moves when forces are applied to it. Any dynamics due to compliance (e.g. springs) or damping, contribute forces that appear behind the admittance. Since the environment appears to the manipulator as an admittance, causality dictates that the manipulator appears to the environment as an impedance [16,23]. The assumption that the environment is an admittance, is not unreasonable since the human arm is made up of limb segments with inherent inertial properties that can be transformed to appear to the manipulator as an admittance.

2.2 Control Alogorithm

The control, in this work, was performed purely in the discrete time domain. The algorithm adopted here was developed in the analog domain and then implemented on a digital computer. Appendix G contains a listing of the control program used to control the manipulandum. With this program, the sampling rate of the digital computer is 286 hertz, which is fast enough so that the discretized algorithm closely approximates the desired analog algorithm (see section 2.3 on computational complexity). For this application the desired impedance is linear and constant in the cartesian endpoint coordinates of the manipulator. This

means that as the manipulator moves around in its workspace, the impedance seen at the endpoint is constant. Hence the joint coordinate impedance is both varying and nonlinear for any motion of the manipulator because it is expressed in a different (non-cartesian) coordinate system.

Impedance control does not postulate a particular strategy for maintaining endpoint dynamics, but several approaches have been proposed and tested [4,17,25]. One particular approach proposed by Hogan [11] attempts to modify the effective endpoint inertance, viscosity, and stiffness:

$$\begin{aligned} \underline{T}_C = {} & \underline{\underline{I}}(\underline{\theta})\underline{\underline{J}}^{-1}(\underline{\theta})\underline{\underline{M}}^{-1}\underline{\underline{K}}[\underline{X}_v - L(\underline{\theta})] + \underline{S}(\underline{\theta}) \\ & + \underline{\underline{I}}(\underline{\theta})\underline{\underline{J}}^{-1}(\underline{\theta})\underline{\underline{M}}^{-1}\underline{\underline{B}}[\underline{V}_v - \underline{\underline{J}}(\underline{\theta})\underline{\omega}] + \underline{V}(\underline{\omega}) \\ & + \underline{\underline{I}}(\underline{\theta})\underline{\underline{J}}^{-1}(\underline{\theta})\underline{\underline{M}}^{-1}\underline{F}_{int} - \underline{\underline{J}}^{T}(\underline{\theta})\underline{F}_{int} \\ & - \underline{\underline{I}}(\underline{\theta})\underline{\underline{J}}^{-1}(\underline{\theta})\underline{G}(\underline{\theta},\underline{\omega}) + \underline{C}(\underline{\theta},\underline{\omega}) \end{aligned} \qquad [2\text{-}1]$$

The algorithm adopted in this work makes no attempt to modify the effective endpoint inertance and thus reduces equation 2-1 to the simplified case:

$$\underline{F} = \underline{\underline{K}}_e(\underline{X}_v - \underline{X}) - \underline{\underline{B}}_e\underline{V} \qquad [2\text{-}2]$$

This algorithm modifies the effective cartesian endpoint and stiffness of the manipulator, $\underline{\underline{K}}_e$ and $\underline{\underline{B}}_e$. Endpoint positioning, in this algorithm, is done by specifiying a virtual trajectory, $\underline{X}_v$, and computing the difference between this virtual endpoint position and the actual endpoint position. This difference is then multiplied by a feedforward gain matrix (stiffness, $\underline{\underline{K}}_e$) to result in driving forces. Damping, $\underline{\underline{B}}_e$, is included in the system by feeding the endpoint velocity through a set

of velocity feedback gains and subtracting the resulting force from the driving forces due to the stiffness K_e. This is all shown below in block diagram form (in joint coordinates), Figure 2.1.

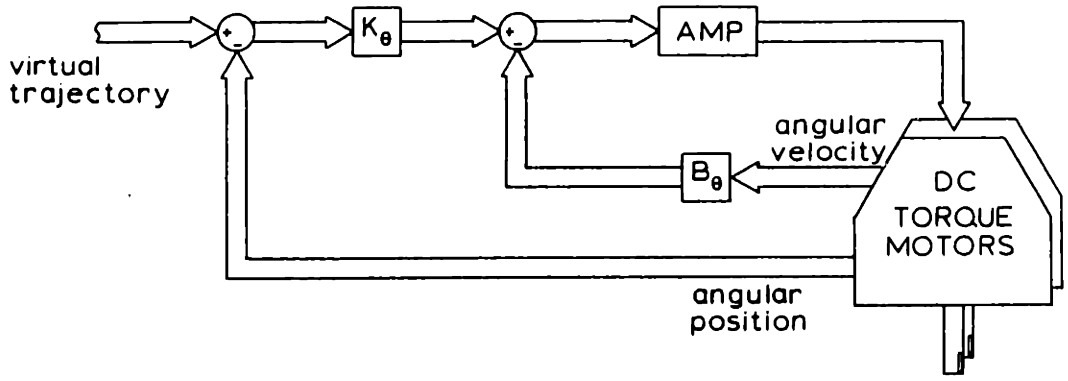


Figure 2.1
IMPEDANCE CONTROLLER IMPLEMENTATION

The actuators and sensors act purely in joint coordinates. Therefore, it is necessary to make transformations to find the current cartesian endpoint position and to compute the needed control torque. The Jacobian is defined as the relation between differential displacements of the cartesian endpoint position, dx, and the manipulator's absolute joint angles, dθ:

$$\delta \underline{X} = \underset{\sim}{J} \delta \underline{\theta} \qquad [2\text{-}3]$$

These angles are considered "absolute" because they are both measured with respect to a fixed non-accelerating reference frame.

The transformation between joint torques and cartesian endpoint forces is shown using the principle of virtual work. The work done for any force (or torque) through a differential displacement is just the dot product of that transposed force (or torque) vector and the differential displacement vector:

Translation-

$$\delta W_f = \underline{F}^T \cdot \delta \underline{X} \qquad [2\text{-}4]$$

Rotational-

$$\delta W_T = \underline{T}^T \cdot \delta \underline{\theta} \qquad [2\text{-}5]$$

Substitution of the Jacobian definition (relation 2-3 above) into equation 2-4 gives:

$$\delta W_f = \underline{F} \cdot \underset{\sim}{J} \delta \underline{\theta} \qquad [2\text{-}6]$$

When the amount of work done is the same for rotation and translation, equating the differential work for each case gives:

$$\underline{F}^T \cdot \underset{\sim}{J} \delta \underline{\theta} = \underline{T}^T \cdot \delta \underline{\theta} \qquad [2\text{-}7]$$

This is now rewritten in the form:

$$\underline{T} = \underset{\sim}{J}^T \underline{F} \qquad [2\text{-}8]$$

It is this relation that will be used to make all the transformations between joint torques and cartesian endpoint forces.

Finally the current endpoint position is needed for comparison to the virtual endpoint position. This is a transformation from measured joint angles to endpoint coordinates, using the forward kinematic equations:

$$\underline{X} = L(\underline{\theta}) \qquad [2\text{-}9]$$

Where L() is a function of the joint angles. Combining the relations 2-2, 2-8, and 2-9 yields the control law:

$$\underline{T}_C = \underset{\sim}{J}^T \underset{\sim}{K}_e (\underline{X}_v - L(\underline{\theta})) - \underset{\sim}{J}^T \underset{\sim}{B}_e \underset{\sim}{J} \underline{\omega} \qquad [2\text{-}10]$$

Where T_c is the vector of command torques that is sent to the actuators. It is important to point out that this algorithm does not require the computation of inverse kinematics to command the endpoint location.

With relation 2-10 the endpoint location need only be specified in cartesian endpoint coordinates. This is a rather significant savings in terms of copmutational workload of the control algorithm.

It is also important to note that this simplified algorithm has some extremely strong stability robustness properties. It has been shown that it can guarantee the stability of the manipulator endpoint. Even more important, it has been shown that if a nonlinear impedance controlled manipulator is stable in isolation, then when it is coupled to a dynamic environment of arbitrary complexity, which is also stable in isolation, the dynamic coupling does not jeopardise its stability.

2.3 Computational Complexity

For a digital control algorithm the execution speed of the real time loop is an important consideration in maintaining stability of the complete system [7]. All of the programming for the control of the two link manipulator was done in the language "C". This language was chosen because it is a higher level language that allows vector addressing. In this way it is possible to do high level programming (implementation of equation 2-10) and address the interface hardware (such as analog to digital converters, parallel interfaces and the real time clock) with the same piece of computer software.

One drawback to programming in C is that trigonometric operations (sine and cosine), that are necessary in computing the Jacobian, execute extremely slowly in C. With a PDP 11/73 a simple cosine computation can

take as long as 800 microseconds. This, in comparison to 80 microseconds for a floating point multiply, is quite a time consuming operation. The trigonometric operation time can be reduced by precomputing all of the sines and cosines and storing them in a table in the computer's extended memory.

In this apparatus, the position of the links is measured, in absolute joint angles, using optical encoders. These angles are considered "absolute" because they are both measured with respect to a fixed non-accelerating reference frame. The input to the computer from the optical encoders is a number between 0 and 4000 (0 to 180 degrees). This number is used as an index for a table stored in the computer's exteneded memory. The numbers that are stored at the location addressed by the index, are the cosine(i), sine(i), cosine2(i), sine2(i) and cosine(i)sine(i). The "i" is the actual angle reperesented by the encoder value. A complete read from the table in extended memory for a single angle takes only 425 microseconds. This read generates sine and cosine and three floating point multiplies. The three floating point multiplies were included because the added time to read them from extended memory is less than the time that the three floating operations would have taken. The two trigonometric operations and three floating point multiplies normally would have taken almost 2 milliseconds per angle, but with the extended memory read, both angles are processed in less than 1 millisecond (850 microseconds).

All of the control operations are done in floating point arithmetic and the final torque command is then scaled to the appropriate integer

units for the digital-to-analog converter. With floating point arithmetic and the trigonometric lookup table it was possible to reduce the execution time of the real time loop to 3.5 milliseconds. This sampling frequency (SF) of 285.7 hertz gives a practical controller bandwidth (BW) of 14.3 hertz according to relation 2-11 below. This bandwidth seems to be sufficient for the control of the apparatus.

$$BW = \frac{SF}{20} \qquad [2\text{-}11]$$

2.4 Gain Changes

The desired stiffness and viscosity are constant in endpoint coordinates. This means that the joint stiffness and viscosity change as the manipulator configuration changes. In terms of manipulator kinematics, these terms change quite significantly as the manipulator moves around in its workspace. The sampling rate used with the digital control algorithm must always be fast enough for the discrete implementation to approximate the analog equivalent, but it must also be fast enough to accomodate gain changes during fast movements. This is not of much concern for slow movements, but with fast movements (endpoint speed greater than 2 m/sec) is important to have a digital sampling rate that is fast enough to make the appropriate gain changes to maintain constant endpoint stiffness and viscosity. This section demonstrates the need to have a controller that changes or updates the effective joint angle feedback gains based on manipulator orientation.

The transformation from endpoint to joint viscosity is made with the Jacobian and its transpose defined in equation 2-3 above.

$$\underset{\sim}{B}_{\theta} = \underset{\sim}{J}^{T}\underset{\sim}{B}_{e}\underset{\sim}{J} \qquad [2\text{-}12]$$

Equation 2-12 is computed for the implementation of the impedance control algorithm, 2-10.

The transformation from cartesian endpoint viscosity to joint viscosity can be readily examined by assuming isotropy. This is a reasonable assumption since any general viscosity matrix can be re-written as the linear combination of the isotropic matrix and some other non-uniform matrix. Mathematically, the simplest case of isotropy can be obtained by assuming that the desired endpoint viscosity is equal to the identity matrix:

$$\underset{\sim}{B}_{e} = \underset{\sim}{I} = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \qquad [2\text{-}13]$$

In this case equation 2-12 reduces to:

$$\underset{\sim}{B}_{\theta} = \underset{\sim}{J}^{T}\underset{\sim}{J} \qquad [2\text{-}14]$$

By further assuming that the lengths of the manipulator links are equal to unity, it is easy to observe how the transformed matrix varies as the endpoint of the manipulator moves around in the workspace. It is important to stress that this transformed matrix represents the effective joint angle gains needed to maintain constant endpoint viscosity. This section examines particular gain changes which are components of these transformed matrices of gains.

For the special conditions of this example (equal diagonal terms and zero off-diagonal terms) it is shown in Appendix D that the diagonal terms of the transformed matrix do not change and remain unity. Figure 2.2 shows the variation in the offdiagonal term of equation 2-14 as the endpoint is moved along straight lines in the workspace. The plot is

comprised of lines of constant y endpoint position as the x position varies from one extreme of the workspace to the other.

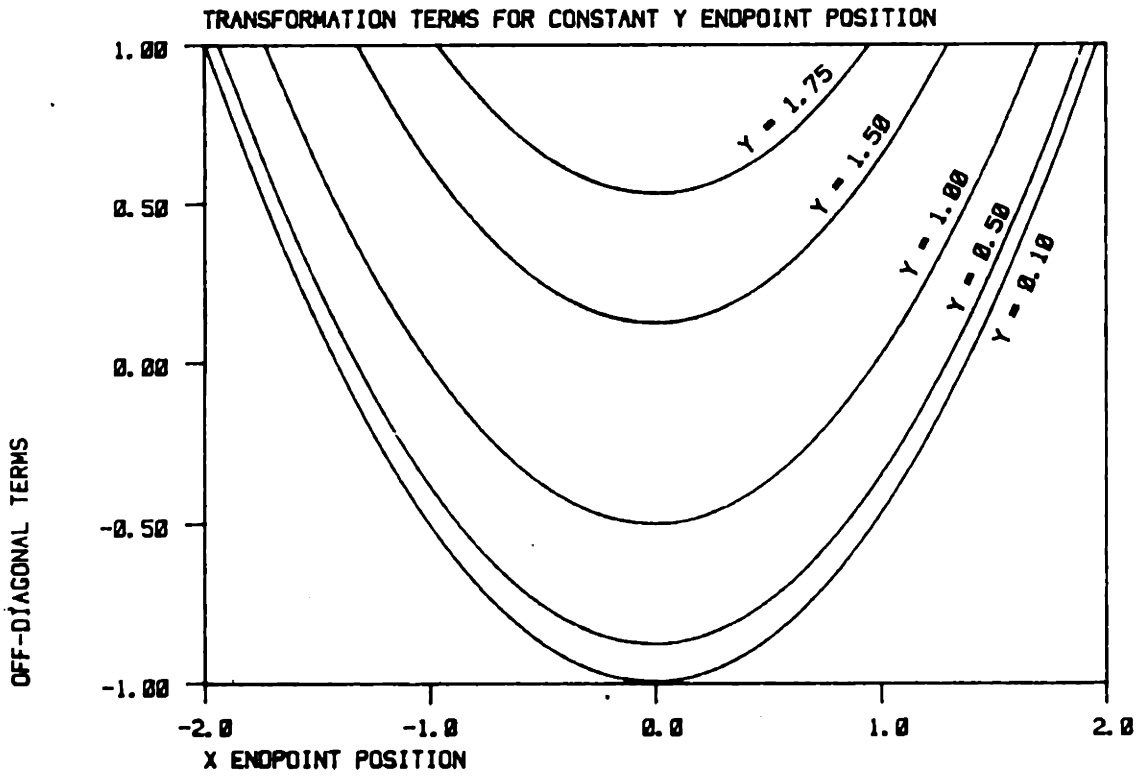


Figure 2.2 Off-diagonal Transformation Terms for Constant Y

An interesting observation of Figure 2.2 can be made at the extreme points in the workspace where the Jacobian loses rank and its determinant becomes zero. If the determinant of the Jacobian and its transpose become zero, then the determinant of $\underset{\sim}{B}_{\theta}$ in equation 2-14 must also be zero. For the determinant of the matrix $\underset{\sim}{B}_{\theta}$ in equation 2-14 to go to zero, as the Jacobian loses rank at the extremes of the workspace, the off-diagonal terms of matrix $\underset{\sim}{B}_{\theta}$ must approach unity. Figure 2.2 shows that this is indeed the case. These off-diagonal terms must go to unity because the diagonal terms in the desired endpoint viscosity matrix are

both unity and the link lengths were also chosen to be unity. For the determinant of matrix $\underset{\sim}{B}_e$ to then be zero, the off-diagonal terms must be unity.

Figure 2.3 shows a plot of the offdiagonal terms when the x endpoint position is held constant and the y endpoint position varies from zero to its extreme. The plot is symmetric about the line formed for x equals zero. Again the offdiagonal terms must approach unity as the manipulator extends to the extremes of the workspace.

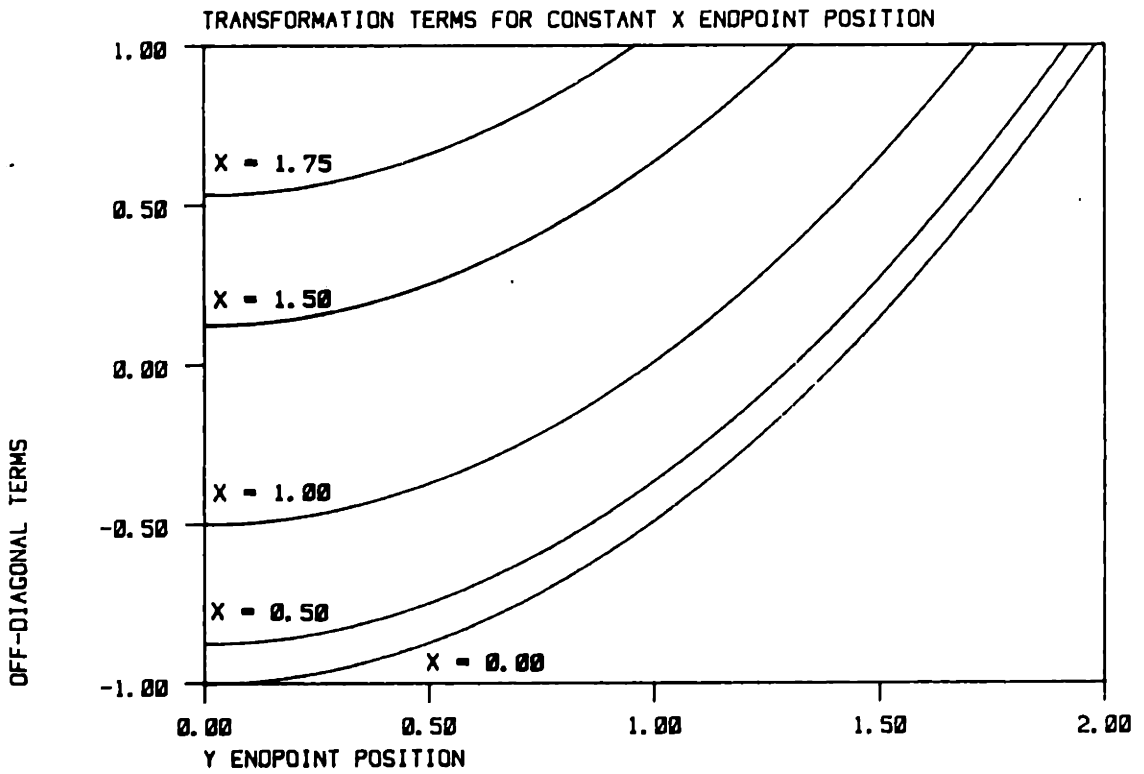


Figure 2.3 Off-diagonal Transformation Terms for Constant X

It is apparent that Jacobian transformed terms change significantly. For the case y = 1.0 in figure 2.2, even if the workspace is restricted to + 1.3 (meters), the change in the offdiagonal term is

almost one hundred percent. Not only that, but the slope changes rather drastically (from 0 to 2.145 1/meter) over this range of x values. Thus the need for a dynamics controller that changes or updates the joint angle feedback gains based on manipulator location is clearly demonstrated.

2.5 Endpoint Force Production

One of the goals of this work is to provide an apparatus that will be capable of producing forces at the handle in any desired direction. Using the torque motor servo amplifier system described in the next chapter, it is possible to generate torques about the axis of rotation of each link. The maximum output of the torque motors determines the maximum forces that can be generated at the the endpoint of the manipulator.

Now the question arises, how much force is achievable for a particular direction at the handle (the manipulator endpoint)? In general the maximum forces for all the directions around the endpoint form a parallelogram as shown in Figure 2.4.

This can be explained by looking at the kinematics of the apparatus. If the torque on the inner link is maximum (positive or negative) and the torque on the outer link is zero, then a maximum force is generated along the axis of the outer link as shown in Figure 2.5a. Likewise if the torque on the inner link is zero and the torque on the outer link is maximum, then a maximum force is generated parallel to the inner link as shown in Figure 2.5b. When both links experience maximum

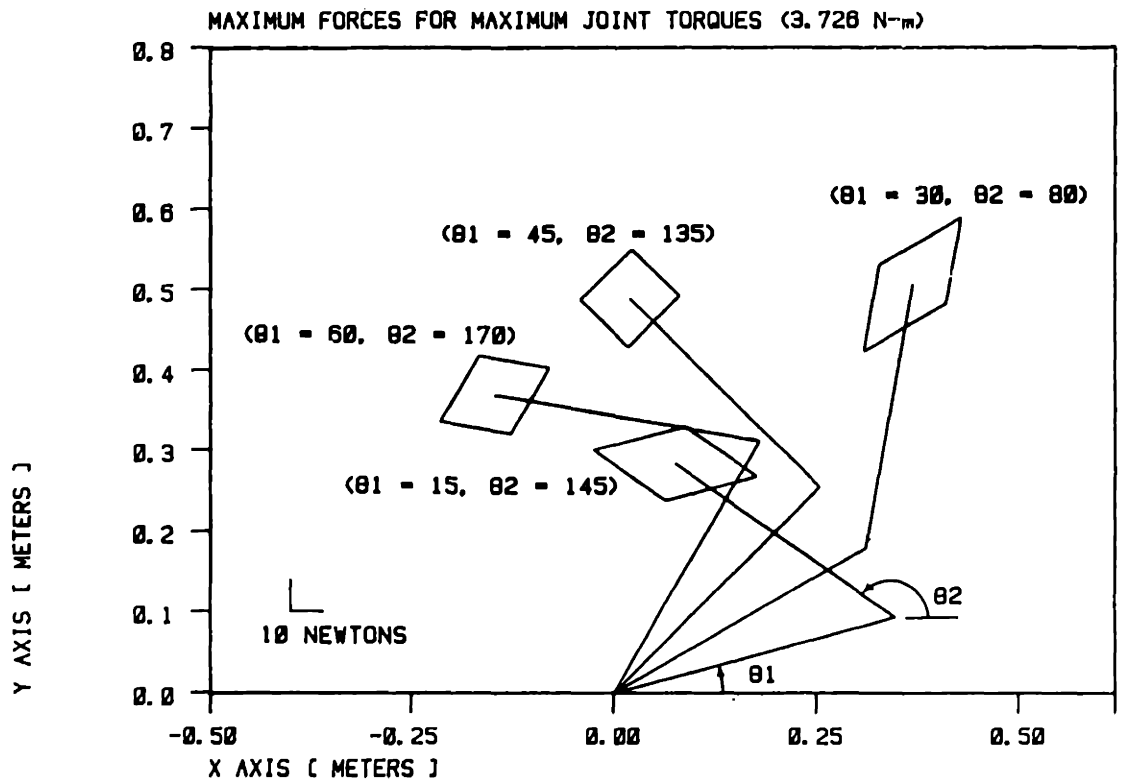


Figure 2.4 Maximum Endpoint Forces for Several Orientations

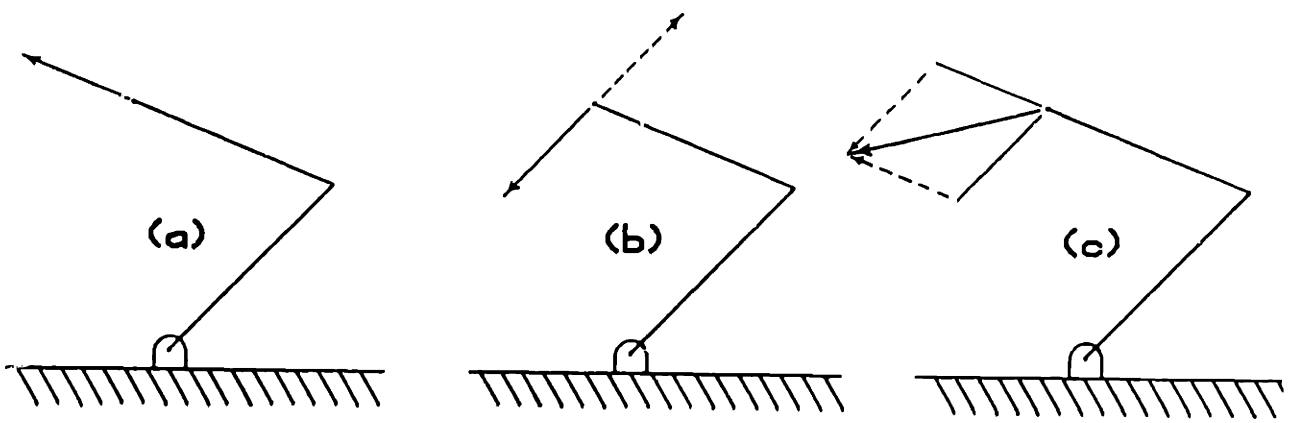


Figure 2.5 Maximum Endpoint Force Development

Figure 2.5a Maximum Torque on the Inner Link

Figure 2.5b Maximum Torque on the Outer Link

Figure 2.5c Vector Addition to Find Maximum Endpoint Forces

torques, the forces from each individual case above will add vectorially as shown in Figure 2.5c. This maximum force makes up one of the corners of the parallelogram shown in figure 2.4.

For this reason the shape of the parallelogram depends only on the relative angles between the links (and reduces to a square when $\theta_2 - \theta_1 = 90$ degrees). The orientation of the parallelogram is dependent only on the inner link angle. For this reason, only one set of parallelograms needs to be plotted. The set of parallelograms that arise as the manipulator endpoint moves away from the shoulder along a straight line, shown in Figure 2.6.

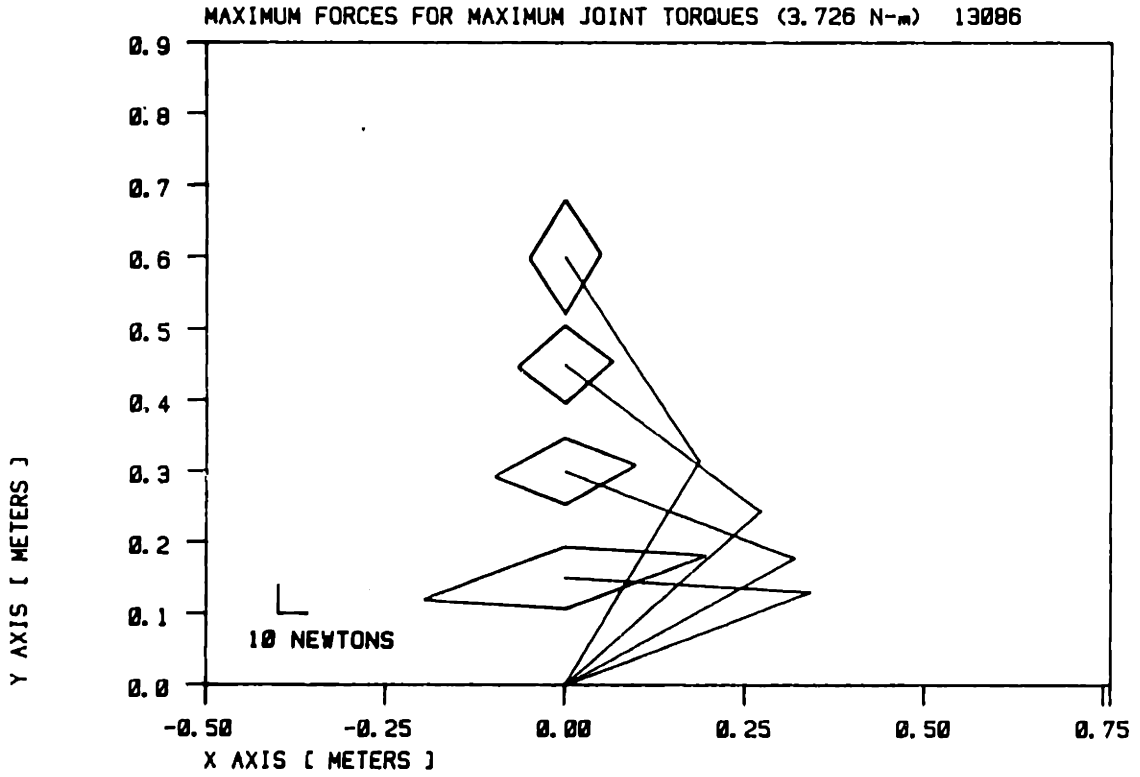


Figure 2.6 Maximum Endpoint Forces for Constant X

All the rest of the points in the workspace can be evaluated simply by rotating this straight line about the shoulder. A consequence of this

is that two of the four directions of maximum force generation form a straight line through the "shoulder" (axis of rotation of the inner link) of the apparatus. This is only true, however, when the link lengths of the manipulator are nearly equal, as they are in this case.

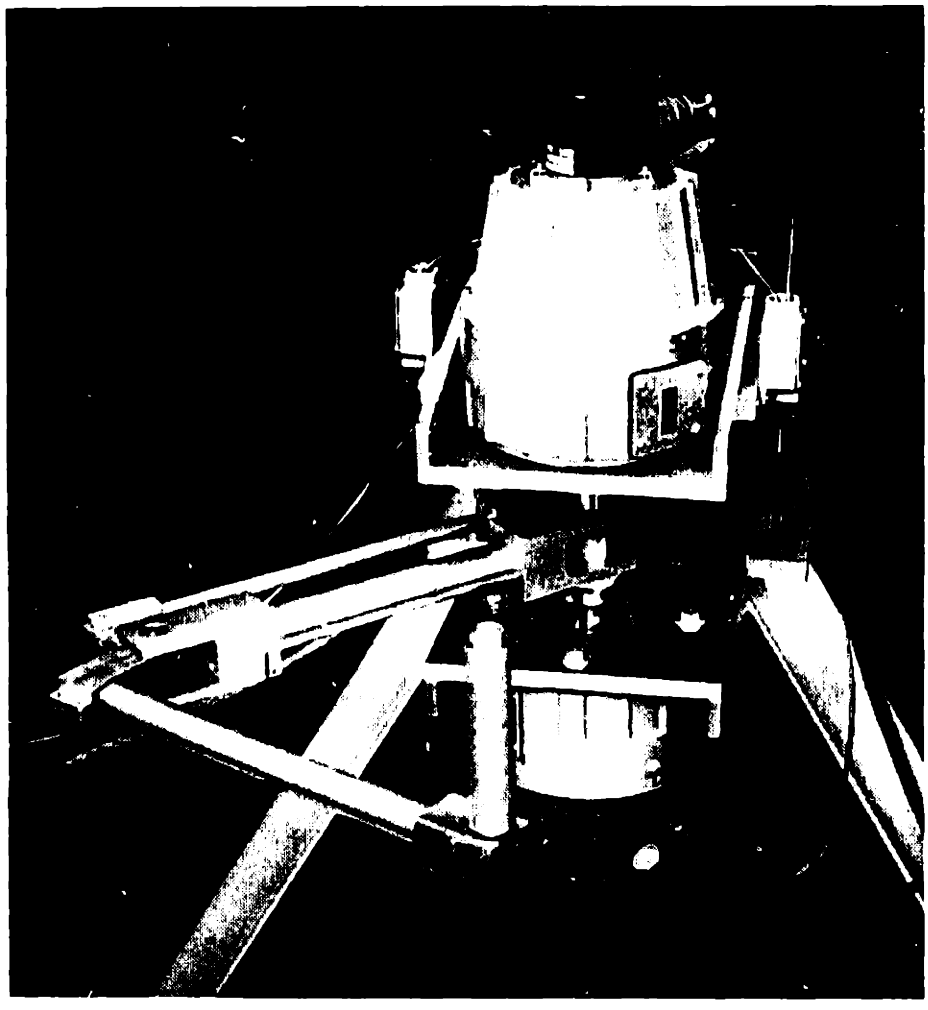
This process has been automated so that plots can be made of the maximum force generated in any direction, for a given linkage orientation. A description of this procedure and the FORTRAN program to do the computations and plotting are in appendix F. The physical hardware (i.e., torque motors, servo amplifiers, mechanical linkage, etc.) that will be capable of producing these forces and applying the control algorithm is described in the next chapter.

CHAPTER 3: MECHANICAL HARDWARE AND DESIGN CONSIDERATIONS

This chapter presents the mechanical hardware used in this work. This chapter includes a description of the manipulandum and the torque motors and servo amplifiers used to drive it. This entire system is designed to be used in experiments with human subjects, therefore it is imperative that every consideration be made for the safety of the subject and the experimenters. The safety considerations in this thesis are addressed in detail in section 3.3 on safety precautions.

3.1 Apparatus Description

The apparatus, shown in Figure 3.1, is a two link serial manipulator (similar to a SCARA robot) that is constrained to motion in the horizontal plane. For all control and discussion purposes the orientation of the links is defined in Figure 3.1. The orientation of the inner link is defined by the absolute angle theta 1 and the orientation of the outer link is defined by the absolute angle theta 2. The relative angle between the links is defined as the difference of the two absolute angles, $\theta_2 - \theta_1$. The links are driven by two DC torque motors that have output shafts that are both co-axial with the axis of rotation of the inner link. Thus one motor is mounted above the other motor so that the output shafts of each motor face each other. These motors are PMI JR16M4CH D.C. Torque motors. Some of the relevant Torque motor characteristics are listed in Table 3.1.



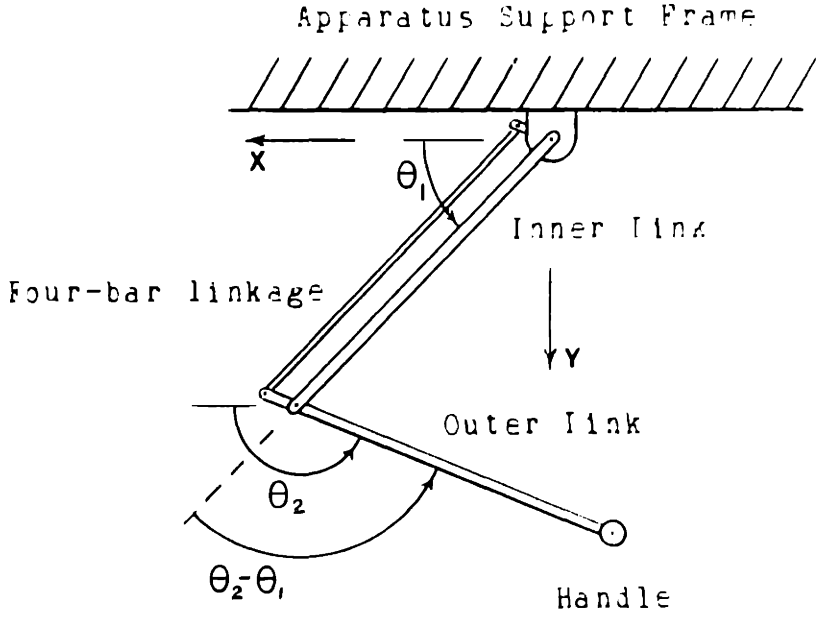


Figure 3.1 Two Link Manipulandum

TABLE 3-1 Torque Motor Characteristics

Description	Units	Value
Manufacturer and Type	PMI	Jk16M4CH
Peak Torque	(N/m)	37.48
Continuous Stall Torque	(N/m)	3.52
Peak Current	(Amps)	100.8
Continuous Stall Current	(Amps)	9.65
Peak Acceleration Without Load	Krad/sec/sec)	63.2
Torque Constant	(N-m/Amp)	0.373
Armature Resistance	(Ohms)	0.74
Viscous Damping Constant	(N-m/KRPM)	0.0644
Moment of Inertia	(N-m-sec-sec)	0.000593
Mechanical Time Constant Without Load	(mSec)	3.15
Motor Weight	(LBS)	17.5

TABLE 3-2 Servo Amplifier Characteristics

Switching Frequency	(hertz)	5000
Bandwidth	(hertz)	1000
Current Monitor Sensitivity	(Amps/volt)	3
Peak Output Current	(Amps)	30
Continuous Output Current	(Amps)	10
Peak Current Time Constant	(sec)	0.5

Channel	Open Loop Gains
1	1.29 < Gain 1 < 10.42 Amps/volt
2	0.62 < Gain 2 < 5.32 Amps/volt
3	0.33 < Gain 3 < 2.30 Amps/volt

Nominal Amplifier Setting 3.0 Amps/volt (channel 2)

The inner link of the apparatus is attached directly to the output shaft of the lower torque motor, but the outer link is driven through a four-bar-linkage, shown in Figure 3.2, that is attached to the output shaft of the upper motor.

The enitre system (motors and manipulator links) are mounted on an aluminum framework, shown in Figure 3.3, that will be refered to as the support frame. The system is mounted on this frame so that the manipulator can be used with a seated subject. It was not possible to mount the system on any wall in the designated workspace because none of the walls were structurally strong enough to support the torque motors and the rest of the system.

Both motors are mounted on an rigid support that can be considered an inertial reference frame. That is, neither motor mount is actually moving as would be the case if the outer link motor was mounted on the end of the inner link. It is common practice with industrial manipulators to mount the torque motors at the junctions of the links. Thus the torque is applied directly to the outer link. This is not practical in this application due to the size and weight of the D.C. torque motors. There is no problem with the bottom motor transmitting torque directly to the inner link, but the top motor must transmit torque to the outer link through a drive mechanism.

A number of drive mechanisms to transmit the torque from the upper motor to the outer link were considered. The manipulator has a range of motion that is similar to, but greater than that of the human arm. The

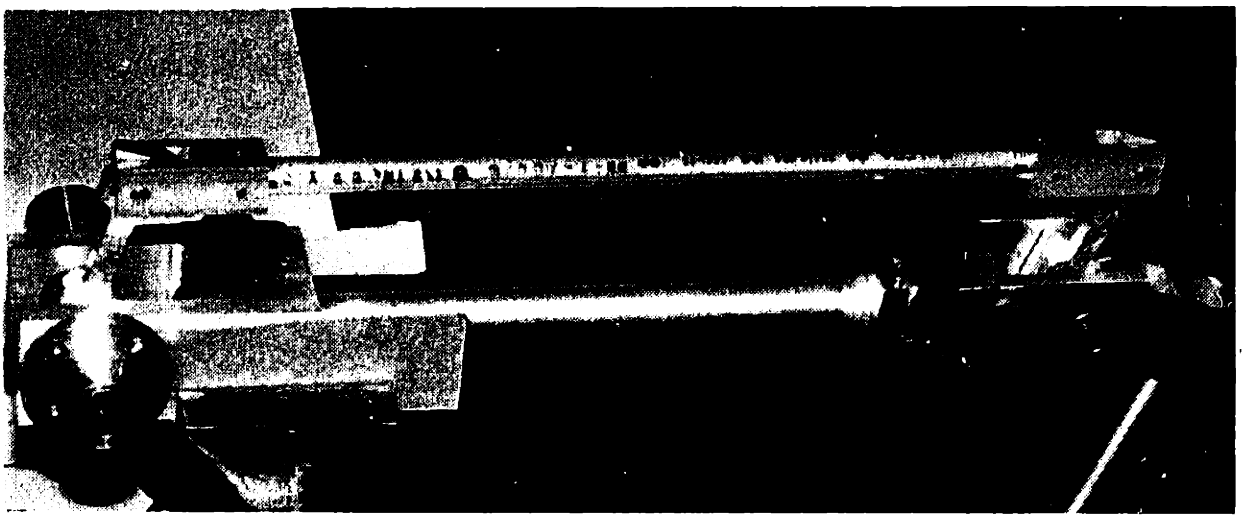


Figure 3.2 Photograph of the Four-bar Linkage

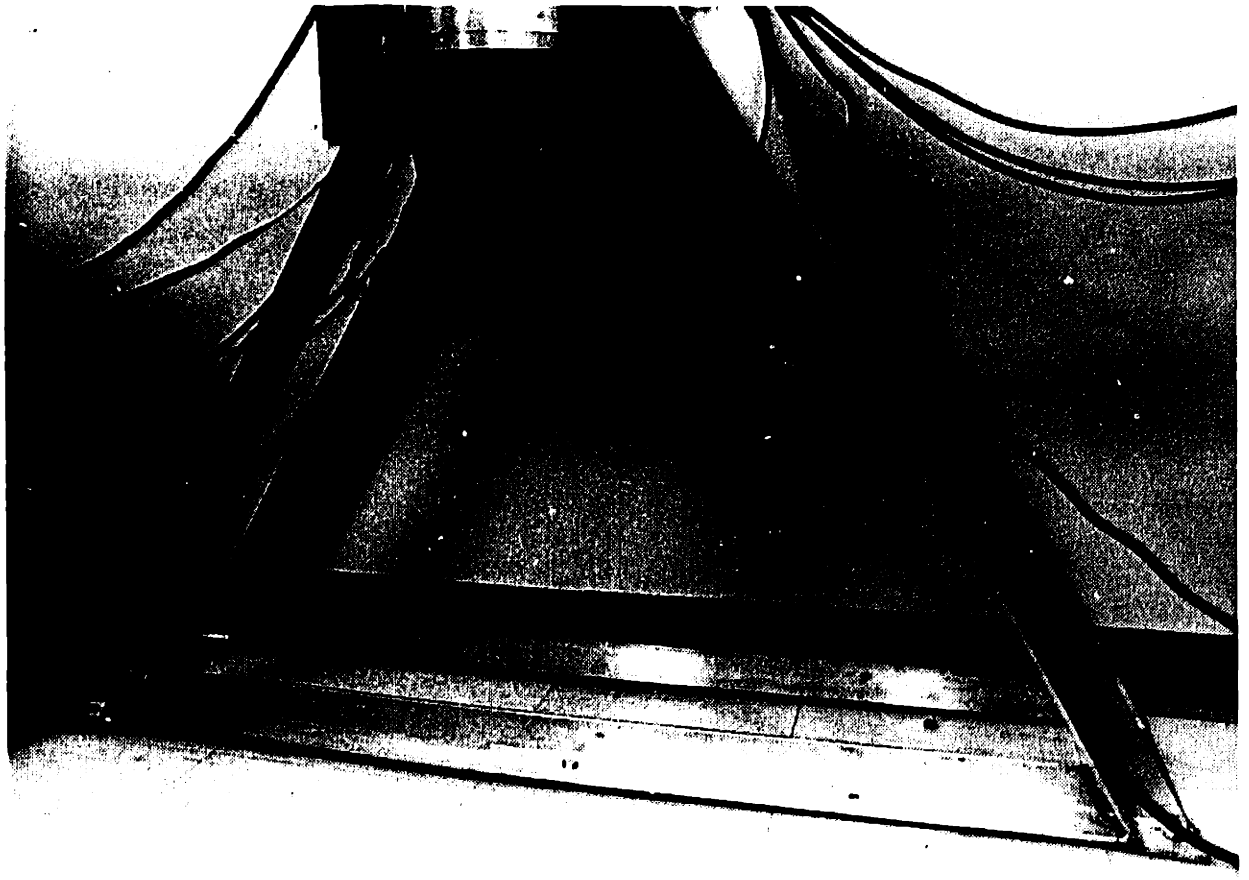


Figure 3.3 Photograph of the Apparatus Support Frame

manipulator endpoint has the ability to move out in the workspace so that the relative angle between inner and outer links is -43 degrees. And the manipulator endpoint has the ability to move into the axis of rotation for the torque motors, where the relative angle between inner and outer links is 175 degrees. This range of motion (spanning 218 degrees) is made possible by using an offset in the elbow of the manipulator, see Figure 3.4 below.

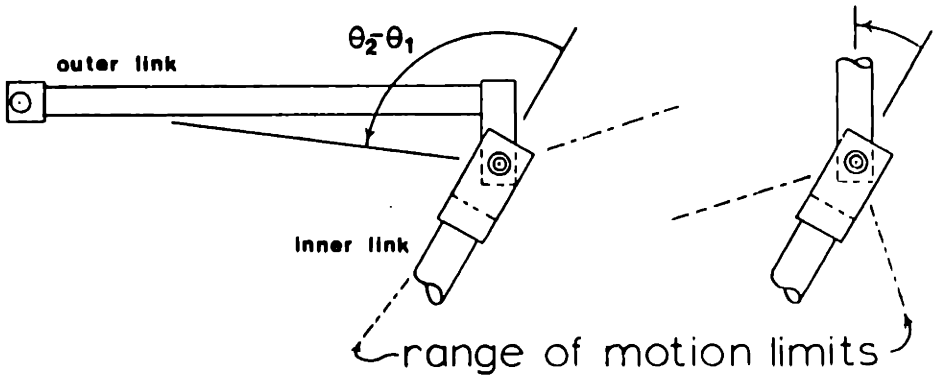


Figure 3.4 Range of Motion for Offset and Straight Elbows

The apparatus was originally constructed with a ladder chain drive, but this was replaced with a four-bar-linkage to avoid problems with compliance that are characteristic of a flexible chain drive. While the four-bar-linkage improves the ability to position the endpoint, its major drawback is that it limits the range of motion of the manipulator. The minimum relative angle between the inner and outer links is now limited to 39 degrees and the maximum relative angle is 139 degrees. Thus the addition of the four-bar linkage reduced the range of motion to less than half the possible range of motion. Although this reduction may appear significant, the reduced workspace is sufficient for experiments currently performed human subjects.

Servo Amplifiers

The motors are driven by PMI SSA 40-10-30 pulse width modulated switching servo amplifiers. The servo amplifiers are supplied with "inductive chokes" and the inductance of the motors is low enough, so that the servo amplifiers are considered to be current sources to the motors. The switching frequency of the servo amplifiers is 5000 hertz and the bandwidth is specified by the manufacturer to be 1000 Hertz. It is therefore reasonable to assume that the amplifier dynamics are sufficiently fast so that the amplifiers can be considered transconductance gains. The complete system of servo amplifier and torque motor shown in Figure 3.5, provides a torque that is proportional to the input voltage to the servo amplifier.

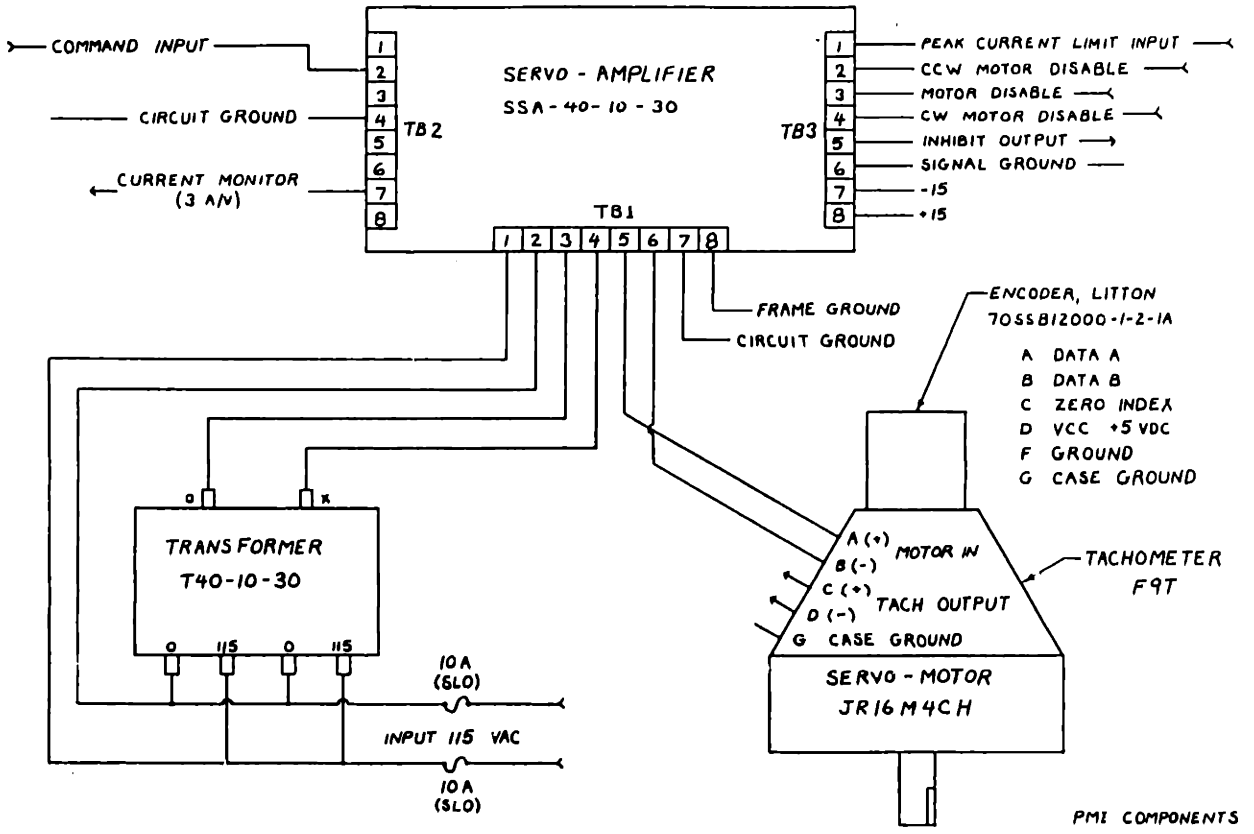


Figure 3.5 Servo Amlifier and Torque Motor System

The servo amplifiers are rated for both continuous and peak current output. In this case the continuous current level (10 amps) is a third of the peak current output, 30 amps. The servo amplifiers have built in safety features that automatically limit the current output to the continuous current level if the continuous current level is exceeded long enough to build up charge on the capacitor that checks the duty cycle of the servo amplifier. These safety features of the servo amplifiers are installed to prevent the amplifiers from overheating the motors. Unfortunately these safety features are useful in protecting the motors, but they complicate the limitations on the amount of torque generated at each joint. The servo amplifiers are designed to provide peak current (and hence peak torque) to the motors for only half a second. After 0.5 seconds of peak current, there is a rapid decay (approximately 0.5 seconds) to the continuous current output. This means that if a sustained force (longer than .5 sec) is desired at the handle, then the magnitude of the force is limited by the continuous force output of the system.

Besides the current limiting capabilities of the servo amplifiers, there was another complication uncovered in testing the servo amplifiers. The servo amplifiers were unable to maintain a constant current output for current levels that are greater than about twice the continuous current limit as demonstrated by the following experiment.

The digital computer was used to generate a command voltage to the servo amplifier and to measure the servo amplifier current monitor. The current monitor is an output from the servo amplifier whose voltage is

proportional to the actual current output. The sampling rate of the test program was 2000 hertz. The current monitor signal was found to contain components of the servo amplifier switching frequency. To avoid aliasing problems with the current monitor signal, this signal was low pass filtered before being input into the computer. The low pass filter consisted of a passive RC filter that had a cutoff frequency (-3 Db) of 500 hertz.

The peak current output of the servo amplifiers is 30 amps. It was found that, even if the servo amplifiers were commanded to output only 27 amps, then the current monitor would indicate that actual output current was oscillating at 120 hertz about 25 amps with a peak-to-peak value of approximately 2 amps. These tests were performed with the servo amplifier output connected to a pure resistive load so that any nonlinearities of the torque motor load would not be present. Figure 3.6 shows the results for various command voltages. These plots show both the command voltage to the servo amplifier and the current monitor output. The gain on the current monitor is approximately 3 amps/volt and the servo amplifier gain is nominally set at 3 amps/volt.
These voltages range from 1 to 9 volts at 1 volt increments. Clearly the system cannot be operated at or above 9 volts. The first indications of this oscillation start at about 8 volts. For this reason, the command output from the computer is always limited to a maximum of +8 volts. This oscillation was seen on both amplifiers and in both directions (positive and negative outputs). It is possible that this ripple is due to the line voltage ripple (120 hertz) on the power supply to the amplifier. This problem was beyond the scope of this work, and its

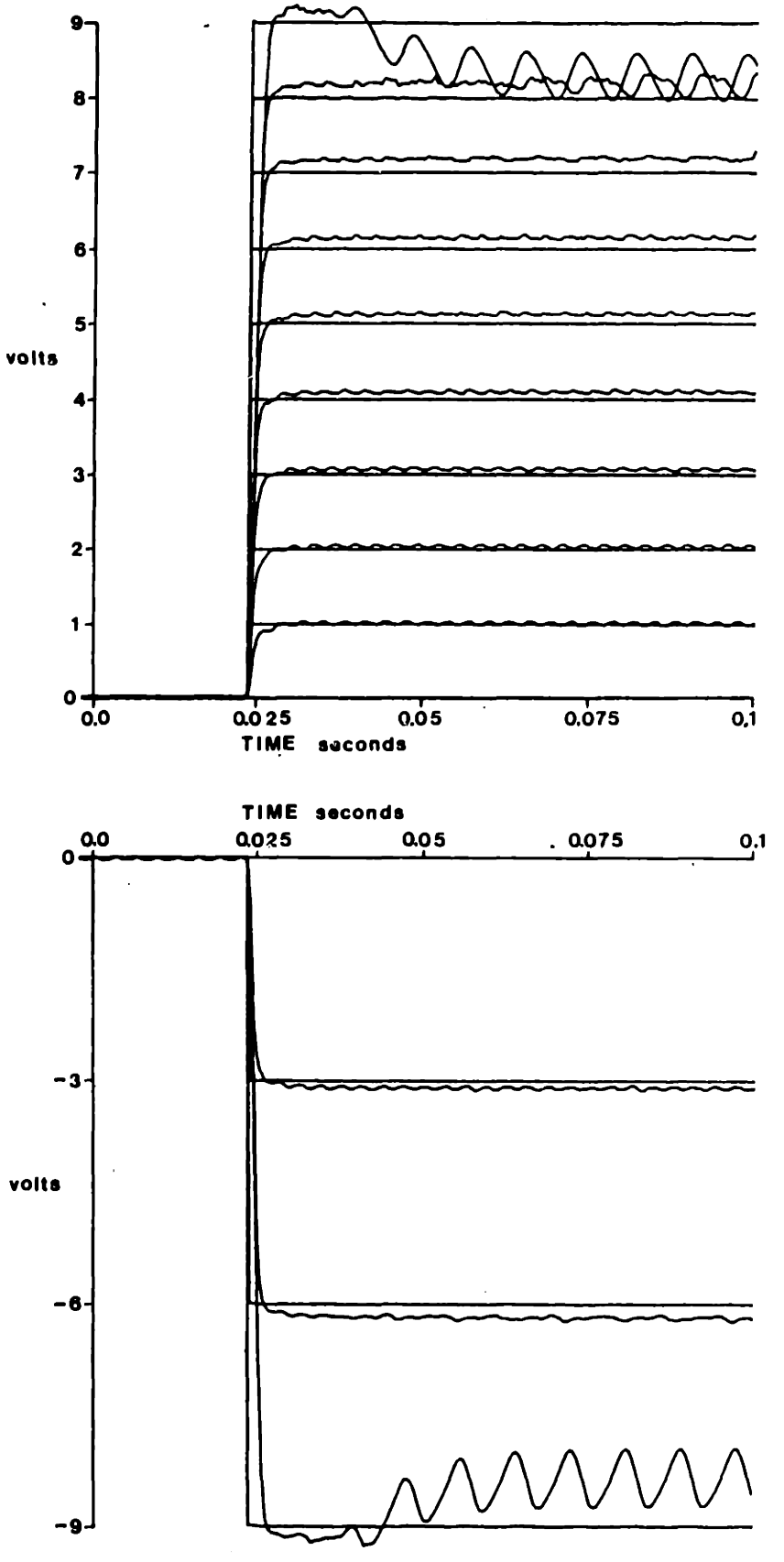


Figure 3.6 Servo Amplifier Step Response

solution is left for further work.

As per a recommendation from PMI, the compensation circuit on the servo amplifiers was disabled. This was done by removing 2 resistors from each servo amplifier (R59 and R62; refer to servo amplifier blue print). This compensation circuit acts to limit the bandwidth of the servo amplifiers. By disabling this circuit the servo amplifiers are more sensitive to noise, but this reduces the open loop gains of the amplifiers. Table 3.2 shows the new open loop gains. This change was made so that the range of inputs to the servo amplifiers would be compatible with the output of the computer's D/A converters. The output range of the D/A converters used in this work is $\pm$ 10 volts. A gain setting of 3 amps/volt (pin 2) on the servo amplifier will yield the maximum current for the maximum output of the D/A converters.

3.3 Safety Aspects

Because the primary use of this apparatus is with human subjects, the safety of the apparatus has been a primary concern of much of this work. The servo system must be completely safe and must in no way injure either the subject or the experimenter. The primary design philosophy behind much of the safety features is to remove power from the motors if anything goes wrong and to remove stored energy as quickly as possible in the absence of power inputs. For an unacceptable condition (for example, the manipulator moves outside allowable workspace) the amplifiers will be disconnected from the motors and the motor terminals will be short circuited.

In the event of problems arising with the manipulator or torque motor system, the servo amplifiers are completely disconnected from the motors so that there is no way that the servo amplifiers can possibly drive the motors. The servo amplifiers should never be run without a load, therefore the same circuit that switches the amplifiers to and from the motors, is used to control a set of relays that will disable the servo amp when the motors are disconnected. The motor terminals are short circuited to provide dynamic braking.

3.3.1 Mercury Displacement Relays

The continuous current output to the motors is 10 amps at 40 volts DC. The maximum current output to the motors is 30 amps. Because these currents are so high it was necessary to use relays that could handle at least 10 amps at 40 Volts DC. The relays that were best suited for this task were Mercury Displacement Power Relays made by Magnecraft. These relays operate by having an energized coil pull a cylinder down into a pool of mercury. The mercury then rises and completes the circuit. In this way, each time a contact is completed there is essentially a new set of points. The gaseous portion of the container holding the mercury is filled with an inert gas, thus oxidation cannot occur.

In the unenergized (or passive) state, there are two types of mercury displacement relays available. Those that are normally closed (used to short the motor terminals) and those that are normally open (used to open the circuit between the motor and the servo-amplifier). This system has 2 normally closed relays, (2) WM35B-120A, (one for each

motor), but 3 normally open relays, (1) WM35AAA-120A. Two of the normally open relays are used for the connections between the motors and the servo amps and the third relay is used as a "latching relay". That is, when the coil for this relay is momentarily energized (by the momentary switch in Figure 3.7), the now energized relay becomes the element that keeps the coil energizing circuit complete. If this circuit is broken at any time, the latching relay will revert to its normally open state and the energizing circuit will be open. At this point, the user must close the momentary switch in order to reactivate the coil energizing circuit.

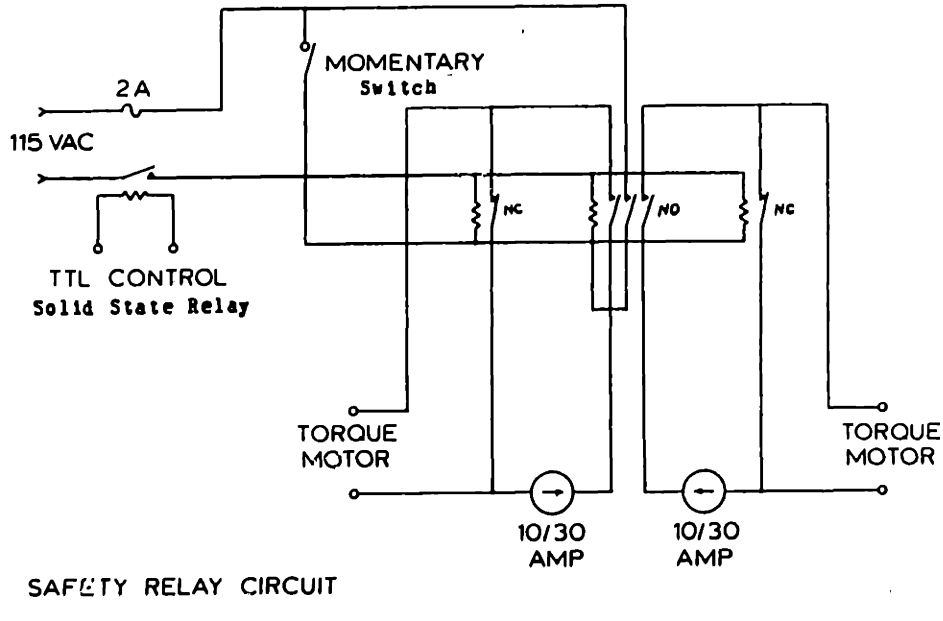


FIGURE 3.7

3.3.2 Relay Control Circuitry

The goal of the relay control circuitry is to monitor several states of the apparatus and enable or disable the ability to energize

the power relays. The states are comprised of the linkage orientation and the status of any control program (that is, is a program running?). The linkage orientation is monitored by physical limit switches on the apparatus and by specific bits in the parallel output from the optical encoder decoding circuitry (described in chapter 4) representing angular position. The combination of limit switches and encoder counter outputs define a workspace outside of which the power relays cannot be energized and hence the motors are unpowered. This workspace is shown in Figure 3.8 (the straight line in this figure is due to the physical size of the target board under which the manipulator operates). To enable the ability to energize the power relays a program must be running and it must set the proper bit on a particular parallel interface line. The details of the relay control circuitry are discussed in this section, but the implementation of the limit switches will be discussed in the next section.

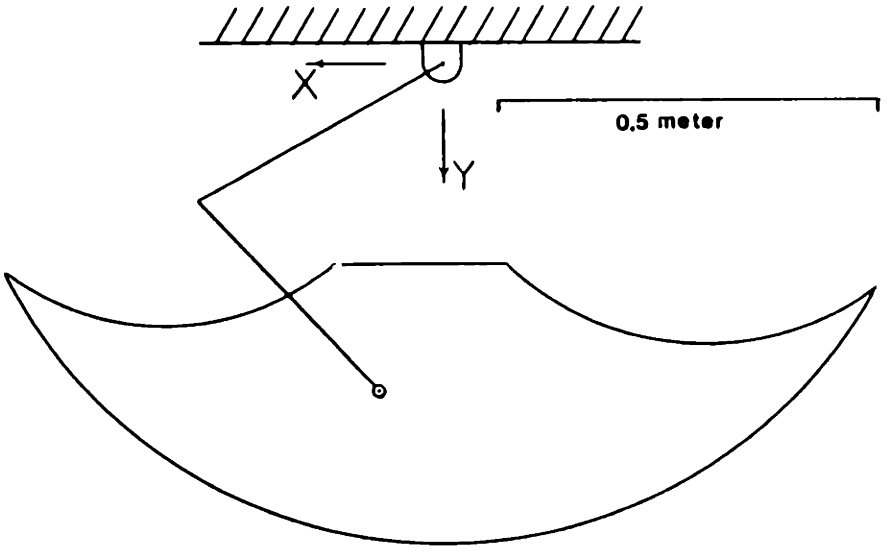


Figure 3.8 Practical Workspace of the Manipulandum

The fail-safe aspect of this setup is that it requires the experimenter to physically activate the coils to energize the relays. This is done with the momentary switch in the energizing coil circuit. It is the energizing coil circuit that is the key component here. Any condition that breaks this circuit will shut the entire system down. Because this circuit is energized by 120 VAC, it is necessry to keep the entire circuit shielded from the experimenters and the subjects. Therefore a solid state relay is used to control the energizing coil circuit, see Figure 3.7. Unless the conditions are proper for this relay to be energized, then the mercury displacement relays can not be energized no matter how long the experimenter pushes on the momentary switch.

The solid state relay in the coil circuit is a Grayhill solid state relay 70S2-04-B-02-H. This solid state relay is capable of handling 2.5 amps at 120 VAC and has a control range of 3 - 30 VDC. Thus it possible to put this relay directly into the energizing coil circuit with with all of the mercury displacement relays. In this way, a single solid state relay is controlling the full set of mercury displacement relays. The energizing coil circuit is only complete when the solid state relay is energized by a 5 volt signal from the TTL (Transistor Transistor Logic) relay control circuitry, Figure 3.9.

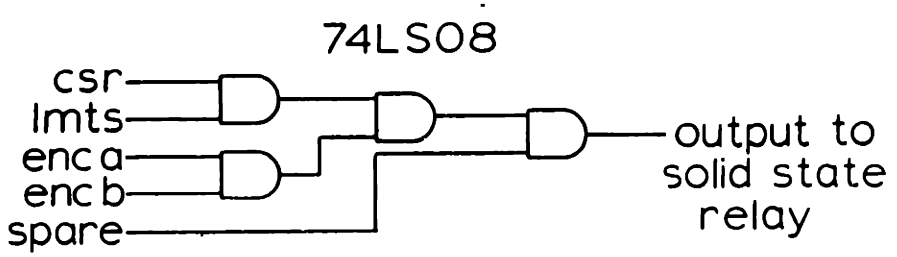


Figure 3.9 Relay Control Circuitry

The TTL control circuitry consists primarily of a single Quad AND gate chip, 74LS09. This chip is wired so that 5 inputs control one output. All of these inputs must be high (5volts) for the output to be high and the solid state relay to remain energized. Two inputs come from the encoder counter information, one input comes from the limit switches on the hardware itself and one input comes from a control bit (the CSR bit) on the parallel interface that supplies the offset angle to the D/A angular position card for theta 1. The offset angle and D/A angular position cards are described in detail later in section 4.3.2. The final input is temporarily wired high as a spare for later application.

The input from the D/A angular position card for the outer link angle is taken from the overflow bit of the binary adder chips (described in in chapter 4). This bit is normally high until the adders output a parallel binary word that corresponds to less than 0 or greater than 180 degrees. It is necessary to use the overflow bit so that changing the angular offset (a parallel binary input word) will not only change the zero reference, but it will appropriately adjust the position at which the relays will be shut down. The input from the binary signal

for the inner angle, however, is taken from the thirteenth bit of the Encoder decoder circuit. The inner link cannot practically span a range that is any greater than 0 to 180 degrees which is the range that is allowed using the thirteenth bit of the encoder decoder circuits. Thus, there was no reason to implement the same strategy on the inner link.

The implementation of the impedance controller involves using the digital computer to generate command voltages to the servo amplifier. To insure that the only commands that the servo amplifiers respond to, are based on desired commands, it is important to have a program active when the relays are active. This reduces the possiblity of having some transient signal on the command input (output from the computer's D/A converters) to the servo amplifier when the relays are energized. The input that comes from the CSR bit (on the parallel input for the angular offset of theta 1) insures that a program must be running in order for the user to activate the relays. This is because the CSR bit is easily set high at the beginning of the program, but returns to its normal low state when the program is terminated. Also, if a condition arises during operation (i.e. excessive speeds), then the CSR bit can immediately be set low and the system will be shut down.

The final spare input is intended to be used with a retriggerable monostable multivibrator TTL 74LS122. With this chip it is possible to insure that the relays can only be turned on if the program is running at the correct speed. The output of the chip remains high as long as its input is pulsed (by the computer) every time step, dt. The duration of dt can be varied depending on the external resistor-capacitor pair

selected [24]. The larger the resistor or capacitor, the longer the delay.

Unfortunately this retriggerable monostable multivibrator cannot be implemented currently because of a problem with the momentary switch and turning on the relays. Sometimes when the relays are energized with the momentary switch, the encoder decoder counters are reset. A significant amount of time was spent trying to solve this problem, but no solution was found. It appears as though the energizing causes some random noise on the circuit ground. This in turn causes a false reset signal on the counters. If the relays were to be re-energized before each experimental run, it would be necessary to also re-calibrate the encoder counters before each run. The solution of this problem is postponed for later work.

3.3.3 Limit Switches

There are three limit switches mounted on the apparatus, Figure 3.10. Two limit switches are mounted on the support where the motors are mounted and one switch is mounted on the four-bar-linkage. The two limit switches mounted on the support act as backups for the encoder counter disable inputs. They prevent the inner link from being driven into the support behind the system. The limit switch on the four-bar-linkage shuts the motors off just before the four-bar-linkage comes in contact with the inner link.

The limit switches are simply mechanical switches and exhibit

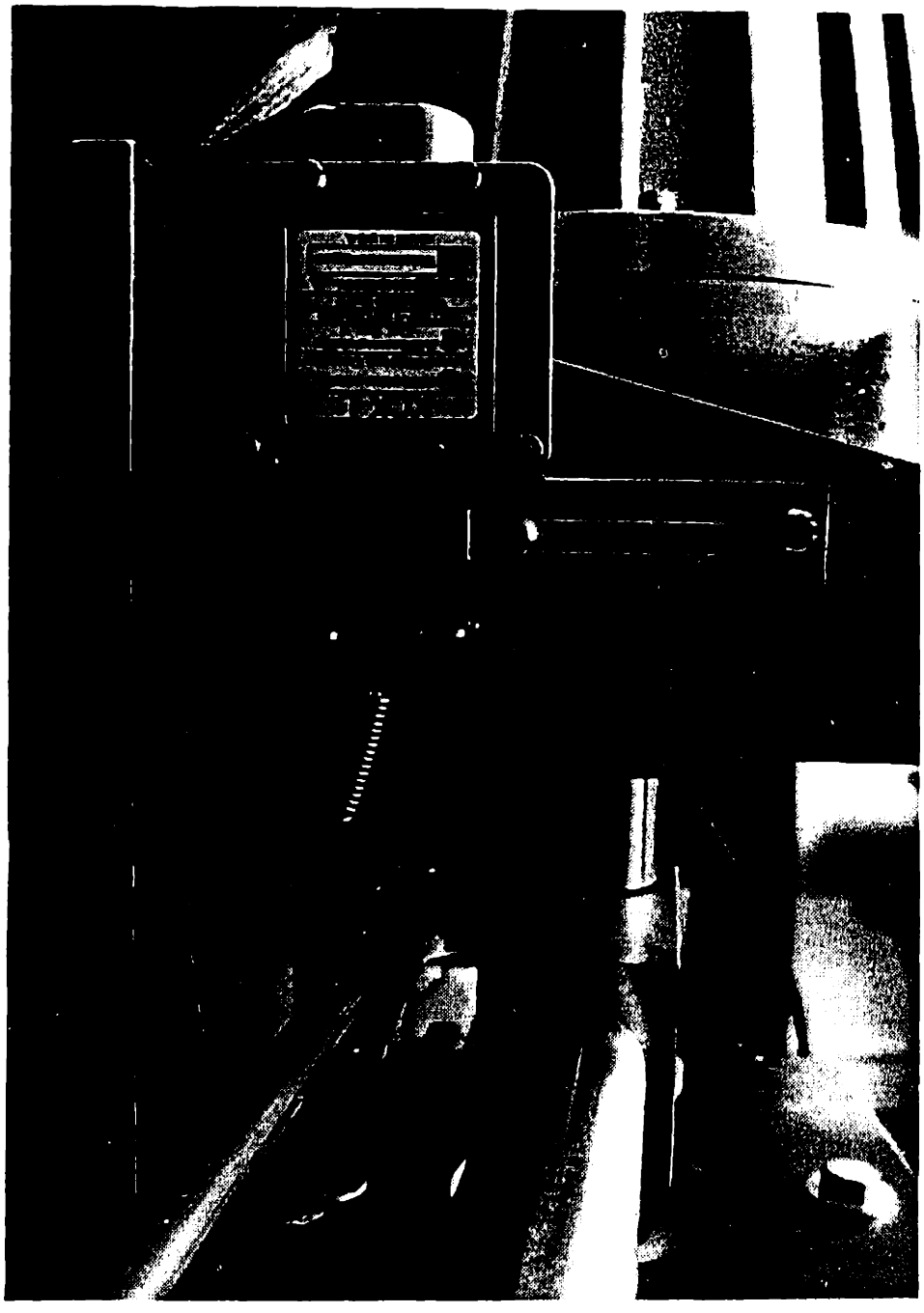


Figure 3.10 Limit Switches

mechanical switch bounce that is a common problem with such switches. To get rid of this problem it was necessary to "de-bounce" the switches with the circuit shown in Figure 3.11.

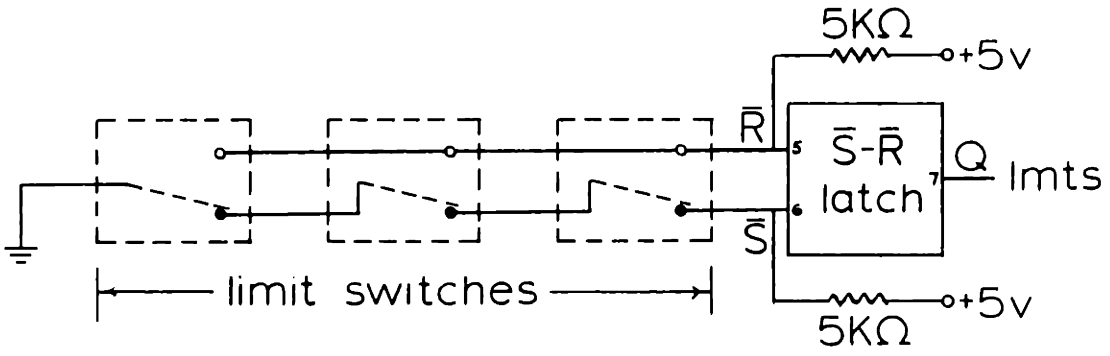


Figure 3.11 Switch De-bounce Circuitry

This circuit utilizes TTL S-R latches 74LS279 [24] and has the limit switches wired in series. If any of the limit switches changes the ground from S to R then the output of the S-R latch will go "low" and the relays will "shutdown". This is "de-bounced" because during the transition phase (switch from S to R), the inputs to the S-R latch are both "high" and in this state the S-R latch continues to hold its last output.

3.4 Relay Operation

There are a set of steps that must be followed in order to set up the system for operation. As discussed above, it is not possible to have the servo-amplifiers drive the torque motors without the relays being energized. Currently there are operations that must be performed in software and to the apparatus itself, these are:

A. CAUTION: Make Sure The Amplifiers are DISABLED.
   (silver switch DOWN)

B. Set the offset values to the Encoder D/A s to zero.

D. Set the CSR bit for Encoder D/A 01 high (1).

E. Move linkage to far left to clear encoder counters.
   (links will have smallest absolute angles)

F. Move linkage back towards the normal workspace.
   (until the limit switches click into operation)

G. Press the black switch on the relay panel.
   (An orange pilot light indicates that the coils are energized (1))

H. Move the manipulator endpoint to a calibration position.

I. Compute and Output angle offsets to Encoder D/A.

J. ENABLE amplifiers, to run (silver switch UP).

(1) If the relays do not come on, then keeping the black switch depressed and moving the arm back and forth will help set the proper state in the relay control circuitry.

CHAPTER 4: CONTROL SYSTEM HARDWARE

4.1 Sensors

To control stiffness and viscosity using feedback it is necessary to measure both position and velocity. Likewise, the modification of endpoint inertia requires force sensing. Restricting ourselves to the control of endpoint stiffness and viscosity will eliminate the need for force sensing. This chapter will first present a discussion of the sensors used in this work and then discuss in detail, the electronics associated with each sensor as well as a backplane that houses the electronics for the position sensing circuitry.

For a system that has rotary actuators (such as the D.C. torque motors used with this apparatus), several options exist for directly measuring the angular postion of the actuator output shaft. Rotary potentiometers provide an analog voltage output that corresponds directly to angular position, while the output of optical encoders is typically in a digital form that is better suited for digital control. The digital output of the optical encoder can be fed directly into the computer through the digital parallel interface. This is generally much faster than the analog to digital (A/D) converter needed to convert the analog potentiometer signal into the equivalent digital signal.

This chapter discusses the application of optical encoders to measure the absolute angle of each manipulator link. This chapter also addresses the issue of measuring the angular velocity of the manipulator

links. The absolute angular velocity is measured by integral tachometers that are built directly into the D.C. torque motors. These measurements are considered absolute in both cases because they are made with respect to a stationary (non-accelerating) reference frame.

4.2.1 Optical Encoders

Position is sensed by incremental optical encoders attached to the output shaft of each torque motor. There are basically two different types of encoders; absolute and incremental. Absolute encoders output a digital code that is distinct for a given location and are most often used in environments where power may be temporarily lost. Incremental encoders provide only a pulse train output. With incremental encoders there is no way to distinguish any of the pulses from any of the other pulses.

Incremental encoders are used here because of high resolution and reasonable cost. These encoders are Litton 70SSB12000-1-2-1A optical encoders and have 2000 lines per revolution. There are two channels of output data that each generate 2000 pulses per revolution and one channel with a single pulse per revolution. With two output channels and the proper decoding circuitry it is possible to generate 8000 counts per revolution (that is, .0007854 radians per count).

Because of physical limitations of the hardware it is not possible to set the manipulator links at the absolute reference angle of zero degrees and initialize the encoder counters. It is also extremely dif-

ficult to mount the encoders so that the zero index is aligned with what has been defined as zero degrees on the torque motor support. For this apparatus the encoders were mounted with the zero index within 20 degrees of absolute zero. The linkage is put in a known calibration orientation and the encoder decoder outputs are measured. Comparing these measurements with the known orientation of the linkage will give angular offsets for θ_1 and θ_2 (as previously defined in chapter 3, Figure 3.1). These angular offsets are used in the digital control program to compute the absolute angles of the links referenced to any orientation.

TABLE 4-1 Endpoint Location Errors Due To Encoder Counter Errors

Nominal Endpoint Position: X = 0.0, Y = 0.4492 (meters)
Nominal Joint Angles: θ_1 = 41.69, θ_2 = 142.926 (degrees)

Joint Angle Error: $\pm$ 0.045 degrees

θ_1	θ_2	X	Y	Δ X	Δ Y	MAG
41.735	142.926	-.0001942	0.4494	-.0001942	-.0002	.00028
41.645	142.926	0.0001875	0.4490	0.0001875	0.0002	.00027
41.735	142.881	-.0000322	0.4496	-.0000322	-.0004	.00040
41.645	142.881	0.0003495	0.4492	0.0003495	0.0000	.00035
41.735	142.971	-.0003561	0.4492	-.0003561	0.0000	.00036
41.645	142.971	0.0000256	0.4488	0.0000256	0.0004	.00040
41.690	142.971	-.0001652	0.4490	-.0001652	0.0002	.00026
41.690	142.881	0.0001587	0.4494	0.0001587	-.0002	.00026

Average Magnitude Error: 0.00032 meters
Standard Deviation: 0.000062 meters

The ability of the encoders to predict endpoint position was evaluated both quantitatively and empirically. For a given linkage orientation

the errors in endpoint position can be computed for errors in the the least signifcant bit of the encoder counters. Table 4-1 shows the results for all the possible combinations of errors in the encoder outputs for the linkage in the orientation; θ_1 = 41.69, θ_2 = 142.926 (degrees). From this table it is possible to see that the error in endpoint position can be as much as 0.4 mm but tends to average about 0.32 mm.

The physical hardware was tested by moving the endpoint along a straight line in the workspace and measuring the joint angles. Figure 4.1 shows the actual orientation and range of this test.

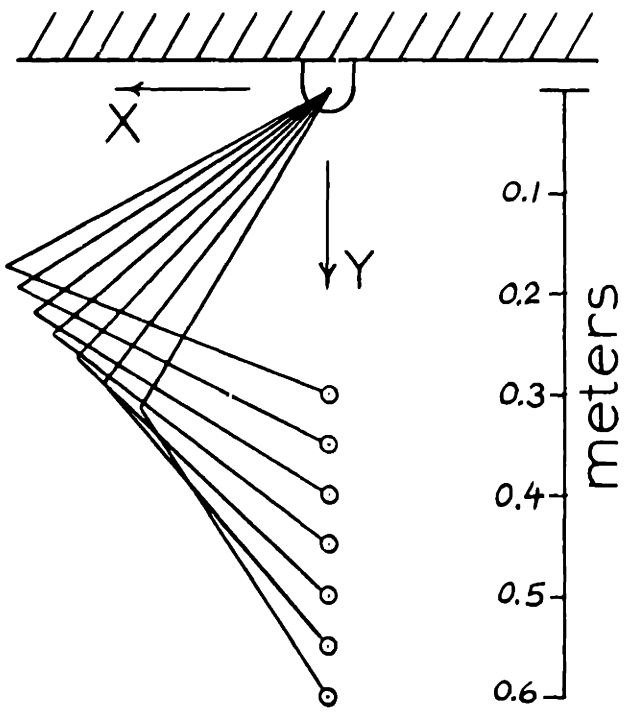


Figure 4.1 Test Orientations for Encoder Verification

Because the manipulator link lengths are nearly equal (ℓ_1 = .3654 m, ℓ_2 = .342 m, $\Delta\ell$ = .0234 m) a plot of the inner link angle against the outer

link angle should be nearly linear for the range over which this experiment was performed. Figure 4.2 shows that this is the case.

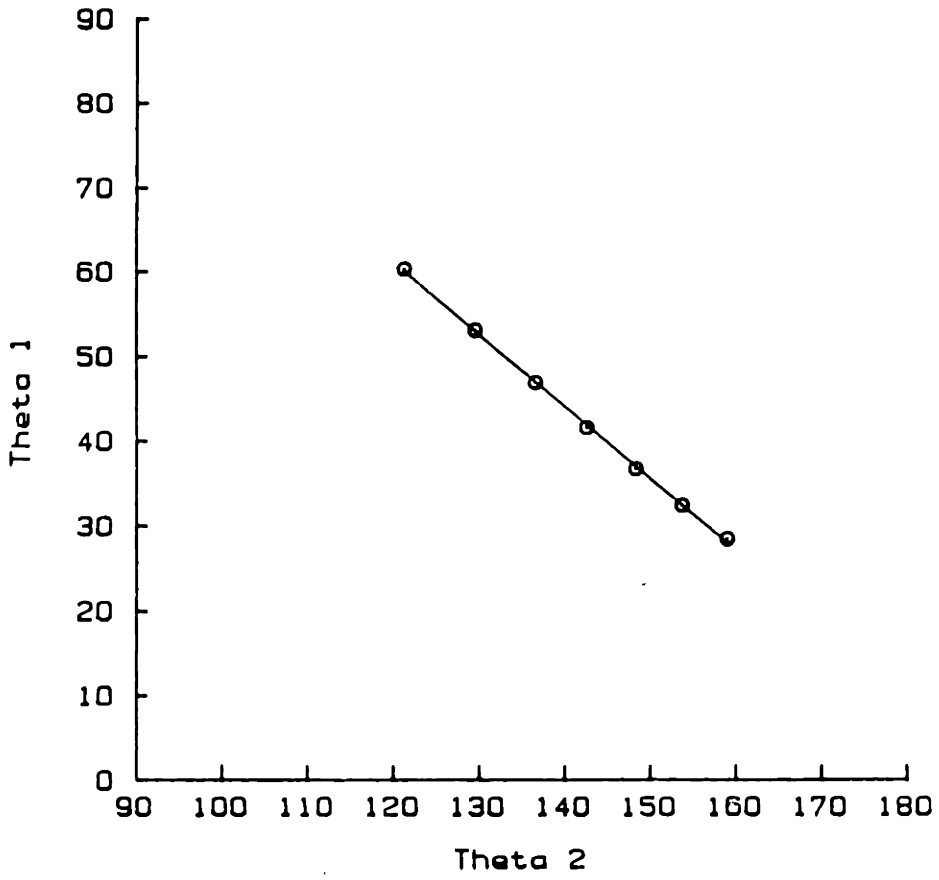


Figure 4.2 Cross Plot of Absolute Angles for Encoder Verification

If the link lengths were exactly equal, then the plot would be exactly straight. For this test the correlation coefficient between the data points is 0.9997. It is reasonable to assume that the optical encoders will serve as an accurate measure of joint angle.

4.2.2 Tachometers

While it is certainly possible to measure only the position and

then differentiate it to get velocity, this is not appealing for several reasons.

In practice, analog differentiation is limited by the range over which differentiation will occur. This constraint arises because it is not possible to implement a pure differentiator. At some point (for increasing frequency) the differentiator will stop differentiating and simply act as a gain. Even a decade before the differentiation stops, there is phase lag in the output of the circuit that leads to degradation of the differentiator performance. As the frequency increases, the differentiator will eventually roll-off and effectively become an integrator. This can all be summed up by equation 4-1 below. In this equation, C is the design cutoff frequency (in RAD/SEC), or effectively the frequency at which differentiation is no longer occuring. And D is the frequency at which integration begins.

$$\frac{V_{out}}{V_{in}} = \frac{CDs}{(s + C)(s + D)} \qquad [4\text{-}1]$$

Another inherent problem with differentiation is that by increasing the cutoff frequency more noise is not only being passed through the filter, but amplified as well.

It is possible to estimate the velocity using digital differentiation techniques. The most common approximation is:

$$V_d = \frac{(x_i - x_{i-1})}{T_s} \qquad [4\text{-}2]$$

For this estimate to work properly the choice of T_s is critical. If T_s

is too large, then the digital implementation of an analog controller will not have adequate bandwidth. If T_s is too small, then the gain on the velocity will be increased and the computed velocity will be more sensitive to errors in measuring positions x_i and x_{i-1}. Another problem arises due to the ability to measure the position. If the amplitude of $(x_i - x_{i-1})$ is less than the quantization level of x for the sampling period T_s, then the relation 4-2 will generate an error on its estimate of the velocity.

Measuring the velocity directly is a alternative to digital or analog differentiation. The tachometers used in this project have a maximum velocity of 4000 RPM or 418.8 rad/sec, which is greater than the maximum unloaded speed of the torque motor, 317.5 rad/sec. The manufacturer of these tachometers suggests that the tachometer signals be low pass filtered at a cutoff frequency of 1000 hertz to reduce tachometer ripple. The tachometer signal in this work is low pass filtered before being input to the computer, but for reasons described below, in the circuitry section, the cutoff is significantly lower than 1000 hertz.

Unfortunately these tachometers were designed to be run at high speeds and have correspondingly low output voltage at low speeds. They were used in this work because they are built in the motor housing and it was not practical to change them. Rather than modify the tachometers themselves, Analog Devices AD524 instrumentation amplifiers were used to amplify the output signals.

To verify the tachometer sensitivity it was necessary to calibrate

the actual tachometers. This not only served to give an accurate estimation of the tachometer sensitivity, but it was done to incorporate the amplifier and the low pass filter into the calibration. The calibration was done by measuring the angular position (from the encoders) and measuring the output of the tachometers. The output of the tachometers was then integrated by computing the area under the curve with a rectangular approximation:

$$A_i = \Sigma_{j=0}^{i} V_j dt \qquad [4-3]$$

This is represented graphically in Figure 4.3.

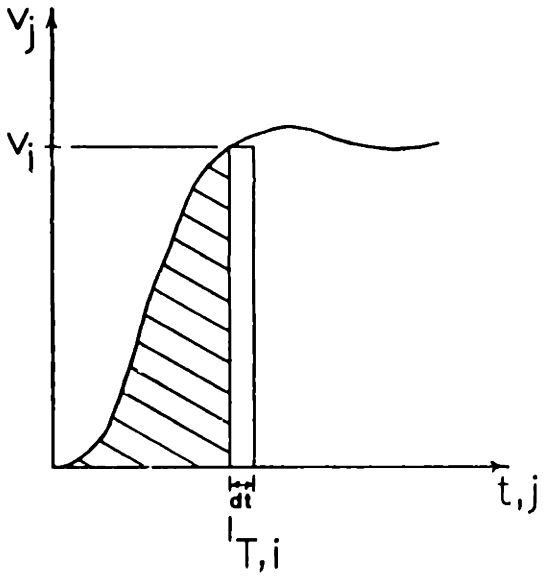


Figure 4.3 Integration Using Rectangular Approximation

Ai represents the integrated tachometer signal at the time T = idt. This integrated tachometer signal is related to the position, at time T, by a constant scaling factor. This calibration coefficient is computed by dividing the area under the velocity curve, at time T, by the position measured at that time. This procedure was repeated several times and the results were averaged to give the calibration factor listed in

table 4-2. Figure 4.4 shows a typical calibration run. The velocity signal starts off at zero reaches a peak and then drops back to nearly zero. The position signal simply shows the transition from one location to another. A curve for the calibration factor is generated by plotting the computed calibration factor for each time T. This curve shows some transient behavior at the beginning of the movement, but then settles to a constant value for the duration of the movement. This transient is most likely due to the numerical integration scheme given in equation 4-3 above. For this reason, only the constant value was considered to be representative of the actual sensitivity of the tachometers.

TABLE 4-2 Sensor Calibration Factors

Sensor	Units	Value
Optical Encoder	(rad/count)	0.000393
Tachometer	(volts-sec/rad)	0.312
Force Transducer	(volts/N)	0.17

Another approach to calibrating the tachometers would be to attach a constant velocity source to the tachometer shaft, spin the tachometer shaft at this speed and measure the tachometer output voltage. This could be done at several speeds and the results averaged together to give a calibration constant. This approach, however, would not work in this project. With the manipulator links attached to the motor output shaft, the tachometers are not free to spin continuously. Every time a calibration needed to be repeated, the manipulator links would have to be removed and a reliable velocity source attached to the motor shaft. The numerical integration calibration approach does not require that motors spin continuously and is easily repeated whenever there are

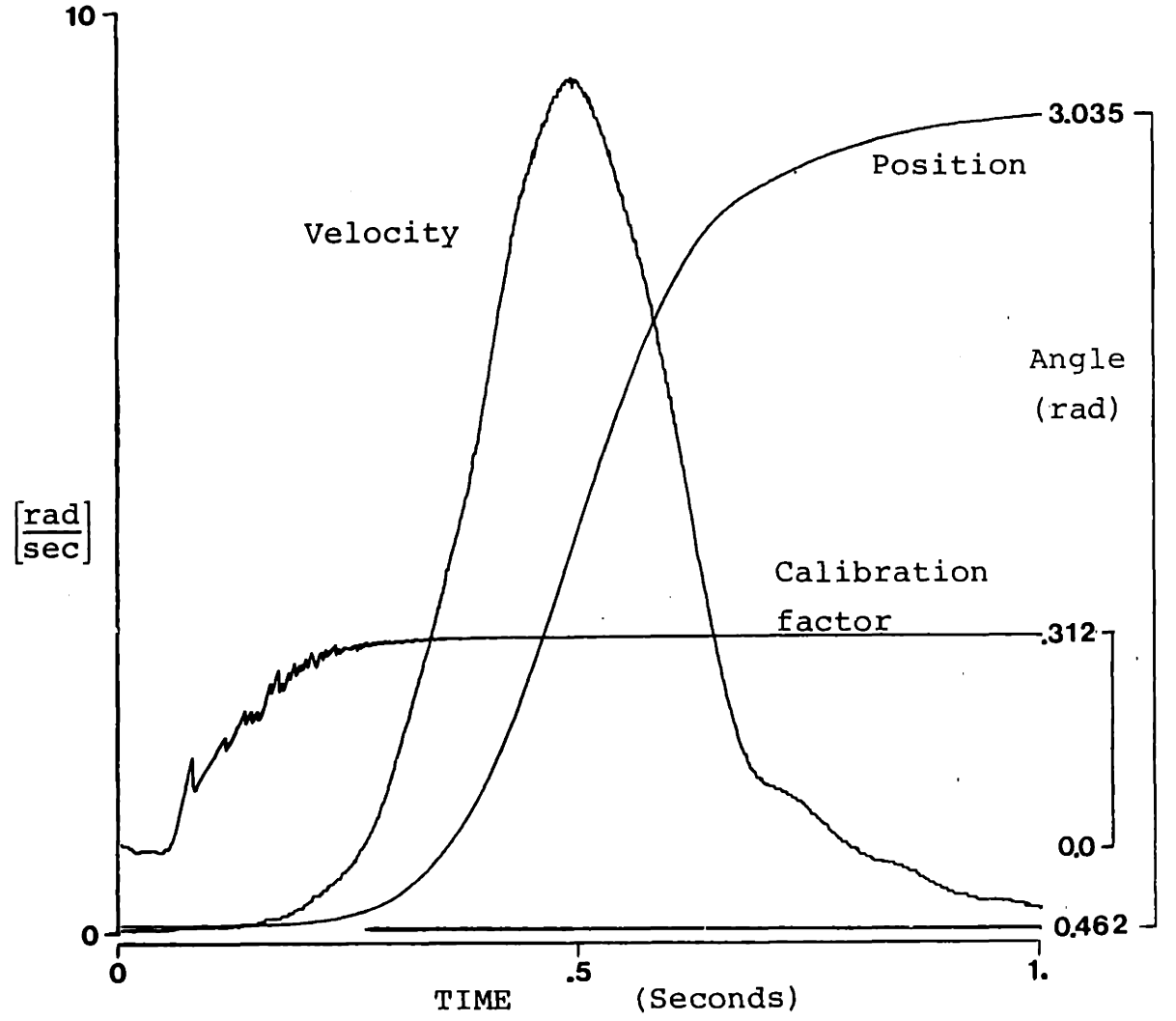


Figure 4.4 Tachometer Calibration Plot

changes made in the amplifier or filter circuits.

4.3 Circuitry

This section describes circuitry that is necessary to decode the optical encoder signal and the circuitry that is needed to amplify and filter the tachometer signal. There is also a description of some circuitry that converts the encoder-decoder output into an analog signal. Finally the backplane that holds the encoder-decoder and D/A converter is described. To minimize noise effects, the tachometer amplifier and filter were put as physically close to the tachometers as possible.

4.3.1 Encoder-decoder

These optical encoders are basically comprised of a disk that has 2000 slits etched into it. On one side of the disk is a light source and on the other side is a light sensitve open collector transistor. The slits in the disk act as windows and allow light to pass through the disk only when the window is in line with the light source and the light detector. When the detector is occluded, then its output signal is low. However, when the light hits the detector, the signal from the detector changes from low to high. It is the motion of the rotor windows and the subsequent high/low transitions of the detector output that generate an output pulse train.

According to Johnson [15] unclocked decoding circuitry can be a problem with incremental encoders. Vibration in the rotor windows can

cause the encoder counter to "run away". Encoder counter run-away is a condition that can exist when input oscillations of the encoder shaft (to which the rotor window is attached) are restricted to one encoder slit. When this condition exists, a false pulse train to the counter will cause the counter to count up or down without the appropriate real motion of the mechanical system. Johnson's paper presents a decoder circuit that is supposed to prevent this phenomenon, but work by other graduate students in the Man-machine lab, at M.I.T., found that this circuit will not work. This is because the signal is never properly latched. The decoder circuit that was used in this work is shown in Figure 4.5. It is comprised of analog amplifiers (to scale the signal up), quadrature and direction sensing logic [5], counters, and latches. The quadrature and direction sensing logic is shown in Figure 4.6a with the appropriate timing diagram 4.6b.

The outputs of these encoders are two quasi-sine waves, see Figure 4.7. below, and a single channel representing the zero index.

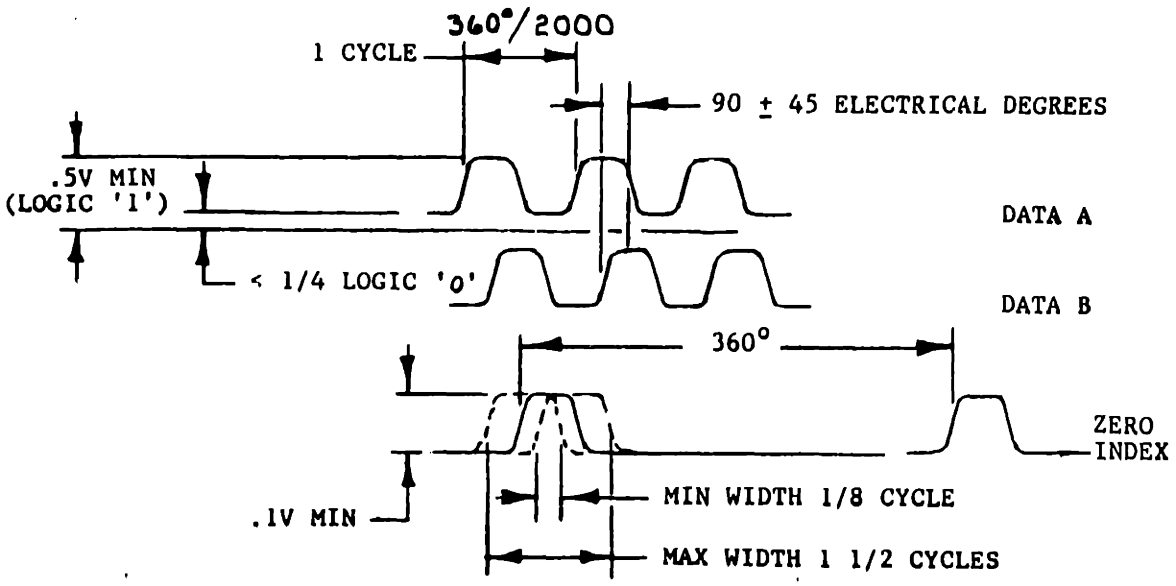


Figure 4.7 Raw Encoder Outputs, Quasi-sine Waves

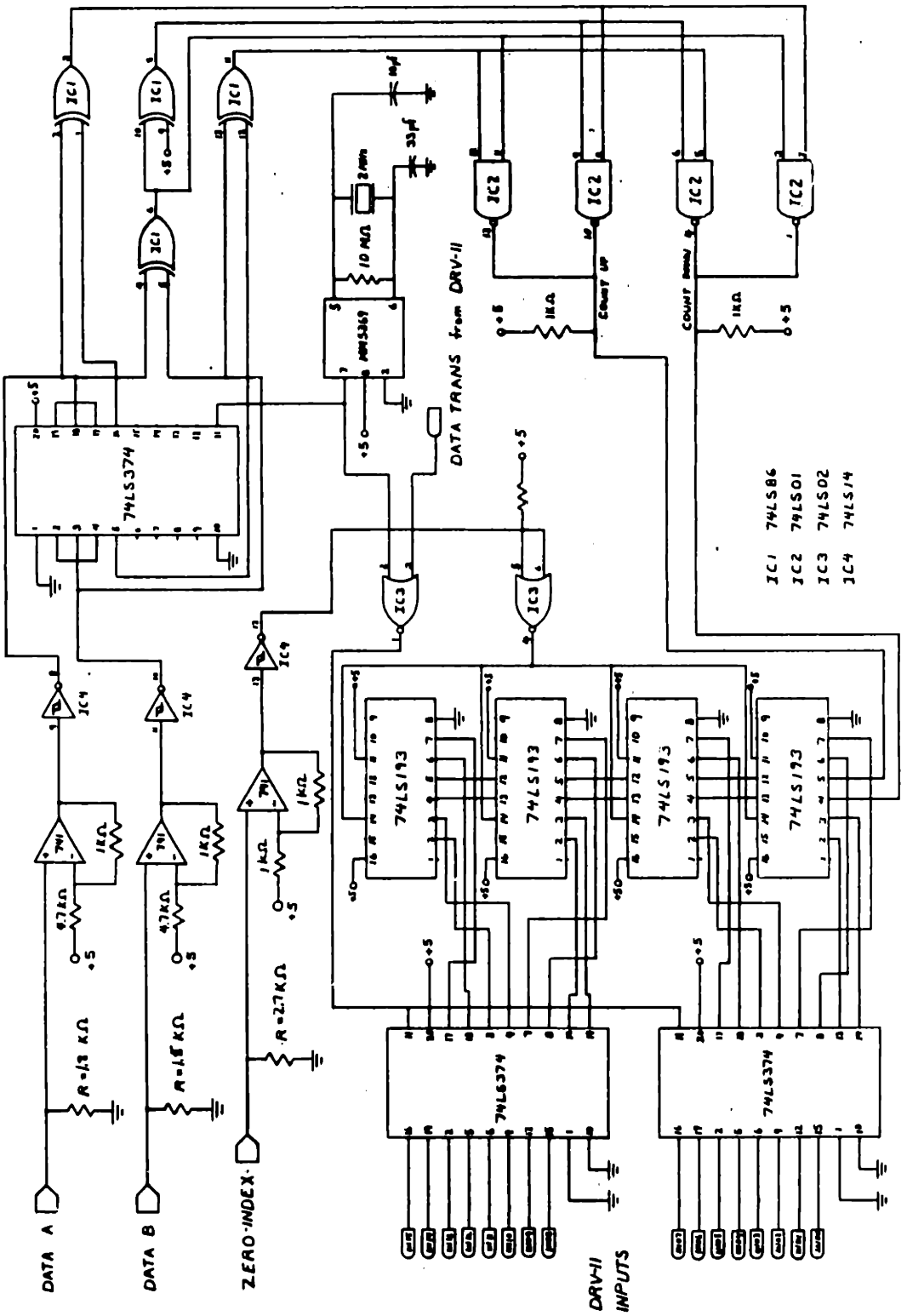


Figure 4.5 Optical Encoder-decoder Circuit

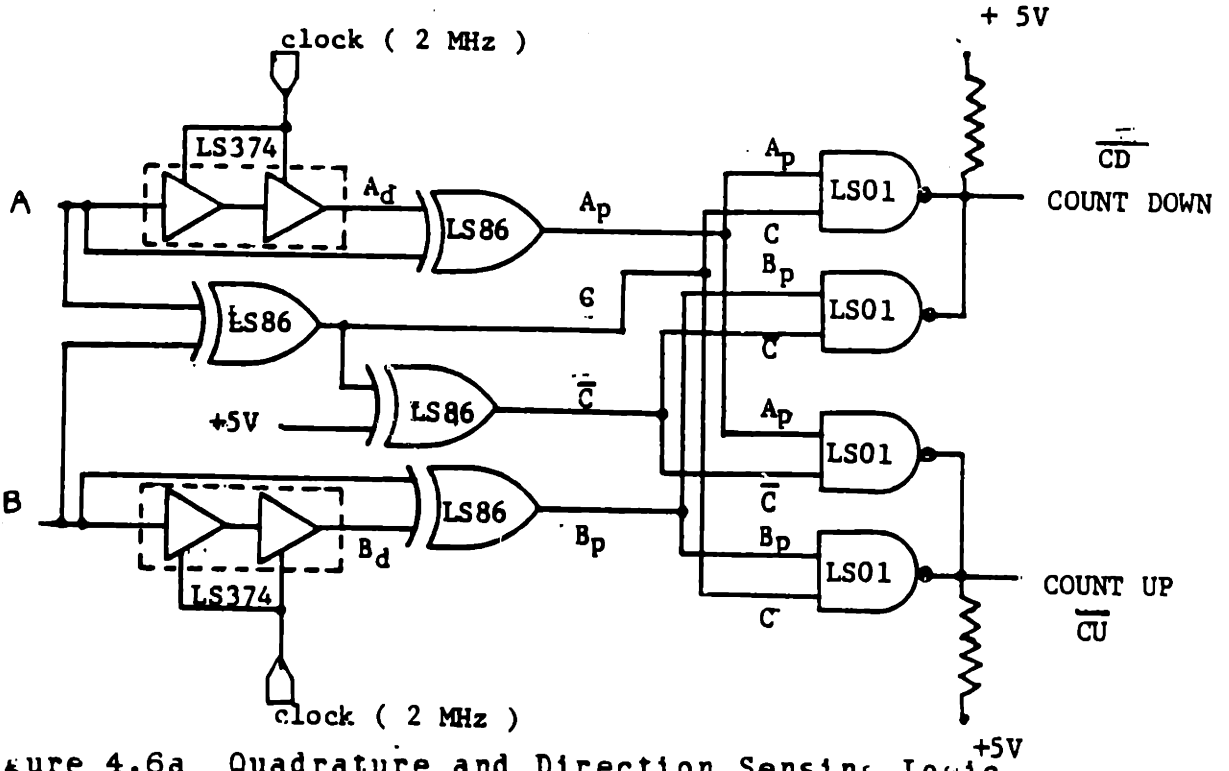


Figure 4.6a Quadrature and Direction Sensing Logic

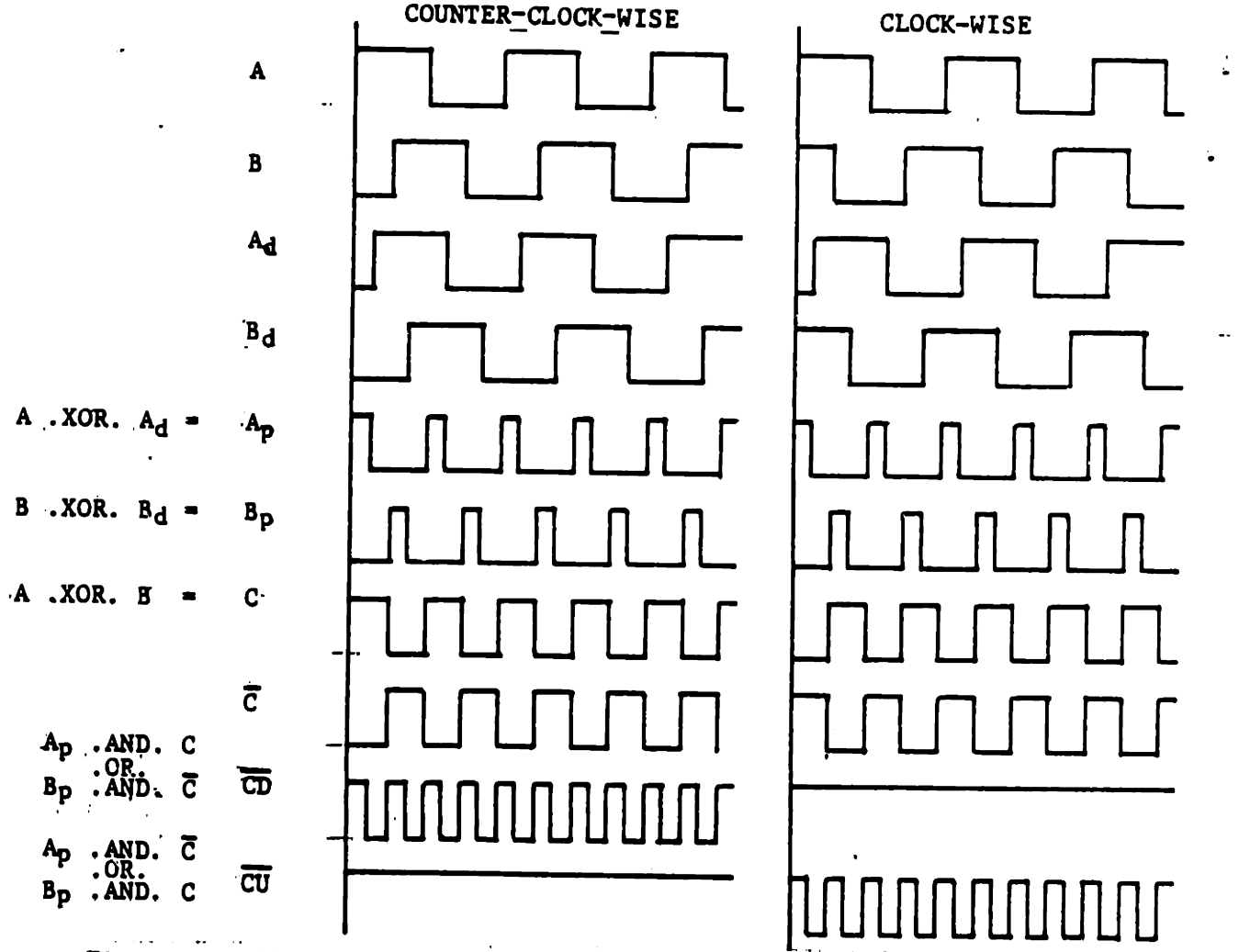


Figure 4.6b Quadrature and Direction Timing Diagram

The quasi-sine waves are 90 degrees out of phase with each other. This is known as phase quadrature [15]. Using phase quadrature it is possible to sense both direction and magnitude of movement. The zero index is fixed on the encoder and provides a pulse each time the encoder shaft turns one complete revolution. On the circuit, shown in Figure 4.5, the zero index is connected to the counter reset. Each time the encoder shaft passes through the zero index, the counters are reset to zero. While it is possible to put the arm in a configuration that will cause the encoder shaft to pass through the zero index (and reset the counters), this orientation, Figure 4.8, is not in the usable workspace shown in Figure 3.8.

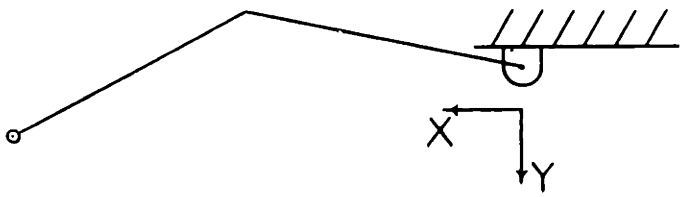


Figure 4.8 Manipulator Configuration for the Zero Index

Again it is stressed that these encoders do not output absolute position information. The decoding ciruitry to count pulses and output the current location in a binary parallel data form is discussed below.

The encoder output is not TTL compatible, thus the signals must first be sent through some operational amplifiers to increase thier magnitudes (TTL levels are 0 to .125 volts for logical low (0) and .24 to 5 volts for a logical high(1)). This signal is then sent through Schmitt triggers so that the edges will be more distinct. This generates TTL square pulse trains that are ready to be decoded. For the following

discussion, this TTL signal is now refered to as the original signal.

The decoding circuitry is comprised of TTL circuitry that is dependent on the phase separation of the two data signals. This phase separation is necessary because the Quadrature logic shown in Figure 4.6a generates a count pulse on both the leading and trailing edges of the original signals. Thus there are now 4 count pulses generated for each cycle of a single original data signal. Depending on which original signal pulse comes first (rotation clock-wise or counter-clock -wise), a count up or count down pulse is generated. These count pulses are then sent to counters.

The counting is done with a cascaded set of Synchronous 4-bit up/down counters (74LS193). These counters count up or down on the low-to-high level transition of the count up or count down signal from the previously described logic. The counters are provided with a reset (currently connected to the zero index) and borrow and carry outputs so that cascading them is straightforward. The outputs of the counters are put into D-type flip-flops (74LS374) that are used as data latches between the counters and the computer's parallel interface.

A timing diagram, shown in Figure 4.9, clarifies the way in which the data is held constant (latched) for the computer to read it. The parallel interface (DRV-11 board) in the computer needs at least 1150 nSec to read the data from the counters. The clock pulse in the encoder-decoder circuit is 500 nSec wide (2 Mhz). Using a Positive-nor (NOR) gate, the Data Transfer signal from the DRV-11 and the clock sig-

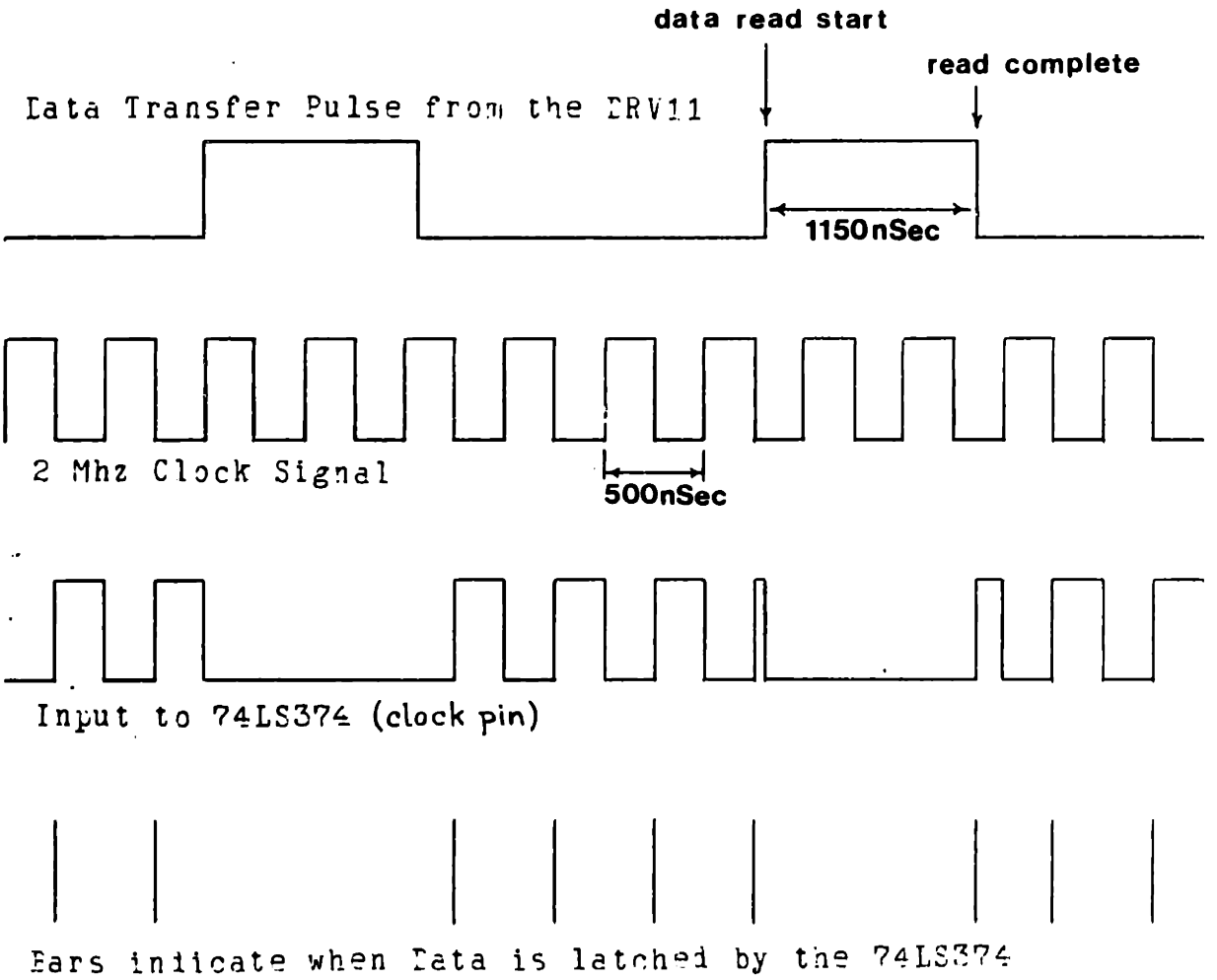


Figure 4.9 Parallel Interface Timing Diagram

nal on the encoder-decoder circuit together generate a pulse train at the frequency of the clock. This pulse train is then used to control the 74LS374 latch. The Data Transfer pulse goes high to signal that the DRV-11 is ready to accept data. On the positive transition of the Data transfer pulse, the data is held on the latch until the read is complete (that is, the data transfer pulse drops low).

This buffered parallel data is then set to two different devices. The parallel signal is sent directly to the digital computer through a DRV-11 parallel interface card and to a Digital-to-Analog converter. The parallel signal sent into the digital computer is then used in the digital implementation of the impedance control algorithm. There is a separate parallel interface board in the computer for each angle. This was done to avoid the unnecessary complexity of multiplexing the parallel signals onto a single parallel interface board.

CAUTION: There is a potential point of confusion about the output of this decoding circuitry. There are 8000 counts per revolution. Any computer program to use this information in the control of motion must use the appropriate scaling factor when determining angular position. The point of confusion is that, the output of the binary counters is indeed binary. If the rotation of the encoder shaft correponds to a count-down command to the binary counters, then there is no problem (with counting down) unless the binary counters are reset to zero (the shaft passes through the zero index configuration) and motion in this direction continues past this point.

There are two problems with this; first is the discrete transition from 0 radians to 2PI radians and the second is the binary representation of the encoder-decoder output. Without special considerations control strategies would have difficulty controlling motion through the discrete transition point. The other complication is that the angle represented by encoder counters is not exactly 2PI radians past the transition point. When the binary counters are counting down from 0 they use a 13 bit representation of the angle which corresponds to a count of 8191 units. This is a problem because there are only 8000 counts per revolution! With 8191 units, the angle just below 0 radians appears to be 6.433 radians (368.6 degrees) when it is actually only 2PI radians (360.0 degrees). The best way to avoid any confusion is to just never count down below zero.

4.3.2 D/A Converters For Encoder-decoders

This system was originally setup for both analog and digital control. D/A converters were constructed to generate analog signals from the digital information output by the encoder decoders. These D/A converters are not used currently in the control of the apparatus. However, it is useful to monitor their output with an oscilliscope to make sure that the encoder decoders are working properly. Each angle has its own D/A converter on a single board. Both of these boards must be plugged into the backplane during operation because of the role that they play in the relay operation covered section 3.3.2.

The D/A converters are Analog Devices AD565 High Speed 12-bit

Monolithic D/A Converters. Unfortunately a complete revolution of the encoder shaft generates 13 bits of information. Either the least significant bit of the encoder counter output must be unused (with the D/A converters), or the encoders must be limited to only 180 degrees of motion. This particular application limits the encoders to only 180 degrees of motion. It was felt that an accurate representation of the shaft angle is more important than the ability to measure beyond 180 degrees. Thus the least significant bit (LSB) of the encoder counter is tied to the LSB of the D/A converter, and the 12th bit of the encoder counter is tied to the most significant bit of the D/A converter. These bits would be wired exactly this way if there was not a problem with angular offsets.

The purpose of the D/A converters is to produce an analog signal that is equivalent to the digital representation of the angle used for control. To do this, it is necessary to adjust the D/A converter outputs with the same angular offsets (described in section 4.2.1) that are used with the angles in the digital computer.

To add the offset angles in this hard wired circuitry, it was necessary to use three 4 bit binary adders with the D/A circuitry. These binary adders add the signal from the encoder-decoders (the actual angle) and the signal from the computer (the angular offset) together. Thus the parallel signal from the encoder-decoders goes directly to the binary adders. The other input to the binary adders comes from the corresponding outputs of the parallel interface boards. On each DRV-11 (parallel interface board) in the computer, the input to the DRV-11 is

the actual encoder measurement and the output is the offset angle. It is the output of the binary adders that then goes to the D/A converters. Thus the D/A converter then generates an analog signal that can be monitored with an oscilliscope or used with analog feedback at a later date.

4.3.3 Tachometer Amplifier And Filter

This section discusses the circuitry necessary to both amplify and filter the tachometer signals. It also provides an explanation for the low cutoff frequency used in the tachometer low pass filter.

The tachometer sensitivity is very low (3 volts per 1000 RPM), therefore it is necessary to amplify the signal. This is done with an Analog Devices AD524 instrumentation amplifier. For this work, velocity is defined as positive in the counter-clockwise direction. Because the two torque motors are facing each other, it is necessary to reverse the tachometer output leads into the instrumentation amplifier. The top motor (outer link) has the postive tachometer lead connected to the positive input of the amplifier and the negative tachometer lead grounded. The bottom motor (inner link) has the negative tachometer lead connected to the positive input of the amplifier and the positive tachometer lead grounded.

The manufacturer of the torque motor/tachometer system suggests that the tachometer signal be filtered with a single pole low pass filter, cutoff frequency of 1000 Hz., to reduce the tachometer ripple. The tachometer ripple is a signal that arises due to tachometer armature

moving past a finite number of magnetic poles. This filter recommendation is probably based on the assumption that the tachometer signal is feed directly into the servo amplifier and that the servo amplifier compensation circuit is intact. This application, however, does not input the tachometer signal directly into the servo amplifier, but directly into the digital computer through a D/A converter.

Tachometer ripple, however, was found to be insignificant in comparison to vibrations associated with the support frame (described in section 3.1). This framework appears to be vibrating at 400 hertz, a frequency which is well below the suggested 1000 hertz cutoff frequency for the tachometer ripple. To demonstrate the complications that arise due to this 400 hertz noise two tests were performed. The first test was done to identify the noise and the second test shows the problems this noise caused when a velocity feedback loop was closed around the system. For both tests the outer link was removed from its torque motor.

The first test was completely open loop. For this test the torque motor terminals were short circuited and the motor was completely passive. Several points on the supporting framework were lightly tapped with a lucite hammer and measurements were made of the unfiltered amplified tachometer signal. These measurements were made on an FM tape recorder with a tape speed of 15 in./sec. (bandwidth of D.C. to 5000 hertz). The tape was then played back onto strip chart recorder at a tape speed of 15/16 in./sec. (16 times slower). By doing this it was possible to retrieve high frequency information on device with a rela-

tivly low bandwidth (the strip chart recorder has a bandwidth of about 200 hertz). Figure 4.10 shows a typical plot of the amplified tachometer signal.

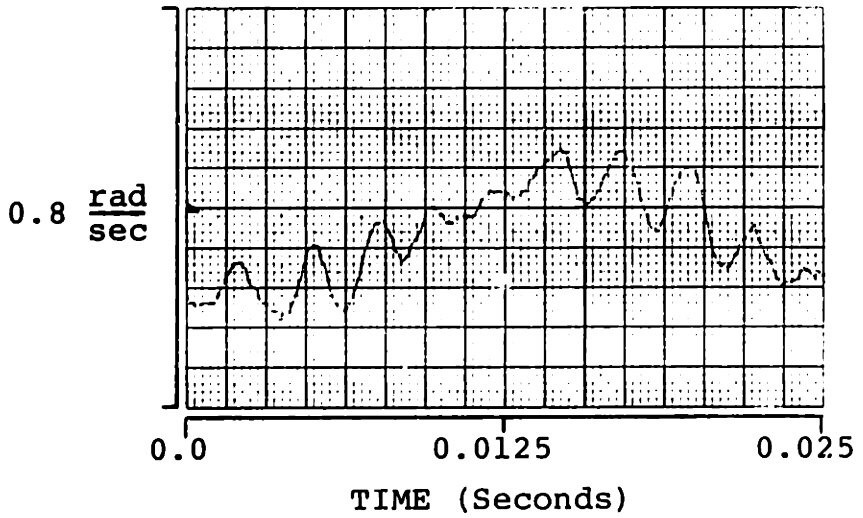


Figure 4.10 Amplified Tachometer Signal for Frame Tap Test

From this plot it is readily apparent that there is an approximately 400 hertz signal on the tachometer output due to the vibration of the support. The problem with this 400 hertz noise becomes even greater when a velocity feedback loop is closed in the system to produce a viscous load on the link.

The second test was done with velocity feedback. For the velocity feedback case, the torque motor was active and the signal that was used in the feedback was filtered with a low pass filter with a cutoff frequency of 21 hertz. Light taps were again applied to the supporting framework and measurements were made of the unfiltered amplified tachometer signal. Figure 4.11 shows a typical plot of the noise present in this case. One thing to note here is the unexplained spontaneous bursts

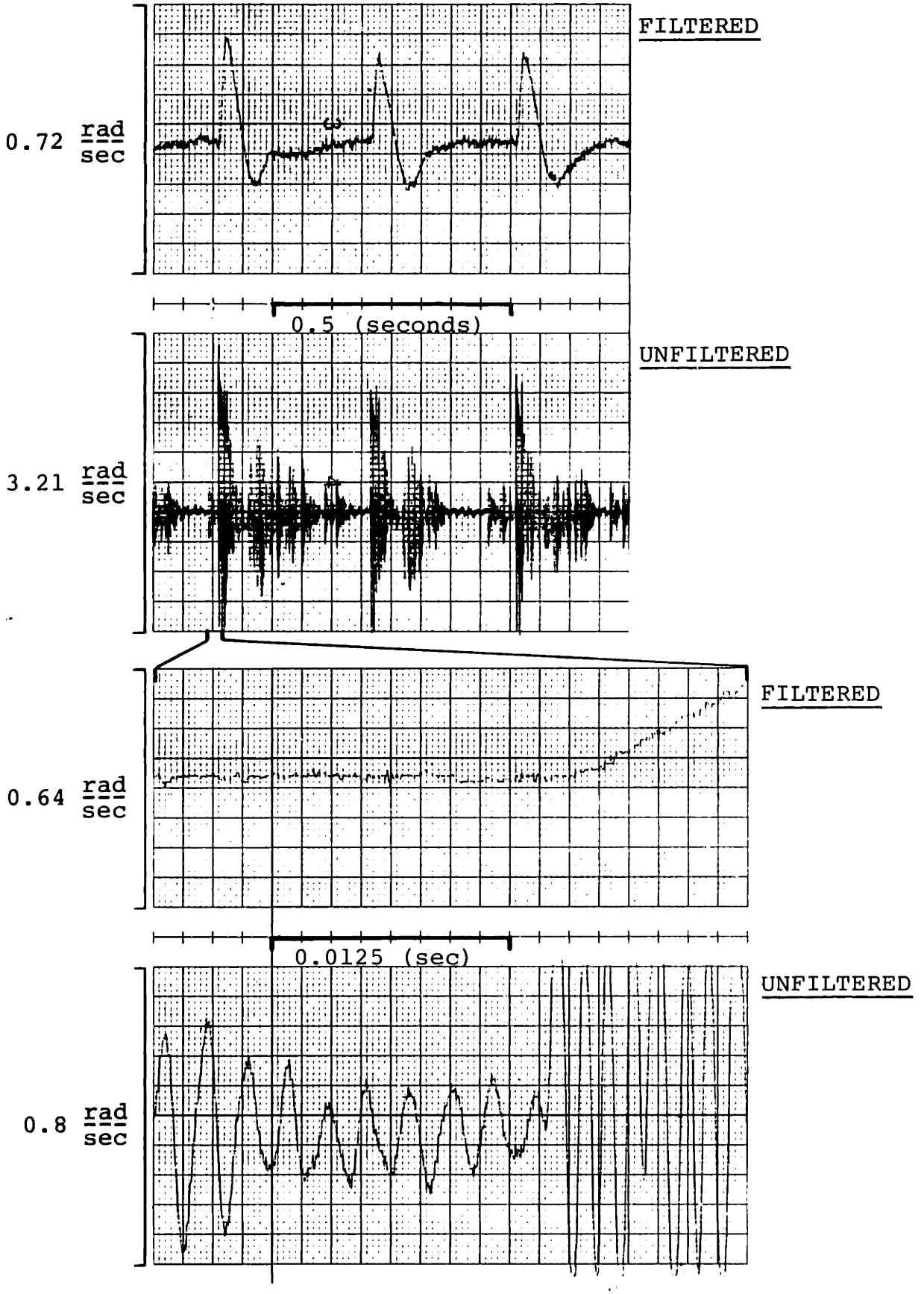


Figure 4.11 Amplified Tachometer Signal with Velocity Feedback

of noise. This figure also shows the filtered velocity signal that was fed back.

Clearly there is a need to filter the tachometer signal to get rid of this noise. A second order Butterworth filter with a cutoff frequency of 50 hertz was tried in the velocity feedback loop, but this was insufficient to reduce the noise. Since this filter was not effective in attenuating the noise, a 4^{th} order low pass filter was implemented. This 4th order low pass filter was found to be effective at eliminating the noise due to the stand vibration. Figure 4.12 shows a plot of the filtered tachometer signal for the same test that produced the unfiltered tachometer signal in Figure 4.10.

This filter is comprised of 4 passive first order filters cascaded together with operational amplifiers set up as voltage follwers. Each stage in this filter in designed to have a nominal cutoff frequency (-3 dB on the magnitude response) of 50 hertz, but the overall cutoff frequency is at 21 hertz. Figure 4.13 shows a Bode plot for the theoretical 4^{th} order filter.

Although this filter is very effective in eliminating the noise, it is by no means the optimal solution to the noise problem. It is used in this application because of its ease of implementation and it does in fact work. One problem with this filter is that it has significant phase lag at low frequencies (5 degrees of phase lag at 1.0 hertz). As shown in chapter 2, the theoretical bandwidth of the controller is approximately 14 hertz. With the phase lag from the filter there will

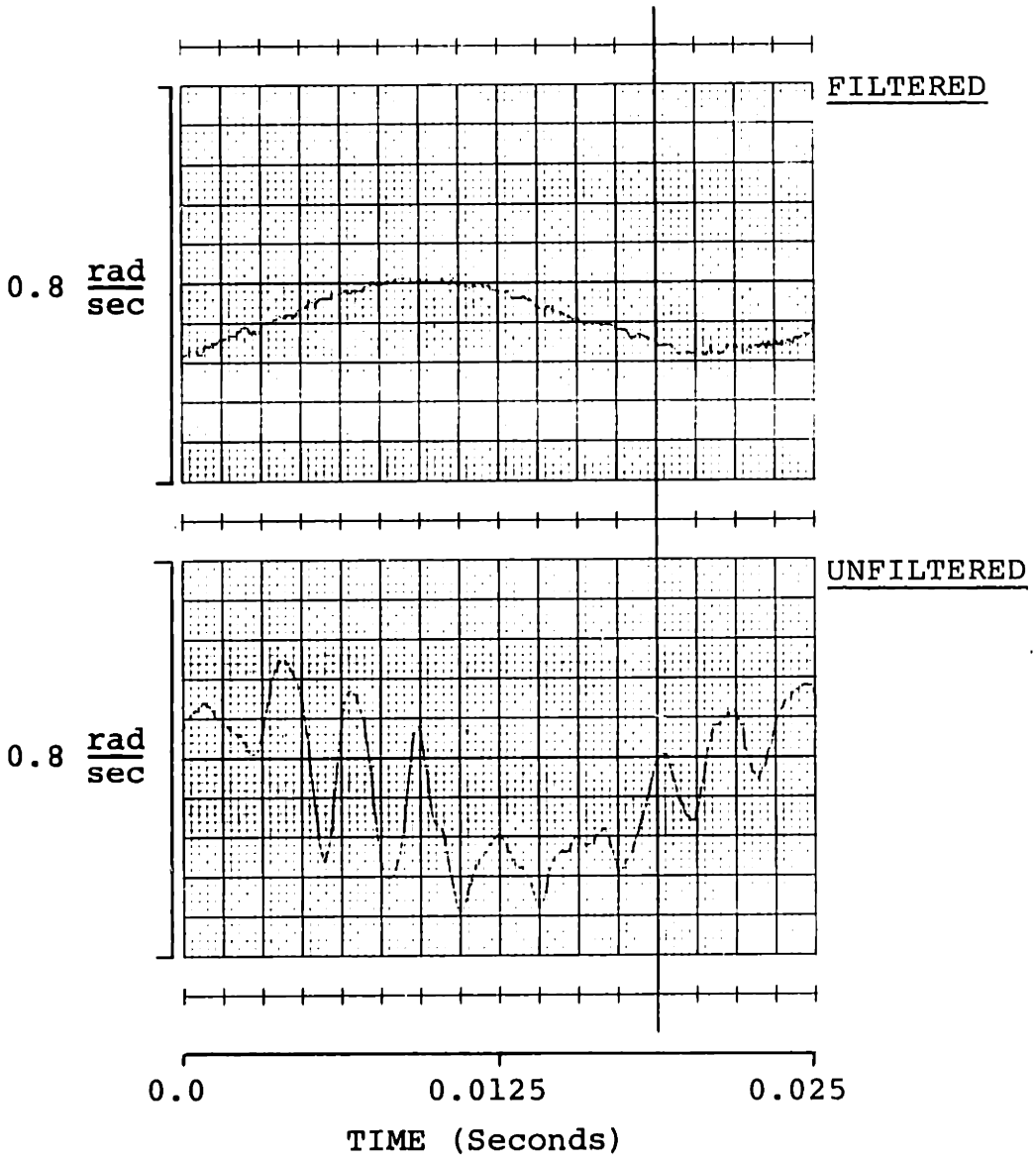


Figure 4.12 Low Pass Filtered Tachometer Signal

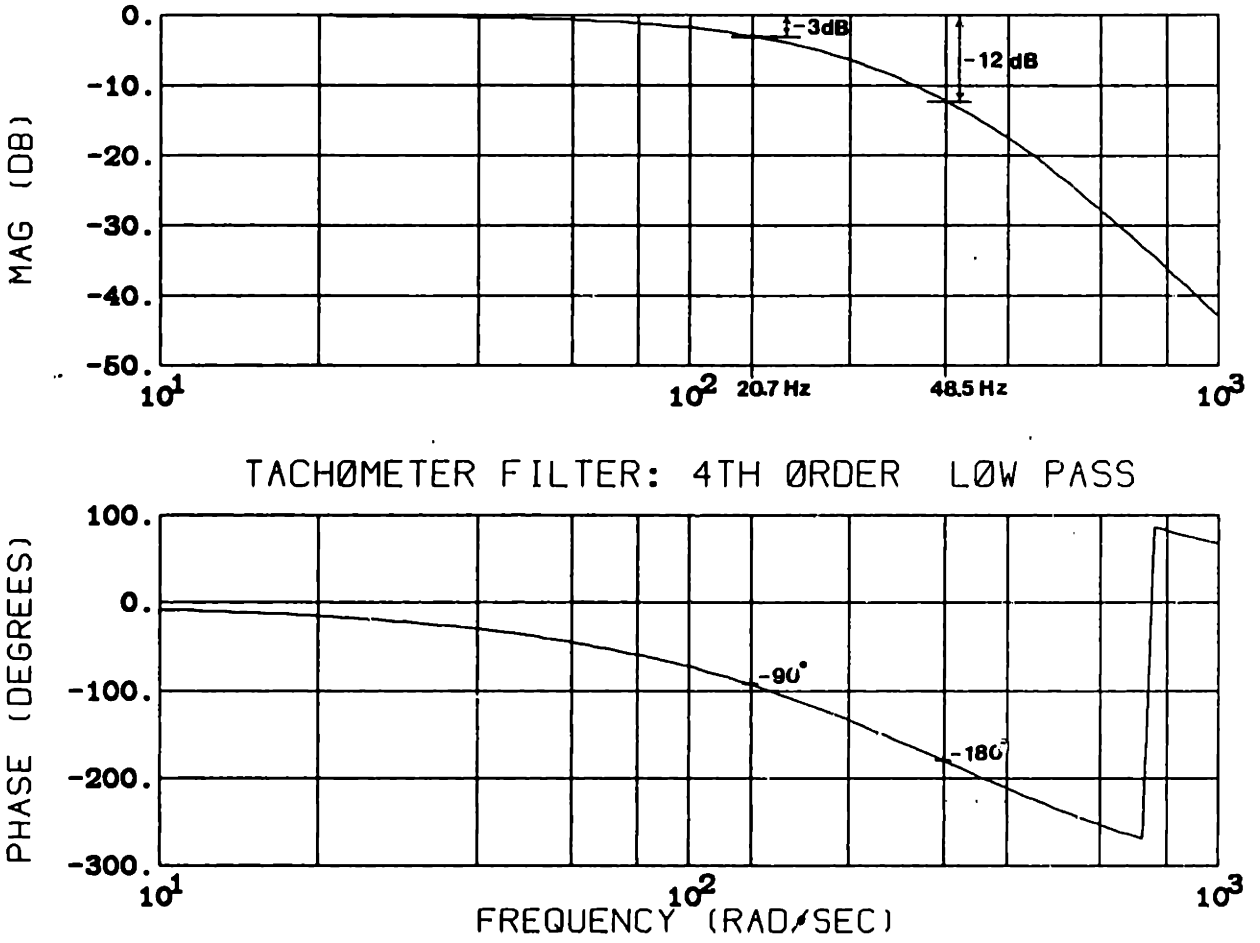


Figure 4.13 Theoretical Bode Plot for 4th Order Low Pass Filter

certainly be some reduction in the controller bandwidth.

4.3.4 Control Circuitry Backplane

The backplane to hold all of the circuitry used here is a VECTOR CCM14A card cage. One purpose of the backplane is to provide power ($\pm$15 volts and +5 volts) and ground to all of the circuit boards. The most important function, however, is to provide a means of making reliable connections between circuit boards. Figure 4.14 shows the basic layout of the backplane as viewed from the back of the card cage. The numbers and letters on the figures correspond to the numbers (and letters) on the edge connectors and on the circuit boards themselves. The backplane contains both analog and digital circuitry. The analog circuitry is not used in this application, but will be used at a later date.

BACK PLANE -LEFT SIDE -REAR VIEW

SPARE		ENCODER INTERFACE		ENCODER DECODER θ_2		ENCODER DECODER θ_1		ENCODER D/A θ_2		ENCODER D/A θ_1	
+15-22	Z-AGND	+15-22	Z-AGND	+15-22	Z-AGND	+15-22	Z-AGND	+15-22	Z-AGND	+15-22	Z-AGND
+5-21	Y--15	+5-21	Y--15	+5-21	Y--15	+5-21	Y--15	+5-21	Y--15	+5-21	Y--15
DGND-2Ø	X-	DGND-2Ø	X-	DGND-2Ø	X-	DGND-2Ø	X-	DGND-2Ø	X-	DGND-2Ø	X-
-19	W-	2CSR1-19	W-2I12	-19	W-	-19	W-	-19	W-	-19	W-
-18	V-	1CSR1-18	V-1I12	-18	V-	-18	V-	-18	V-	-18	V-
-17	U-	-17	U-	-17	U-	-17	U-	-17	U-	-17	U-
-16	T-	DGND-16	T-LMTS	-16	T-	-16	T-	-16	T-	-16	T-
-15	S-	-15	S--5	-15	S-	-15	S-	-15	S-	-15	S-
-14	R-	DGND-14	R-RLY	-14	R-	-14	R-	-14	R-	-14	R-
-13	P-	-13	P-	2I12-13	P-	1I12-13	P-	2I12-13	P-	1I12-13	P-
-12	N-	AGND-12	N-AGND	2I11-12	N-	1I11-12	N-	2I11-12	N-	1I11-12	..
-11	M-	-11	M-	2I1Ø-11	M-	1I1Ø-11	M-	2I1Ø-11	M-	1I1Ø-11	M-
-1Ø	L-	-1Ø	L-IDX2	2Ø9-1Ø	L-	1Ø9-1Ø	L-	2Ø9-1Ø	L-AGND	1Ø9-1Ø	L-AGND
-9	K-	-9	K-DATB2	2Ø8-9	K-	1Ø8-9	K-	2Ø8-9	K-2DR15	1Ø8-9	K-1DR15
-8	J-	-8	J-DATA2	2Ø7-8	J-	1Ø7-8	J-	2Ø7-8	J-2DR14	1Ø7-8	J-1DR14
-7	H-	-7	H-	2Ø6-7	H-	1Ø6-7	H-	2Ø6-7	H-2DR13	1Ø6-7	H-1DR13
-6	F-	-6	F-	2Ø5-6	F-	1Ø5-6	F-	2Ø5-6	F-2DR12	1Ø5-6	F-1DR12
-5	E-	-5	E-	2Ø4-5	E-	1Ø4-5	E-	2Ø4-5	E-FBP2	1Ø4-5	E-FBP1
-4	D-	-4	D-IDX1	2Ø3-4	D-IDX2	1Ø3-4	D-IDX1	2Ø3-4	D-2REQA	1Ø3-4	D-1REQA
-3	C-	-3	C-DATB1	2Ø2-3	C-DATB2	1Ø2-3	C-DATB1	2Ø2-3	C-2INIT	1Ø2-3	C-1INIT
-2	B-	-2	B-DATA1	2Ø1-2	B-DATA2	1Ø1-2	B-DATA1	2Ø1-2	B-2CSR1	1Ø1-2	B-1CSR1
-1	A-	-1	A-	2ØØ-1	A-	1ØØ-1	A-	2ØØ-1	A-2NDR	1ØØ-1	A-1NDR

AGND	ANALOG GROUND
DGND	DIGITAL GROUND
#IØØ	DIGITAL DATA TO DAC θ_n
LMTS	LIMIT SWITCHES
RLY	RELAY CONTROL

FBP#	POSITION FEEDBACK OUTPUT OF θ_n
IDX#	ENCODER θ_n ZERO INDEX
DATA#	ENCODER θ_n DATA A
DATB#	ENCODER θ_n DATA B
#DRØØ	DRVII DIGITAL INPUTS TO DAC θ_n

#I12 TIED TO #REQA

Figure 4.14 Back Plane Distribution, Left Side, Rear View

CHAPTER 5: SYSTEM MODEL

This chapter presents a mathematical model of the two link manipulator. This model of the manipulator is developed for two reasons: 1) It is to be used in simulations aimed at evaluating the effectiveness of the controller, and 2) It is developed in anticipation of controlling the apparatus with a model based impedance controller. The impedance controller proposed by Hogan [11] requires information about the manipulator dynamics to modify inertance. To implement this impedance controller in further work, it is important to develop an accurate model of the manipulator. Once this model has been developed, it is possible to simulate movements of the modelled manipulator and compare those movements with actual movements of the physical manipulator. The final section of this chapter discusses the method of simulation.

Development of this model involved assumptions about both the manipulator links themselves and the torque motors used to drive them. These assumptions and modelling decisions are discussed in the following next couple of sections.

5.1 Manipulator Dynamics

It is assumed that the manipulator consists of rigid links that have both mass and rotational inertia. For this model, the structural dynamics of the links and the four-bar-linkage drive mechanism are ignored. It is assumed that there is some ideal transmission mechanism

that transmits the torque from the motor, mounted on the support frame, to the outer link. Any structural dynamics of the four-bar-linkage are neglected in the model.

The contribution of the four-bar-linkage to the inertia tensor was approximated by assuming its mass is added into the rotational inertia of the outer link. The quantitive value for the rotational inertia of the outer link was determined by a swing test, and the quantitive value for the rotational inertia of the inner link was determined by direct computation of the contribution of each of the link components. These computations and data are included in Appendix C. In the swing test, the outer link was allowed to swing freely in the vertical plane about its axis of rotation. The damped natural frequency of the oscillations was measured and used with a measurement of the link mass to compute the rotational inertia. This parameter is listed in table 5-1.

TABLE 5-1 Manipulator Link Parameters

Parameter	Units	Value
Inner link length, ℓ_1	(meters)	0.3654
Outer link length, ℓ_2	(meters)	0.342
Outer link center of gravity, h_2	(meters)	0.1572
Outer link mass, m_2	(Kg.)	0.5952
Inner link moment of inertia, J_1	(N-m-sec-sec)	0.0285
Outer link moment of inertia, J_2	(N-m-sec-sec)	0.0285

The swing test was used to determine not only the outer link rotational inertia, but also to quantify the amount of friction present in the joint between the inner and outer links. This swing test was done with the four-bar-linkage attached to the outer link. Thus, this test

also quantifies the friction in the drive mechanism as well as the contribution of the dyanmics of the four-bar-linkage to the inertia matrix below in equation 5-1.

$$\begin{bmatrix} J_1 + \ell_1^2 m_2 & \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) \\ \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) & J_2 \end{bmatrix} \qquad [5\text{-}1]$$

For this reason, the rotational inertia computed using the swing test is used in the model and in all the simulations.

The apparatus is designed to be set up in the horizontal plane and hence gravitational forces are absent on the manipulator links and not included in the model. This model assumes low (but not zero) friction at the connection between the two links. This friction, as determined with the swing test, is probably due to Coulomb friction but was included in the model as a viscous force. This modelling decision was made to simplify the model and could be changed at a later date if the differences between Coulomb friction and viscous friction are found to be significant.

5.2 Torque Motors and Servo Amplifiers

The torque motors are mounted on a nearly static support (the vibration of the support framework was discussed in section 4.3.3). This model, however, assumes that the motors are applying pure torques from an absolute (non-accelerating) reference frame. This reference frame is located, as shown in Figure 3.1, at the base of the inner link.

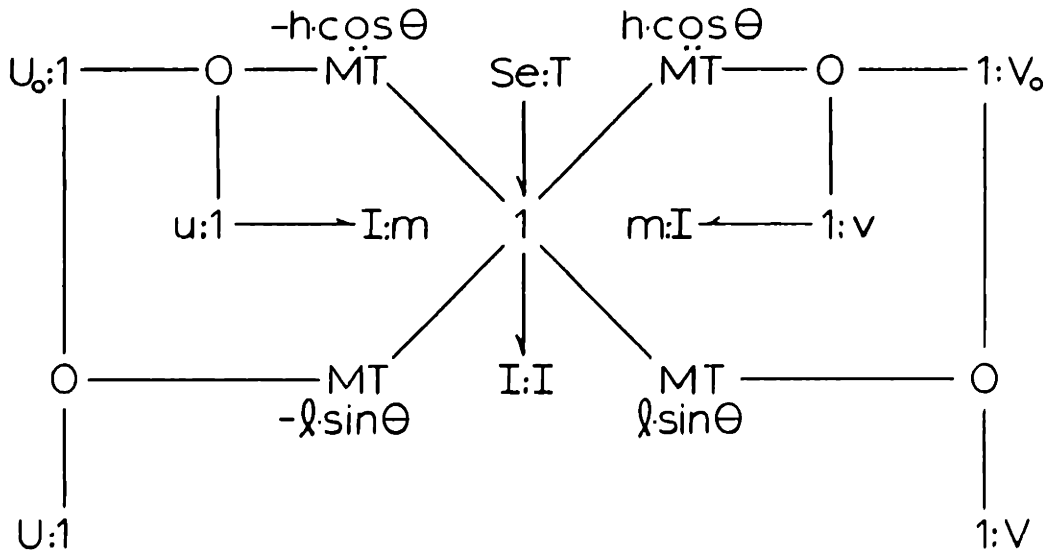
Each motor is also modelled with non-zero internal bearing friction

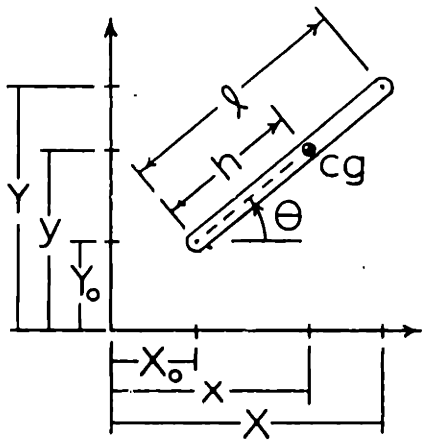
and some rotational rotor inertia. The servo amplifiers are modelled as pure voltage controlled current sources (a transconductance gain, Ka) and the motor inductance is ignored. Thus the model includes the motors as voltage controlled torque sources. All of the aforementioned parameters for linkage dynamics are listed in table 5-1 and the motor constants are listed in table 3-1.

5.3 Modelling Technique

The development of a complete system model was approached using an extremely powerful modeling technique known as bond graphing [16,23]. As shown in Figure 5.1, it is possible to apply bond graphs to kinematic mechanisms. This figure shows a bond graph model that is a specific case for a single link or component of a mechanism. These single component bond graphs can be easily cascaded to develop models of more complex planar systems.

For this work, two components of Figure 5.1 were cascaded and then reduced to a manageable form. Figure 5.2 is the reduced form of the bond graph used to derive the equations of motion.





$$U_o = \frac{dX_o}{dt} \qquad V_o = \frac{dY_o}{dt}$$

$$u = \frac{dx}{dt} \qquad v = \frac{dy}{dt}$$

$$U = \frac{dX}{dt} \qquad V = \frac{dY}{dt}$$

m is the mass of the link

I is the moment of inertia about the center of gravity

T is the torque applied to the link

Figure 5.1 Bond Graph Model for a Kinematic Component

From course notes N. Hogan

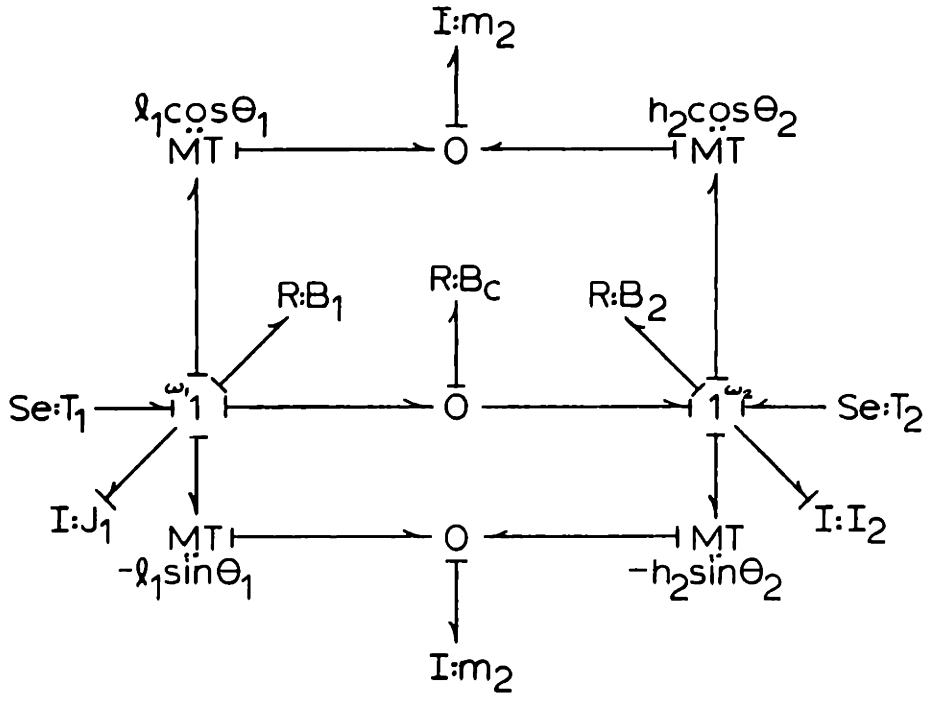


Figure 5.2 Bond Graph Model for a Two-link Manipulator

The derivation is done in Appendix A. A derivation using Lagrange techniques is done in Appendix B. Both derivations are presented here so that readers unfamiliar with bond graphing can follow the math in the Lagrange derivation. Both derivations give the same results, but in the bond graph derivation care must be taken in assigning the power flow directions. If the power flow is not assigned properly for the dissipator (resistive element modelling the friction between the links) on the elbow joint, then energy may be added to the system instead of dissipated.

The final results of the derivation are a nonlinear model of the manipulator:

$$\begin{bmatrix} J_1 + \ell_1^2 m_2 & \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) \\ \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) & J_2 \end{bmatrix} \begin{vmatrix} \frac{d\omega_1}{dt} \\ \frac{d\omega_2}{dt} \end{vmatrix} = \qquad [5\text{-}2]$$

$$\begin{bmatrix} -(B_1 + B_c) & B_c \\ B_c & -(B_2 + B_c) \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} + \begin{bmatrix} T_1 + \ell_1 h_2 m_2 \omega_2^2 \sin(\theta_2 - \theta_1) \\ T_2 - \ell_1 h_2 m_2 \omega_1^2 \sin(\theta_2 - \theta_1) \end{bmatrix}$$

Where B_1 and B_2 are the frictional contribution from each motor, and Bc is the friction at the interface between inner and outer links. From equation 5-2 it is clear that the inertia tensor is not a static quantity. With the torque motors applying torques to both links from a stationary reference frame, the diagonal terms of the inertia matrix are constant. The off-diagonal terms, however, are position varying terms and are dependent on the relative angle between the links. Likewise, the Coriolis terms are also dependent on the relative angle between the links.

This model has been developed here to be used later with simulations of manipulator movements. These simulations are aimed at verifying the model so that it can be used in further work with impedance controllers. A comparison of actual manipulandum movements and simulated manipulandum movements is shown in the next chapter. Obviously, this type verification is directly dependent on the proper

operation of the control strategy. The control strategy in the simulation must perform in the same way as that used in the actual hardware. It will be shown in the next chapter that actual control strategy is performing satisfactorily.

This model has been developed in the absolute joint angle coordinates of the manipulator. The impedance controller, however is controlling the endpoint in cartesian endpoint coordinates. Although the absolute and relative angles have been defined in chapter 3, they are re-examined here to clarify the relationships between the coordinate frames.

5.4 Coordinate Frames and Transformations

There are several different coordinate frames that are of interest in both the control and dynamic representation of a manipulator. In this application the joint torques are applied in an absolute reference frame, but the control of endpoint impedance is done in endpoint cartesian coordinates. Thus it is necessary to develop transformations between endpoint coordinates and absolute joint coordinates. The most natural representation of inertial components is with an inertial coordinate frame [4].

For this work, two coordinate frames are of interest: absolute or joint coordinates, and endpoint coordinates.

A) Joint coordinates:

$$q_1 = (\theta_1, \theta_2) \qquad [5\text{-}3]$$

B) Endpoint coordinates:

$$q_2 = (x, y) \qquad [5\text{-}4]$$

The transformation from joint coordinates to endpoint coordinates, the forward kinematic relationship, is simple and straightforward:

$$\underline{X} = L(\underline{\theta}) \qquad [5\text{-}5]$$

or

$$q_2 = L(q_1)$$

Where L() is a function of the joint angles. This can be written out:

$$x = \ell_1 \cos(\theta_1) + \ell_2 \cos(\theta_2) \qquad [5\text{-}6]$$

$$y = \ell_1 \sin(\theta_1) + \ell_2 \sin(\theta_2) \qquad [5\text{-}7]$$

The Jacobian has been defined as the relation between differential displacements of the cartesian endpoint position and differential displacements of the manipulator's absolute joint angles. The Jacobian matrix, J, can also be thought of as a transformation between endpoint cartesian endpoint velocities and joint angular velocities. Differentiating relationships 5-6 and 5-7 above:

$$u = -\ell_1 \omega_1 \sin(\theta_1) - \ell_2 \omega_2 \sin(\theta_2) \qquad [5\text{-}8]$$

$$v = \ell_1 \omega_1 \cos(\theta_1) - \ell_2 \omega_2 \cos(\theta_2) \qquad [5\text{-}9]$$

Written in matrix form:

$$\underline{V} = \underset{\sim}{J}\underline{\omega} \qquad [5\text{-}10]$$

Where V is the vector of endpoint velocities and W is the vector of joint angular velocities. The Jacobian is then written in matrix form:

$$\begin{bmatrix} -\ell_1 \sin(\theta_1) & -\ell_2 \sin(\theta_2) \\ \ell_1 \cos(\theta_1) & \ell_2 \cos(\theta_2) \end{bmatrix} \qquad [5\text{-}11]$$

The transformation from endpoint coordinates to joint coordinates,

the inverse kinematic relationship, is nontrivial and is thought to be the most difficult problem in manipulator control [20]. The "inverse kinematics" problem, however, is not difficult for a two link manipulator, and it is again pointed out that the impedance control relationship given in equation 2-2 does not require the computation of inverse kinematics, but only the forward kinematic relation given in relation 5-5. Thus in the control of the manipulator in this work, the inverse kinematics problem is not an issue because it is not done.

With all of the relations above, it is possible to develop a competent simulation of the manipulator and any control strategy that might be used to control the apparatus.

5.5 Dynamic Simulation

As stated previously in this chapter, one reason for developing an accurate model of the manipulandum is to be able to simulate actual movements of the manipulator. Now that the model has been developed in equation 5-2, a simulation can be developed that will generate a time history of the states of the system. That is, this simulation will take the equations of motion (given in equation 5-2) and a set of initial conditions, and will output time histories of the absolute joint angles and joint angular velocities. This is done by solving the equation of motion for the angular accelerations in equation 5-2.

$$\begin{bmatrix} \frac{d^2\theta_1}{dt^2} \\ \frac{d^2\theta_2}{dt^2} \end{bmatrix} = \begin{bmatrix} J_1 + \ell_1^2 m_2 & \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) \\ \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) & J_2 \end{bmatrix}^{-1}$$

$$\left[\begin{bmatrix} -(B_1 + B_c) & B_c \\ B_c & -(B_2 + B_c) \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} + \begin{bmatrix} T_1 + \ell_1 h_2 m_2 \omega_2^2 \sin(\theta_2 - \theta_1) \\ T_2 - \ell_1 h_2 m_2 \omega_1^2 \sin(\theta_2 - \theta_1) \end{bmatrix} \right] \qquad [5\text{-}12]$$

These angular accelerations are then integrated by the simulation package to give the angular velocities of the links. The angular velocities are then integrated to yield the angular positions of the links. Using the forward kinematics in relation 5-5 (relations 5-6 and 5-7) it is possible to have the simulation package output the endpoint trajectory of the manipulator.

The simulation of the nonlinear manipulator model was written using DYSYS. DYSYS is a FORTRAN program which integrates first order ordinary differential equations. These equations can be linear or non-linear and the integration technique used is a fourth order Runge-Kutta approximation. The relation 5-12 above is written in a form that can be converted easily into FORTRAN code and input to DYSYS. These equations are presented to DYSYS in the form of a subroutine known as an EQSIM (EQuation SIMulator). This EQSIM subroutine can also contain other FORTRAN statements that will add other non-linearities, such as servo-amplifier saturation and static friction, to the simulation.

The DYSYS used for all of these simulations is a version called Portable DYSYS. All of the simulations were done on a PDP 11/60 computer (With an RSX11M V4.1 operating system) in the Newman Laboratory For Biomechanics And Human Rehabilitation. There is approximately 64k of addressable memory in this computer and this is the only limitation to the the complexity of the EQSIM. Portable DYSYS on the PDP 11/60 is currently limited to 20 state variables. A copy of the EQSIM for the simulation of the manipulator is listed in Appendix E.

The simulation was performed with DYSYS and the results were found to be quantitatively consistent with actual data from movements made with the manipulator. The simulation data along with the work that was done to verify the endpoint impedance controller is all presented in the next chapter.

CHAPTER 6: CONTROLLER EVALUATION

This chapter presents the results from experiments that were aimed at evaluating force control at the manipulator endpoint (equation 2-8) and experiments to evaluate the ability of the impedance controller (equation 2-2) to produce the desired endpoint behavior. Data on force generation at the endpoint will be presented as well as data on the use of the servo amplifier current monitor to estimate the forces that are commanded at the endpoint. Experiments are then discussed and data presented for the evaluation of the impedance controller.

The final sections of this chapter present two experiments that were performed with human subjects to demonstrate some of the uses of this manipulandum. These experiments are included to give the reader an idea of what types of new investigations might be done with the apparatus.

6.1 Static Force Evaluation

One of the goals of this work is to provide an apparatus that will be capable of producing forces at the handle in any desired direction. Chapter 2 presented a discussion of the control necessary to generate forces at the manipulator handle and this section addresses the question of how accurately endpoint forces can be produced.

The verification of this desired endpoint force production is sim-

ple and straightforward. A two axis force joystick (Measurement Systems Model 436 Hand Control) was mounted rigidly out in the workspace of the manipulator and the endpoint of the manipulator was attached to the force sensor as shown in the Figure 6.1. The manipulator endpoint was then commanded to push with the same magnitude (4 and 10 Newtons) at zero impedance in various directions.

One problem encountered with this force transducer was that there was an offset in the transducer that could not be nulled out. It was possible to minimize this problem with software. Before each force generation run, 100 samples of the force transducer output were taken and averaged to get an estimation of the offset. This offset was then subtracted in post-processing of the data.

The endpoint force production was tested in three different orientations as shown in Figure 6.2.

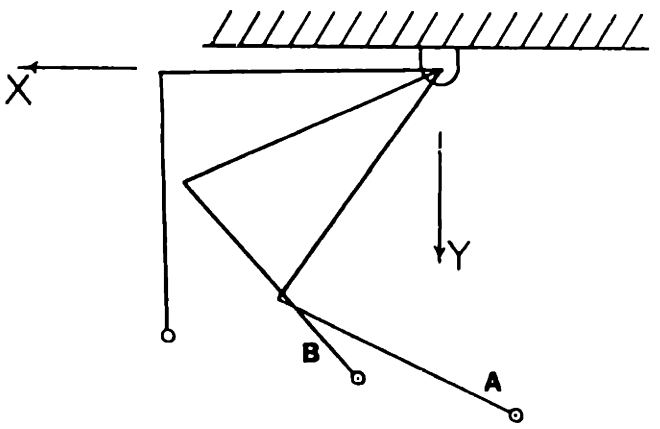


Figure 6.2 Orientations of the Manipulator for Force Generation Tests

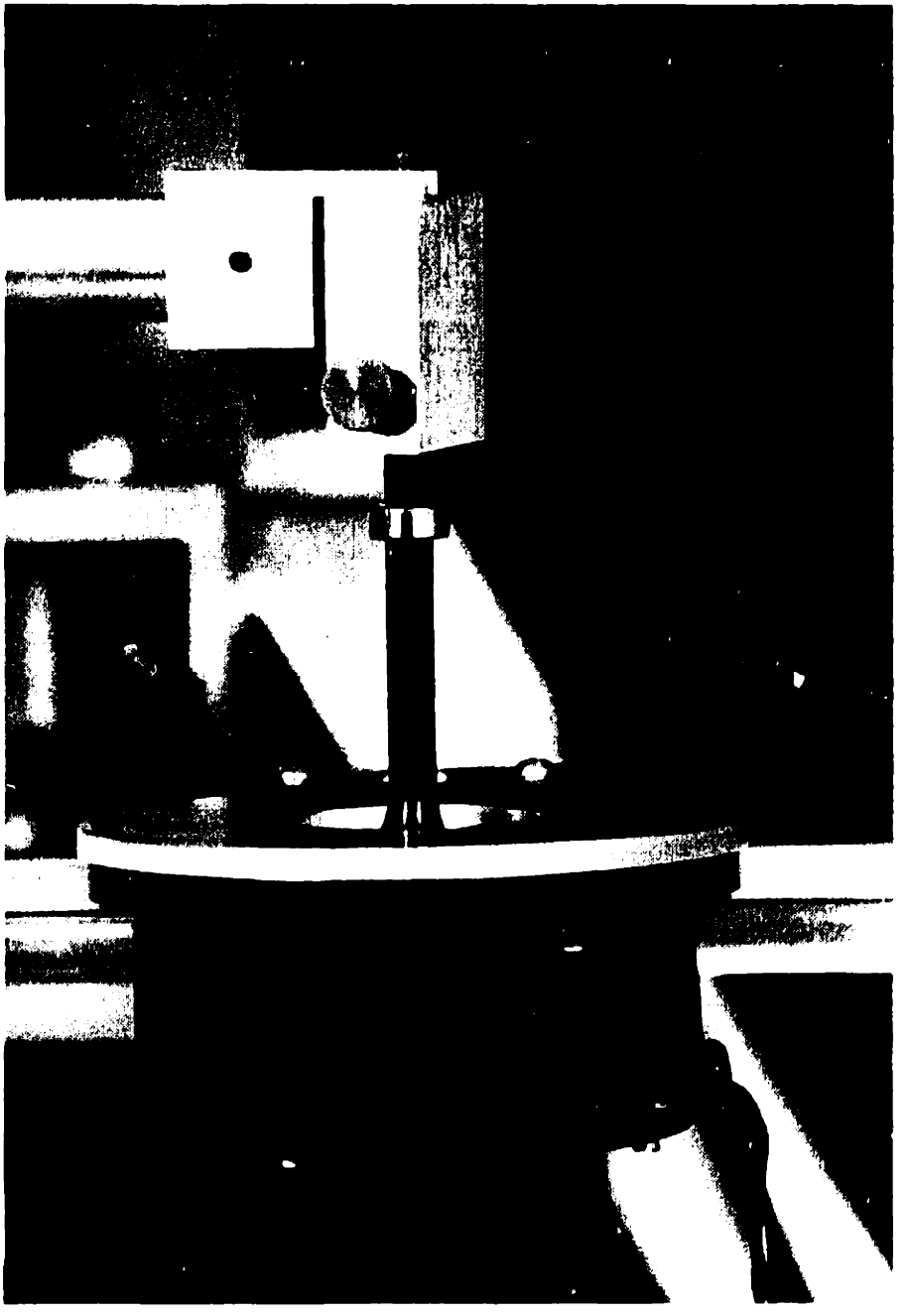


Figure 6.1 Manipulator Endpoint Attached to Force Transducer

One orientation corresponds to the position where the encoders are calibrated. The other two orientations were picked as endpoint locations in the workspace which would be representative of experimental locations used with human subjects. One orientation, A, had the Cartesian endpoint coordinates x = -.1016 meters, y = .40 meters and the other orientation, B, had the Cartesian endpoint coordinates x = .1056 meters, y = .4496 meters. For the calibration and A orientations, the forces were 10 Newtons in all the directions, and for orientation B, the forces were 4 Newtons in all the directions. For the calibration orientation, forces were generated in only the 4 directions shown in Figure 6.3.

Figure 6.3 shows that the endpoint force production in the calibration orientation is reasonably accurate. The greatest error in magnitude is .75 Newtons in the positive x direction. This error, however, is only 7.5 percent of the maximum commanded force, 10 Newtons. This figure also shows that there is an error in the orientation of the measured forces. When force was commanded in the positive x direction, a small force was observed in the negative y direction. It also shows a small force in the positive y direction when a force was commanded in the negative x direction. Independent tests with another two-limk manipulandum have shown similar orthogonality errors in the measured force using this particular force transducer. Thus it is possible that these orthogonality errors are due to the force transducer itself.

The test results presented for the two test orientations are for force generation tests that were performed 6 months after the force

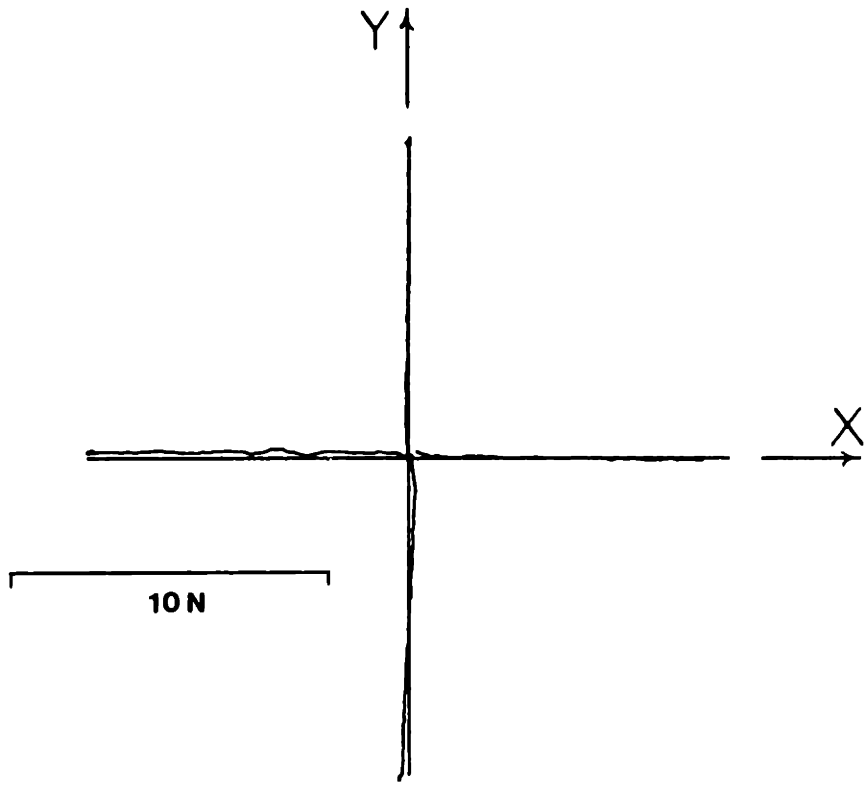


Figure 6.3 Endpoint Forces Generated for the Calibration Orientation

tests done in the calibration orientation. In these two test orientations the forces were generated in 8 directions equally spaced around a circle. Figures 6.4 and 6.5 show the results of the tests. The steady state errors in magnitude are not significantly worse than those errors in the calibration orientation, but the orthogonality errors are greater. The greatest error in magnitude in Figure 6.4 is 0.812 Newtons, which is 8 percent of the steady state commanded force (10 Newtons). In Figure 6.5 the greatest error in magnitude is 0.274 Newtons, which is 6.8 percent of the steady state commaned force (4 Newtons). In Figure 6.4 the greatest orthogonality error is 0.7 Newtons and occurs when a force is commanded in the positive x direction. The greatest orthogonality error in the calibration orientation is only 0.2 Newtons and occurs when a force is commanded in the negative y direction. From this data it is clear that the ability to control the direction of endpoint forces is degraded when the manipulator moves away from the calibration orientation. The position dependence of these orthogonality errors implies that there may be some calibration error in the mechanism.

It should be pointed out that because of the construction of the frame that supported the force transducer, the orientation of the transducer when measuring forces in the half of the workspace containing position A is 180 degrees different from its orientation when measuring forces in the other half of the workspace, which contains position B and the calibration position. This change in the transducer's orientation may account for the differences in the orthogonality errors seen in the

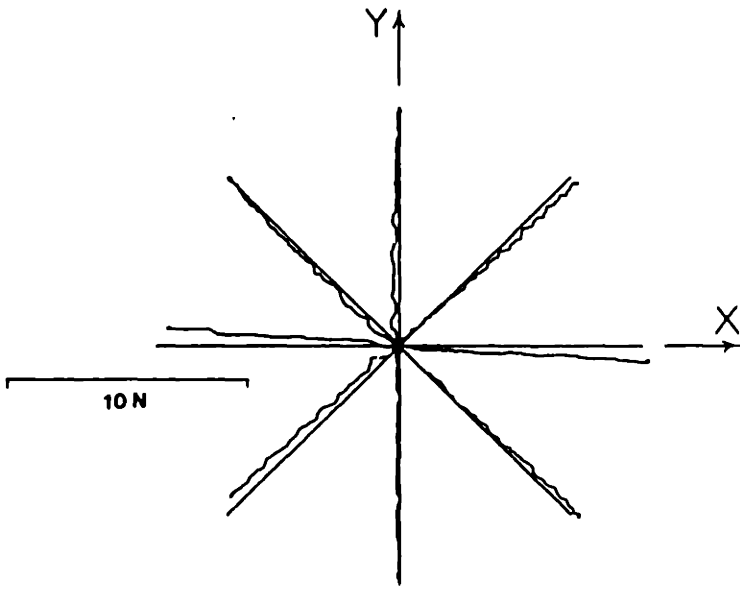


Figure 6.4 Endpoint Forces Generated for Test Orientation A

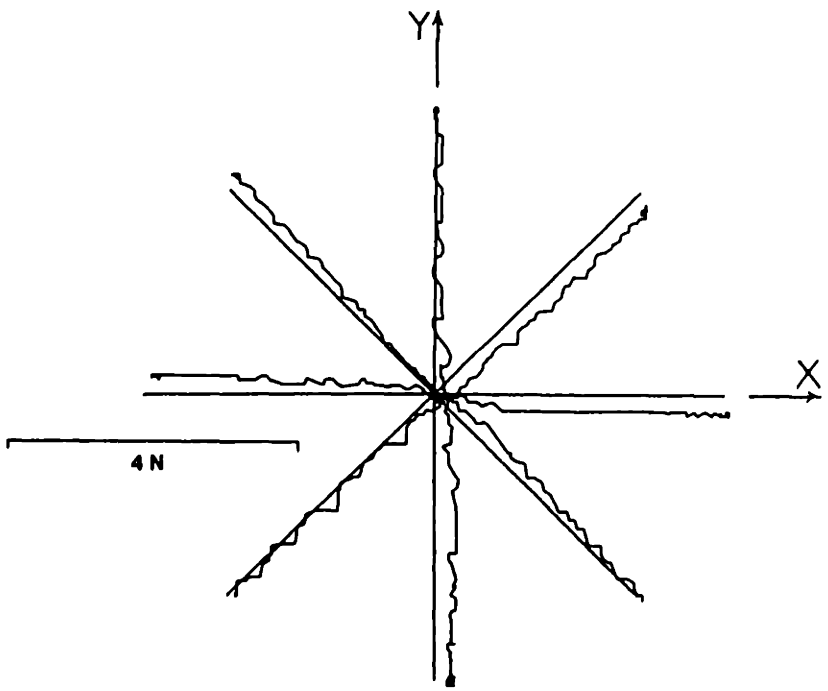


Figure 6.5 Endpoint Forces Generated for Test Orientation B

calibration position and position A. If Figure 6.4 is rotated 180 degrees then it appears as though the offset forces for forces applied in the y directions are similar for both position A and the calibration position.

For all of these experimental runs, the desired force was ramped to its maximum value in 0.32 seconds and then held at the maximum for 0.32 seconds. The sampling rate of the digital computer was 200 hertz. A time response of the actual measured force is shown in Figure 6.6. This time response is plotted for a desired force in the positive y direction and a desired force in the positive x direction. These plots represent the best and the worst of the data from this test and show that the actual force output from the transducer tracked the commanded force with at most a 26 milli-second delay. For the force generated in the y direction, the largest error in the steady state is less than 0.2 Newtons (less than 2% error). In the worst case, force generated in the x direction, the greatest offset force is 0.70 Newtons (7.0% error).

Performing a static test with the impedance controller showed that there is an apparent hysteretic loop that may be attributed to the force transducer. Without any motion of the endpoint, postion commands in the impedance controller resulted in forces at the endpoint depending on the selection of the endpoint stiffness. An isotropic stiffness of 40 Newtons/meter was imposed on the endpoint so that commanded displacements of 10.0 cm in a particular direction gave rise to forces of 4 Newtons in only that direction. The stiffness was chosen to be small so that the

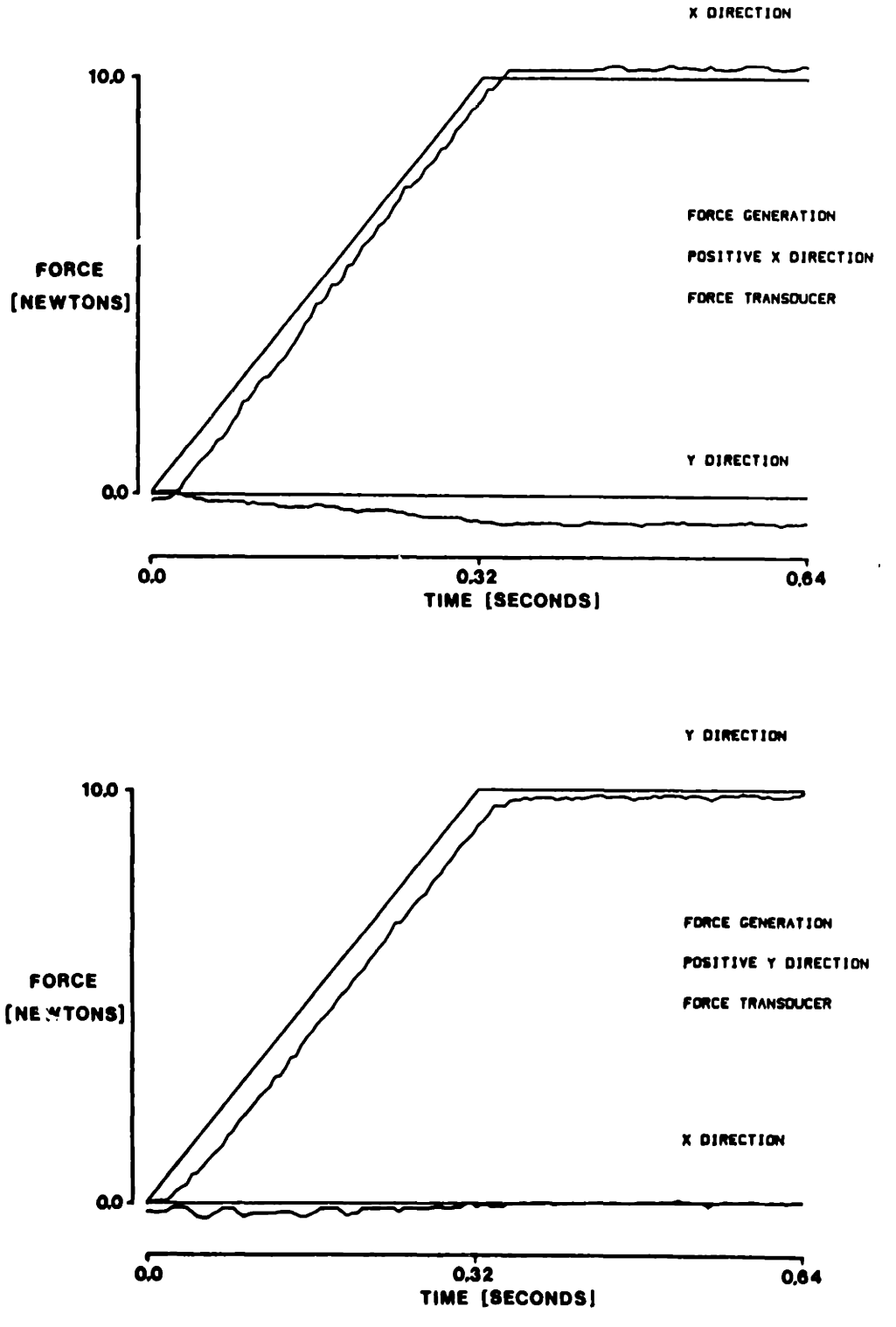


Figure 6.6 Desired and Measured Endpoint Force Versus Time

command displacement had to be large to generate a reasonable interface force. It was found that with small command displacements, the impedance control algorithm was sensitive to small displacements in the manipulator endpoint.

This static test with the impedance control algorithm involved commanding an endpoint position that moved out to the final position and then back to the starting position. This test shows the variation in transducer output for increasing and decreasing forces. The hysteretic loop arising from the loading and unloading of the transducer is shown in Figure 6.7. In all of the tested directions, this figure indicates that the increasing force applied to the transducer was different than the decreasing force applied to the transducer. It is interesting to note that when an increasing force was applied in the positive x and y directions, the decreasing forces came back to zero in the first quadrant. Likewise, when an increasing force was applied in the negative x and y directions, both decreasing forces came back to zero in the third quadrant. This symmetry may indicate that the transducer has some inherent preferred direction of displacement.

This preferred direction of displacement would account for the slight variation of the command force in each direction shown in Figure 6.7. The command force was based directly on the actual measured endpoint position of the manipulator. Any variation in the tip position would change the command force. The assumption here, is that the non-ideal behavior of the force transducer causes the variations in the

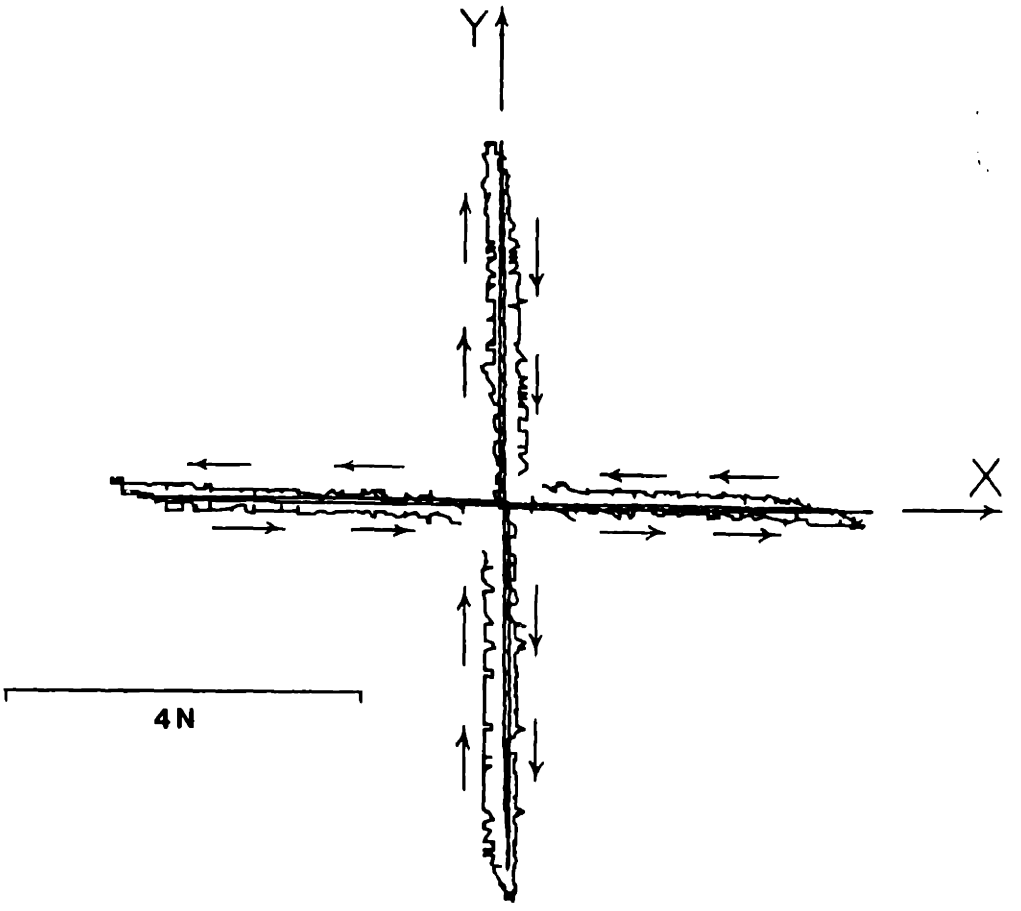


Figure 6.7 Hysteresis for Loading and Unloading the Force Transducer

command force (through actual motion of the manipulator endpoint) and not vice-versa. However, further work is needed to verify this.

Comparing Figures 6.3 and 6.4 (10 Newton forces) with Figures 6.5 and 6.7 (4 Newton forces) it is clear that the errors in the latter figures are getting close to the resolution of the measurement system. Typical errors are less than 0.2 Newtons, which is 0.17 percent of the full range of the force transducer and the computer's A/D converters.

6.2 Current Monitor Force Estimations

As described in chapter 3, the PMI servo amplifiers are equipped with current monitors that output a voltage proportional to the amount of current that is being output to the motors. Since there is not yet a direct measure of force at the manipulator endpoint, forces at the endpoint can only be measured in the static case (when the endpoint is attached to the Measurement Systems force joystick). In order to determine endpoint forces in the non-static case, an alternative approach was explored. A test was performed to see if it was possible to use the output of the current monitor to estimate the amount of force that the system was generating.

This test was performed in the static case, with the endpoint locked to the force transducer. The output from the computer was again a command voltage to the servo amplifiers to generate a desired endpoint force. The computer sampled both forces (Fx and Fy) measured by the

force transducer and the current monitor voltage at a sampling rate of 200 hz. The current monitor output voltage from the servo amplifiers is proportional to the output torque of the motors, therefore it was necessary to transform these estimated torques into estimated endpoint forces using the following transformation (from equation 2-8):

$$\underline{F} = (\underset{\sim}{J}^{T})^{-1}\underline{T} \qquad [6\text{-}1]$$

Without an anti-aliasing filter on the current monitor output there appeared to be 60 hertz noise on these measurements. Therefore a finite impulse response low-pass filter was used off-line to filter the data before plotting. The cutoff frequency was picked to be half of the frequency of the apparent noise; 30 hz.

Figure 6.4 shows an XY plot of both the desired force and the force measured by the force transducer. Figure 6.8 shows an XY plot of both the desired force and the force estimated by the current monitor output voltage for the same data presented in Figure 6.4 (orientation A). It is possible to compare the quantities measured by the force transducer and those estimated by the current monitor by looking at the ramp reponse shown in Figure 6.9. This figure is the same as Figure 6.6, but now the actual measured force trace is replaced with the current monitor estimation. These figures demonstrate that the current monitor is clearly inferior to measuring force directly, but that it could used to make reasonably accurate predictions about the direction of a force and less accurate predictions about the magnitude of a force.

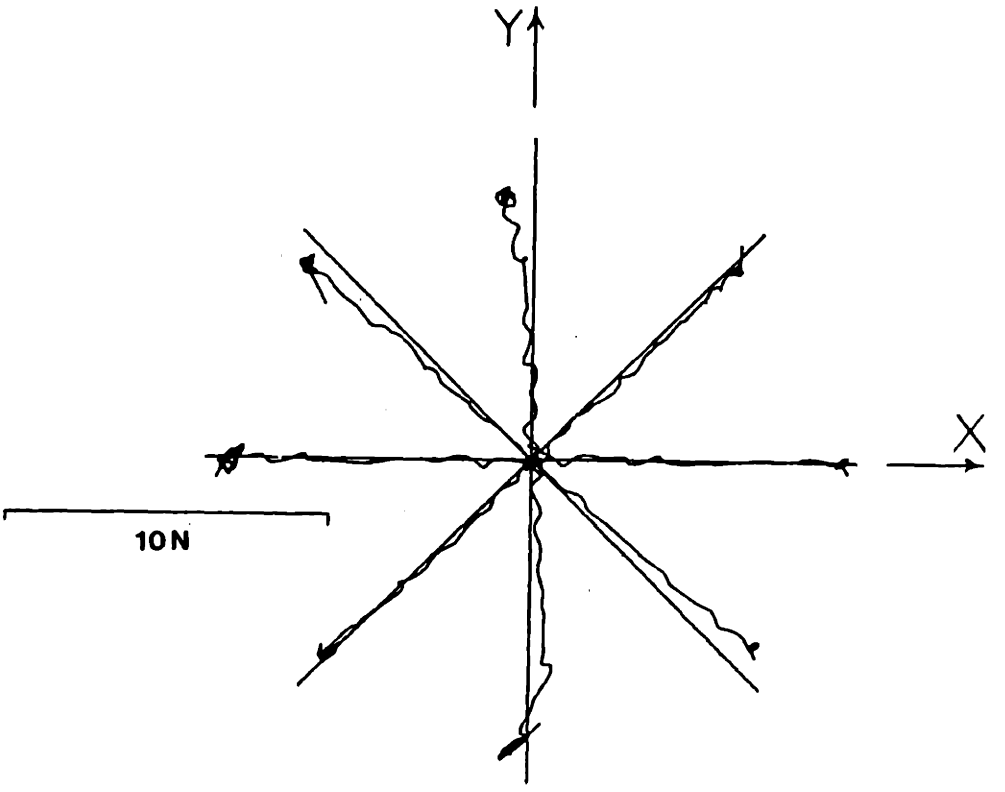


Figure 6.8 Endpoint Forces Estimated by the Current Monitor, Orientation A

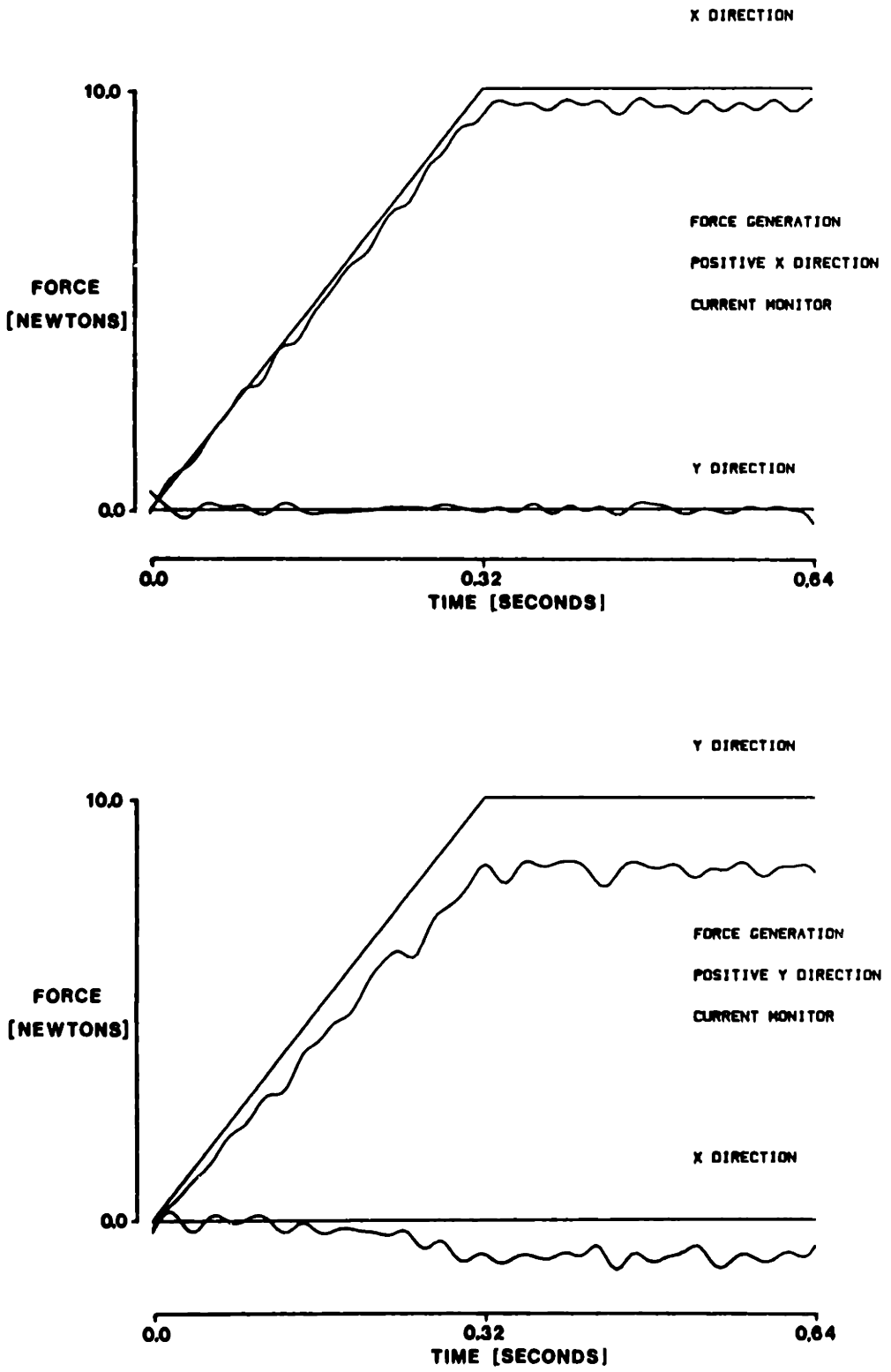


Figure 6.9 Desired and Current Monitor Estimated Endpoint Force Versus Time

6.3 Impedance Controller Evaluation

The evaluation of the impedance controller involved several steps. The first step was to evaluate the control of Cartesian endpoint stiffness in the static case. Once the ability to control static Cartesian endpoint stiffness was verified, then free movement tests were performed with only stiffness and no viscosity. The purpose of these tests were to verify the manipulator model discussed in chapter 5 and to evaluate stiffness in a dynamic and unconstrained way. To evaluate viscosity, step responses with variations in desired endpoint damping were compared. The viscosity was also verified by specifying viscosity and no stiffness and then having a subject move the manipulator around in the workspace and recording the commanded force. Finally, movements over a larger portion of the workspace (a 25 cm movement), with nominal $\underset{\sim}{K}$ and $\underset{\sim}{B}$, were made and compared to the reults from the simulation. Each one of these steps is expanded in more detail in the following sections.

6.3.1 Static Stiffness

The endpoint stiffness of the manipulator is controlled by specifying the stiffness of the endpoint in Cartesian endpoint coordinates. The goal of this section is to present to what extent it was possible to specify endpoint stiffness in the static case.

These tests were done by fixing the end of the manipulator to the Measurement Systems force joystick (as in Figure 6.1) and then command-

ing the endpoint to move to a point 10 cm from the current location. However, because the endpoint was not free to move, forces arose at the tip of the manipulator due to the endpoint stiffness and the commanded displacement. If the desired stiffness was zero in the x direction, and a move was commanded in that direction, there would be no force generated in that direction. For a desired stiffness of 40 N/m in the x direction, a 10 cm move commanded in the x direction would result in a 4 Newton force generated in that direction.

Several tests were performed in this manner. Two different linkage orientations were tested with isotropic stiffness at the endpoint. In one test orientation, tests were performed with non-zero terms in the off-diagonal elements of the stiffness matrix. The linkage orientations for the isotropic stiffness are shown in Figure 6.10. More exhaustive tests of different locations are needed, but are postponed for further work.

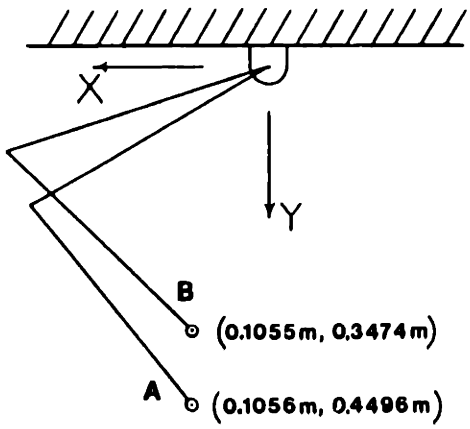
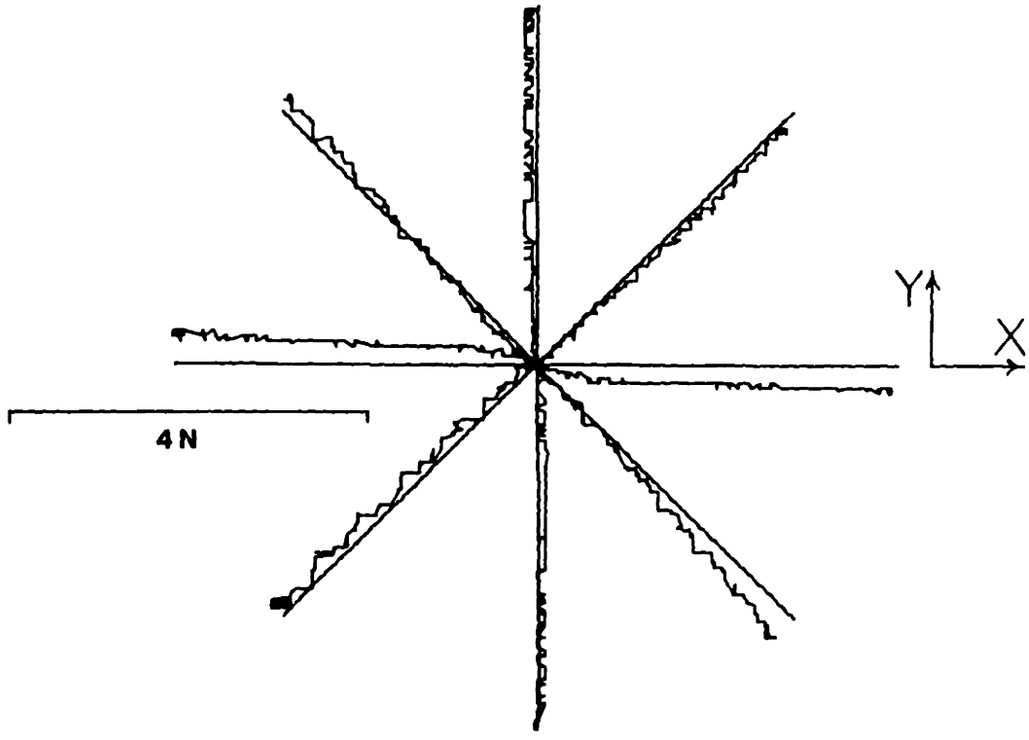


Figure 6.10 Manipulator Orientations for Endpoint Stiffness Evaluation

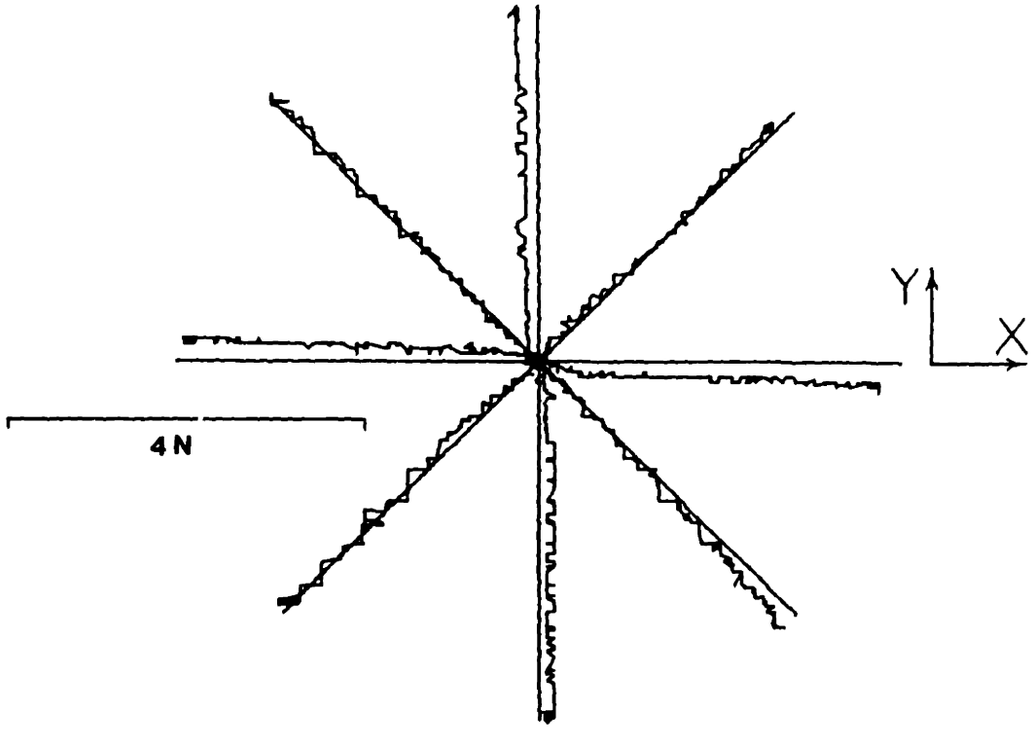
Orientation B is the orientation in which the non-isotropic stiffnesses were tested. Figure 6.11 shows the results for the isotropic stiffness tests for both orientations, A and B. It is clear from these plots that the stiffness appears to be the same in both orientations. These plots show the Cartesian force generated by the impedance control algorithm and the actual force measured by the transducer. The Cartesian forces generated by the impedance controller are also referred to as the expected interface forces. These forces were kept small (4 Newtons) to minimize any displacement of the manipulator endpoint.

The tests in Figure 6.11 were performed by specifying an isotropic stiffness of 40 N/m and then commanding the 10 cm displacement in 8 directions equi-distant around a cirlce. The same position commands were also used in the evaluation of the non-isotropic stiffness matirx. In the isotropic stiffness case the tips interface force vectors were expected to lie on a circle, but in the non-isotropic stiffness case, the tips of the interface force vectors lie on an ellipse for the same position commands. The selection of diagonal and off-diagonal terms determine the shape and orientation of this force ellipse.

Figure 6.12 shows the results from the tests that were performed with the non-isotropic stiffness. The stiffness matrix that produced the results in Figure 6.12a had diagonal stiffness terms of 40 N/m and off-diagonal terms of 20 N/m. As would be expected, the orientation of the major axis of the resulting force ellipse is at 45 degrees. Likewise, Figure 6.12b shows results for a stiffness matrix with diago-



Orientation A



Orientation B

Figure 6.11 Isotropic Stiffness Test Results

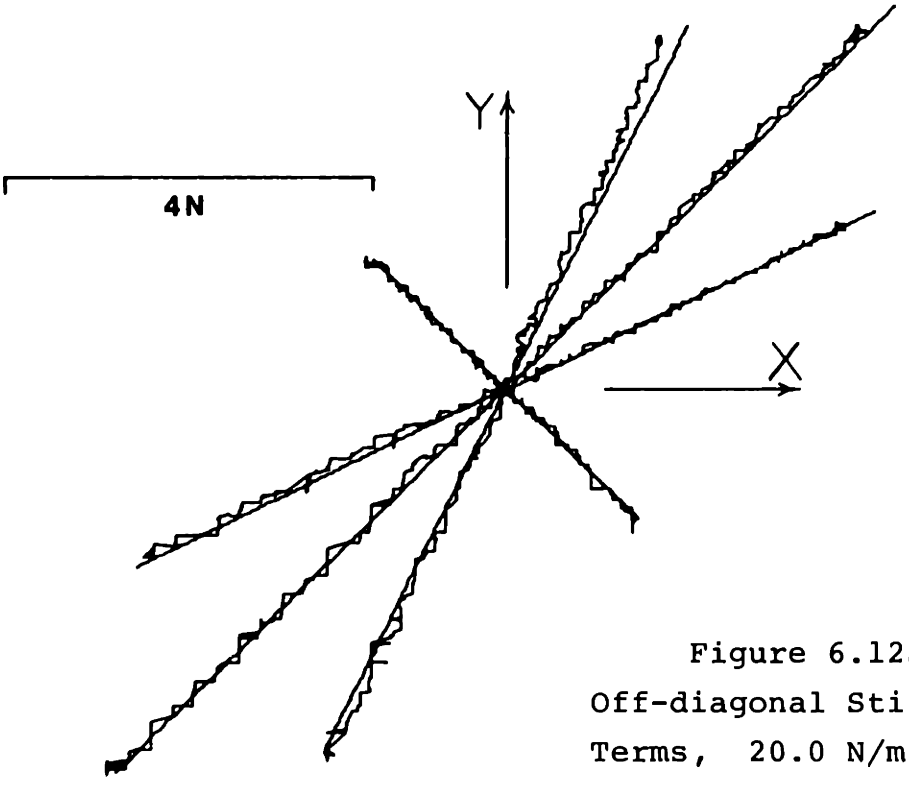


Figure 6.12a
Off-diagonal Stiffness
Terms, 20.0 N/m

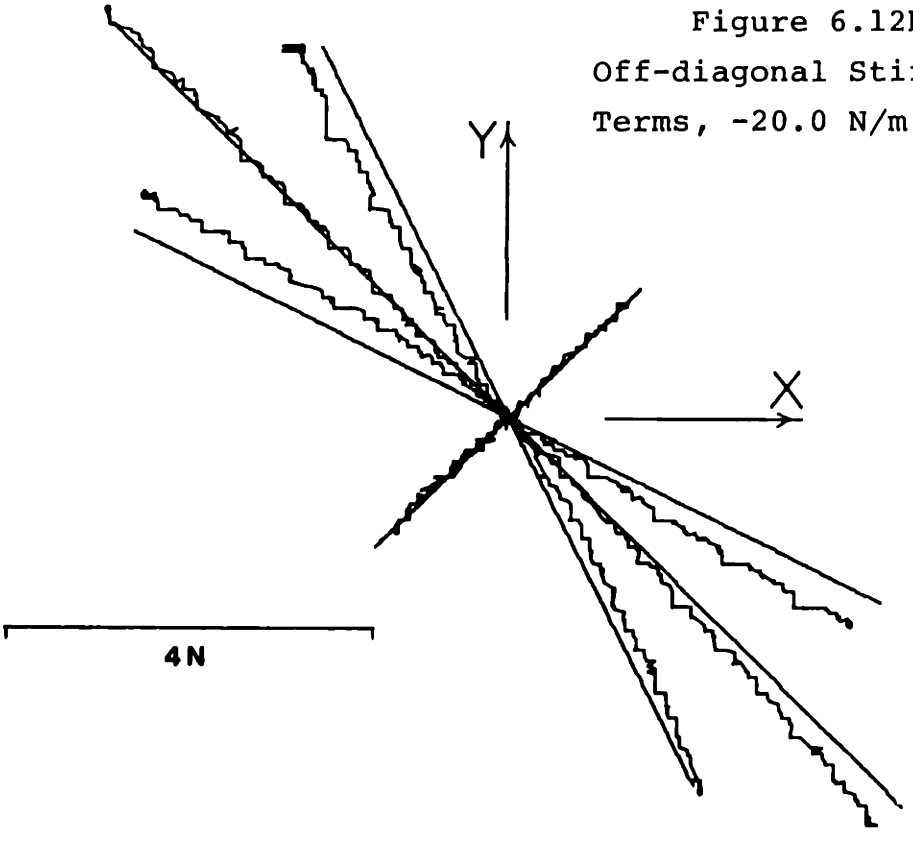


Figure 6.12b
Off-diagonal Stiffness
Terms, -20.0 N/m

Figure 6.12 Non-isotropic Stiffness Test Results

nal stiffness terms of 40 N/m and off-diagonal terms of -20 N/m. And, as would be expected, the orientation the major axis of the resulting force ellipse is at 135 degrees (a 90 degree shift from the orientation of the ellipse in Figure 6.12a).

Clearly, the impedance controller has the ability to control stiffness in this constrained case. Deviations in the transducer output, from the expected interface force, are attributed to either the non-ideal behavior of the force transducer or calibration errors in the mechanism.

6.3.2 Dynamic Stiffness with No Damping

While it is possible to control the Cartesian endpoint stiffness, it is not yet possible to control the endpoint inertance (without force feedback). Thus, as the manipulator moves around in the workspace and the relative angles between the links changes, the interia matrix from equation 5-2 changes:

$$\underset{\sim}{I}_{\theta} = \begin{bmatrix} J_1 + \ell_1^2 m_2 & \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) \\ \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) & J_2 \end{bmatrix} \qquad [6\text{-}2]$$

These tests involve computing the endpoint inertia matrix, $\underset{\sim}{I}e$, (in Cartesian endpoint coordinates) from equation 6-2 for the location x = .1016 meters and y = 0.4492 meters. The endpoint inertia matrix is computed with the following relation:

$$(\underset{\sim}{I}_e)^{-1} = \underset{\sim}{J}(\underset{\sim}{I}_{\theta})^{-1}\underset{\sim}{J}^T \qquad [6\text{-}3]$$

An endpoint stiffness matrix is chosen based on this endpoint inertia matrix and undamped oscillations for step in position are observed about the point of interest. These tests not only confirm the ability to control the endpoint stiffness, but also verify the inertial aspects of the manipulator model.

The stiffness matrix was computed as a factor of the computed endpoint inertia matrix to align the stiffness ellipse with the inertia ellipse. The stiffness and inertance ellipses are ellipses that are determined by the eigenvalues and eigenvectors of each matrix. Because the stiffness matrix is a factor of the endpoint inertia matrix, the shape and orientation of the stiffness ellipse should be the same as that of the inertia ellipse. Thus, for this location, the frequency of the all the oscillations in any direction should be given by:

$$\omega_n = \sqrt{\frac{K_{ij}}{m_{ij}}} \qquad [6\text{-}4]$$

While it is possible to compute a different natural frequency for each set of stiffness and inertia elements in each matrix, the task of determining a stiffness matrix was simplified by choosing a single value for the natural frequency and then computing each element in the stiffness matrix with the following relation:

$$K_{ij} = \omega_n^2 \, m_{ij} \qquad [6\text{-}5]$$

Figures 6.13 and 6.14 show oscillations that were produced in two different cases near the desired point (x = .1016 meters and y = 0.4492

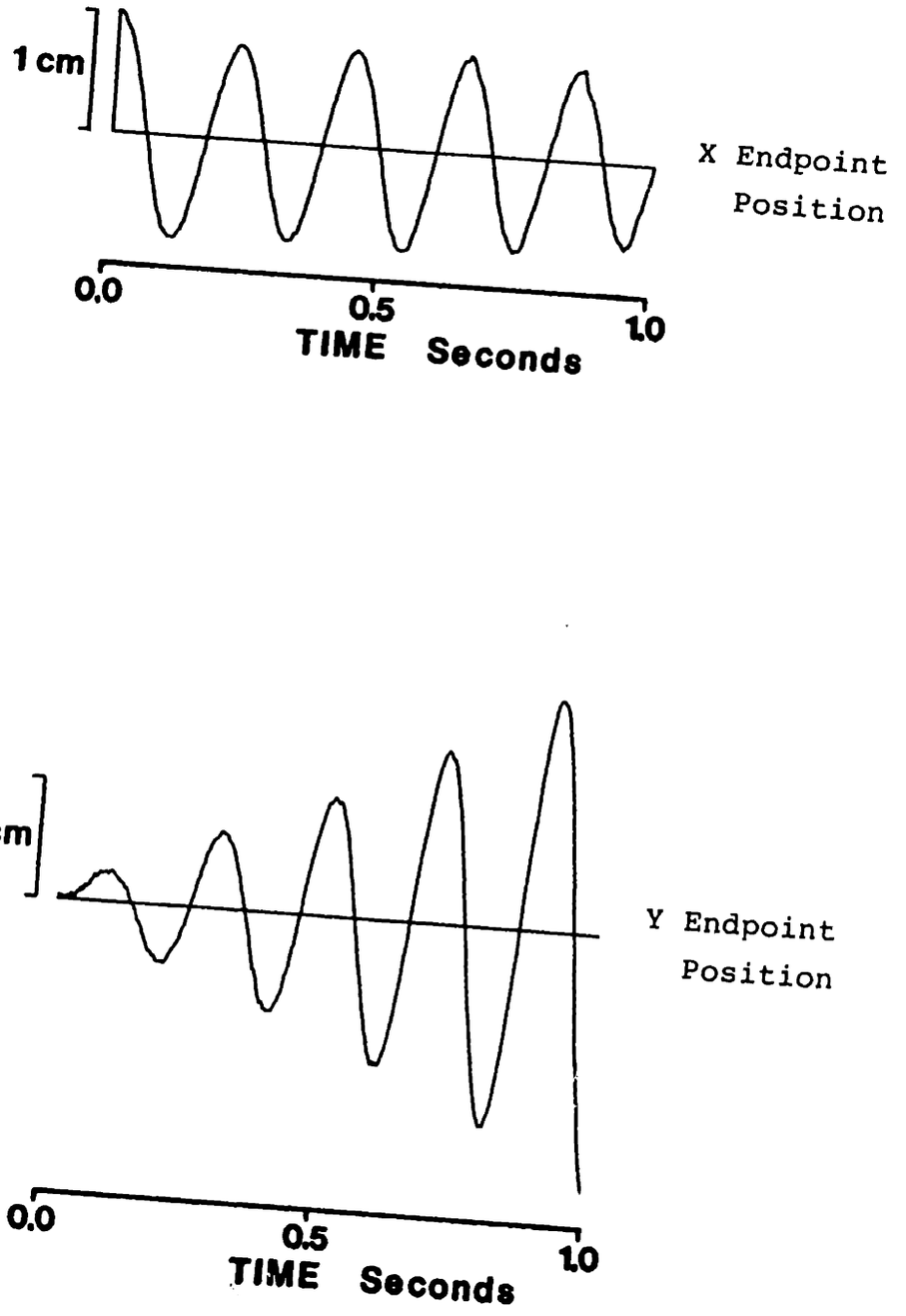


Figure 6.13 Undamped Step Response in the Negative X Direction

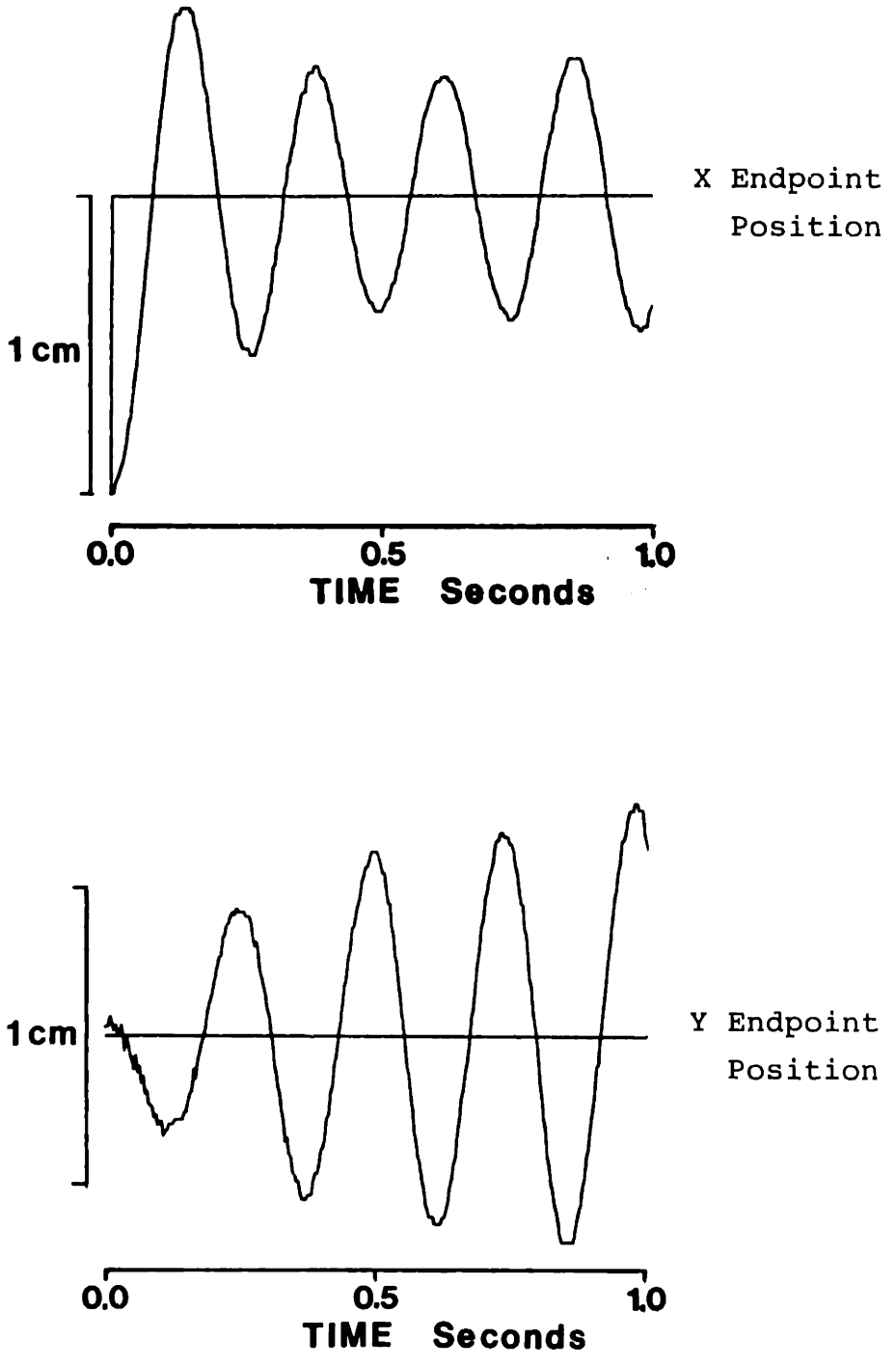


Figure 6.14 Undamped Step Response in the Positive X Direction

meters) and for two different desired frequencies. The oscillations were initiated by commanding a step of 1 cm from this endpoint location. These position commands were steps in the positive and negative x direction. In Figure 6.13 the desired stiffness matrix was:

$$\underset{\sim}{K}_e = \begin{bmatrix} 441.42 & -299.44 \\ -299.44 & 580.26 \end{bmatrix}$$

Thus the natural frequency of the oscillations should be 5.03 hertz. Computations from the plot show that the frequency of oscillations is 5.04 hertz (a .2 percent error). In Figure 6.14 the stiffness matrix was:

$$\underset{\sim}{K}_e = \begin{bmatrix} 286.92 & -194.64 \\ -194.64 & 377.17 \end{bmatrix}$$

Thus the natural frequency of the oscillations should be 4.06 hertz. Computations from the plot show that the frequency of oscillations is 4.092 hertz (a .79 percent error) Although these tests are not conclusive, they do show that it is possible to specify a natural frequency based on the model of the endpoint inertance and the commanded endpoint siffness.

Without any velocity feedback ($\underset{\sim}{B} = 0\underset{\sim}{I}$) the oscillations in this test were unstable. As the low negative damping ratio (-0.022) for figure 6.14 indicates, the system is barely unstable. This instability may be attributed to the sampled data nature of the controller. With zero velocity feedback the poles for this second order system will be stable, but very close to the imaginary axis (they will not be on the imaginary axis due to friction in the system). A delay due to the finite sampling

rate of the digital computer can be simplistically modelled as a pole on the real axis of the s-plane plot. The addition of this pole increases the order of this system and forces the originally stable second order poles into the unstable region. With the small negative damping ratio there is no noticeable change in the damped natural frequencies and hence the frequency data presented above is valid even for this unstable system. A detailed analysis of this instability is postponed for further work.

6.3.3 Dynamic Stiffness and Damping

A demonstration of the ability to control endpoint damping is shown in Figure 6.15. For various values of damping, these Figures show the time reponse of the endpoint position to a commanded step change in position. In this case the commanded step changes in postion were 1 cm in the negative x direction and 1 cm in the positive y direction. The starting position of the endpoint was always x = 0.1016 meters and y = 0.4492 meters. For these tests, the stiffness was chosen so that the natural frequency of the oscillations at the endpoint would be 4.06 hertz and the damping matrix, $\underline{B}$, was chosen to be non-zero and isotropic.

From Figure 6.15 it is clear that increasing B had an effect on the response of the system. This figure shows that as the damping increased, the amplitude and the number of oscillations decreased. From the peak value of the first overshoot it was possible to compute a value

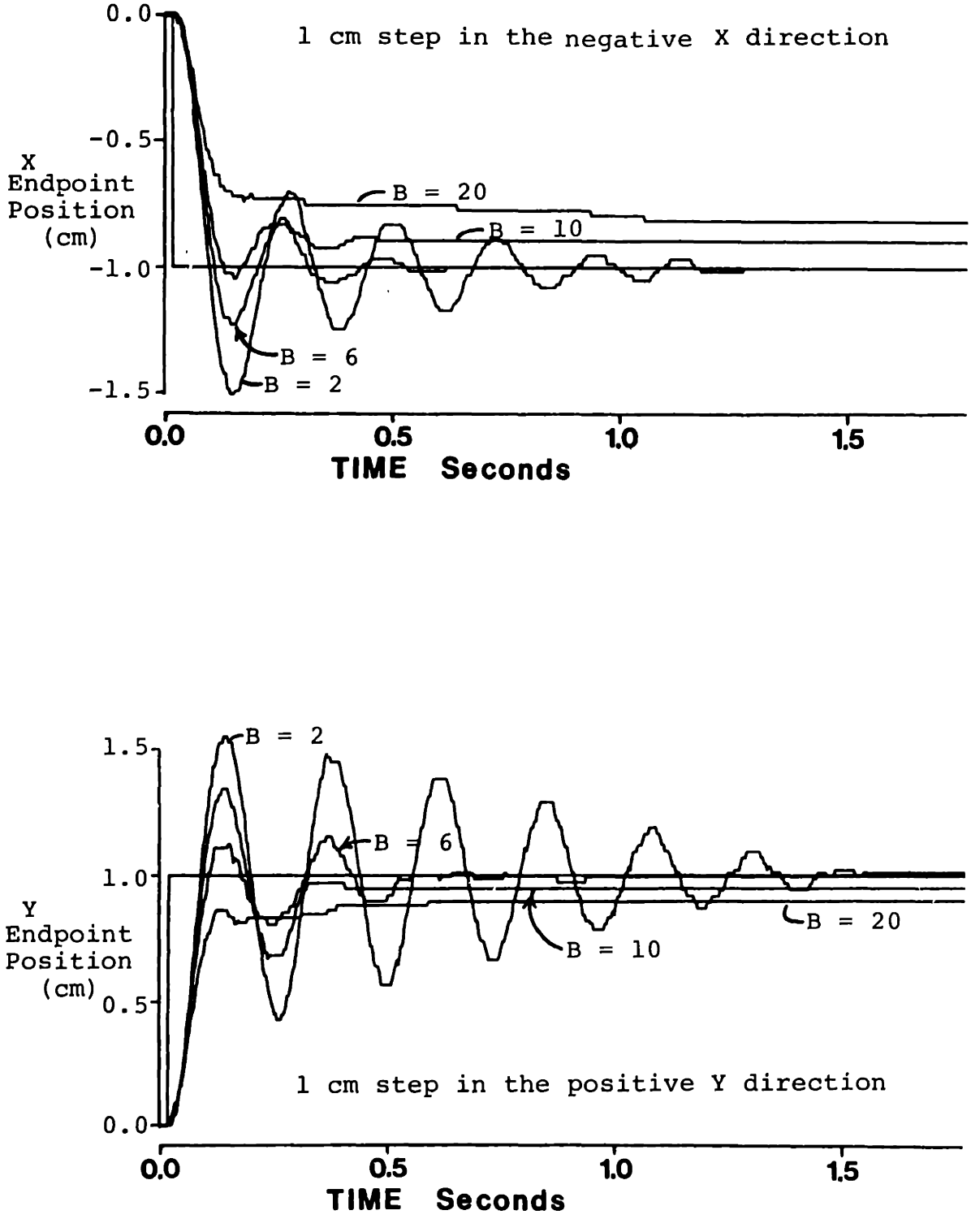


Figure 6.15 Step Response of the Endpoint for Various Values of Damping

TABLE 6-1 Theoretical and Measured Values of Damping Ratios

1 cm movement in the positive y direction

B	ζ cal	ζ meas	%error
2	.125	.188	50.0
6	.373	.321	13.9
10	.622	.555	10.8
20	1.235	---	----

1 cm movement in the positive y direction

B	ζ cal	ζ meas	%error
2	.153	.216	41.2
6	.459	.426	7.2
10	.766	.714	6.7
20	1.531	---	----

for the effective damping ratio of the endpoint of the manipulator [19]. Table 6-1 lists the measured damping ratio and the the theoretical damping ratio for each value of endpoint damping, $\underset{\sim}{B}$. Because the measured damping ratios were computed from the peak overshoot it was not possible to estimate damping ratios for $\underset{\sim}{B} = 20\underset{\sim}{I}$. For low values of endpoint damping the error in the measured damping ratio and the theoretical damping ratio is large (as great as 50 percent). With a lightly damped system, the motion of the manipulator endpoint may be large enough to cause errors in the effective endpoint mass. It is also possible that the effects of static friction (which is not included in the theoretical prediction of the damping ratio) become significant for small damping (the measured damping ratio is 50 percent greater than the theoretical damping ratio). For larger values of endpoint damping there is reasonable agreement between the theoretical (computed) and measured damping ratios. The largest error is 13.9 percent ($\underset{\sim}{B} = 6\underset{\sim}{I}$ in the positive y direction) and the smallest error is 6.7 percent ($\underset{\sim}{B} = 10\underset{\sim}{I}$ in the positive y direction). While these errors may be due to errors in estimating the effective endpoint mass, they may also be due to the phase lag of the filter used with the velocity feedback signal.

For large values of the damping matrix, $\underset{\sim}{B}$, this figure shows that the endpoint of the manipulator came to rest a significant distance from its desired equilibrium point. This is complicated by noting that the distance from the equilibrium point is clearly a function of the magnitude of the damping matrix. The greater the damping matrix, the further from the equilibrium point the manipulator endpoint came to rest. The

cause of this problem is clearly demonstrated by plotting the absolute joint angles of the system during the movement. Figure 6.16 shows plots of the joint angles for the both directions of movement when $\underset{\sim}{B} = 10\underset{\sim}{I}$ and $\underset{\sim}{B} = 20\underset{\sim}{I}$. In both cases it was the outer link that did not reach its equilibrium position. This clearly shows that there is significant static friction in the four-bar linkage. While both motors have some internal friction, it was only the outer link that experienced the friction of the motor and the added friction in the four-bar linkage.

When $\underset{\sim}{B} = 20\underset{\sim}{I}$ the offset in the outer link is 0.00704 rad. The torque needed to drive the outer link to its final position was found to be 0.1089 N-m. This torque is 1 percent of the total torque available from the torque motors (11.178 N-m), a very small torque.

6.3.4 Damping with No Stiffness

Damping was also evaluated for a large motion without any stiffness. Figure 6.17 shows data for a move across the workspace that was performed with only viscosity commaned at the endpoint. For this experiment the handle at the manipulator endpoint was grasped by the subject and moved from one point in the workspace to another after the real time control loop was started. The endpoint trajectory for this move is shown in Figure 6.18. The damping matrix that was used was:

$$\underset{\sim}{B}_e = \begin{bmatrix} 15 & 0 \\ 0 & 15 \end{bmatrix}$$

The plot shows both the time history of the endpoint velocity and the

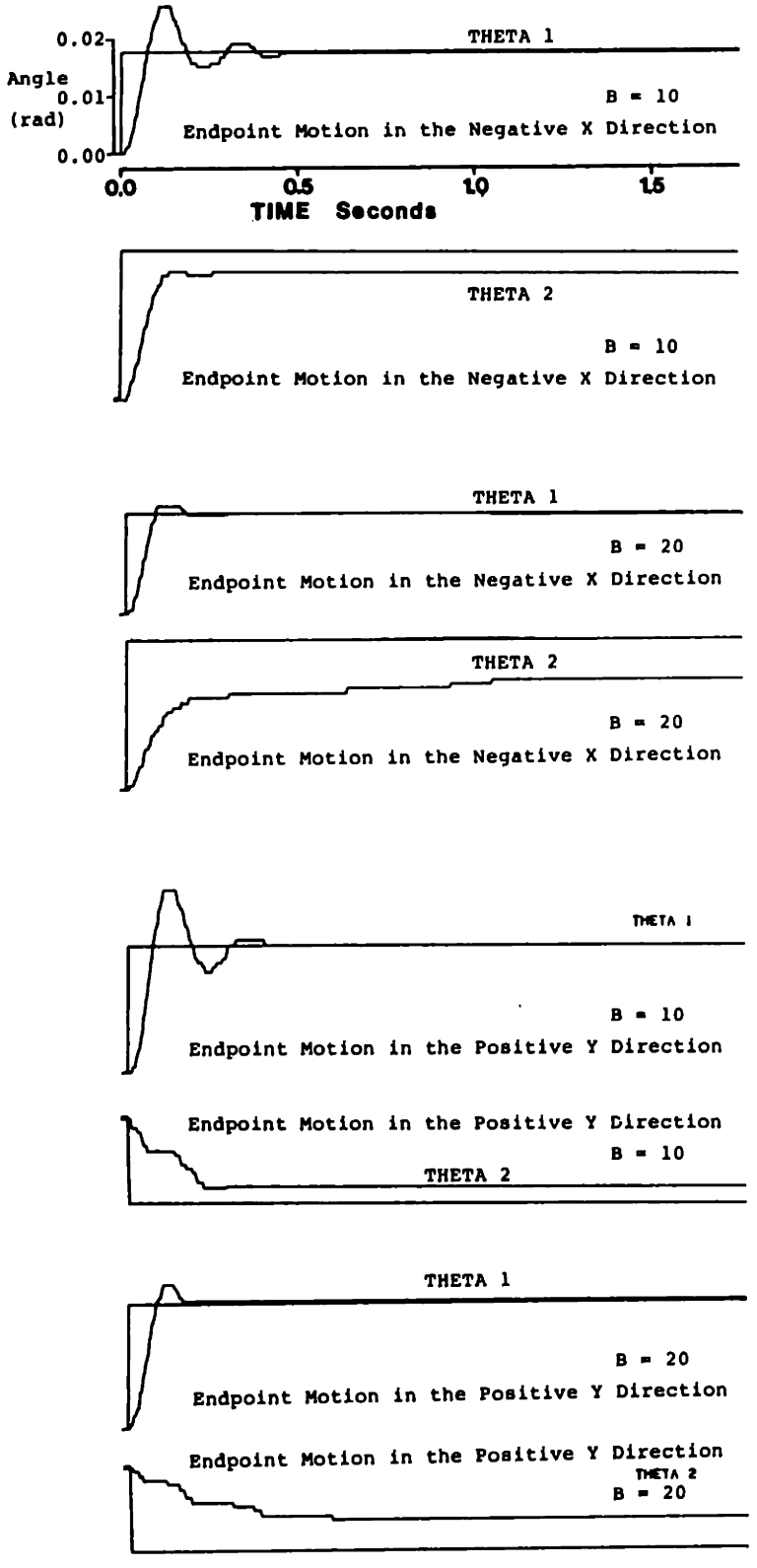


Figure 6.16 Joint Angle Plots for Step Responses in Figure 6.15 (B = 10 and B = 20)

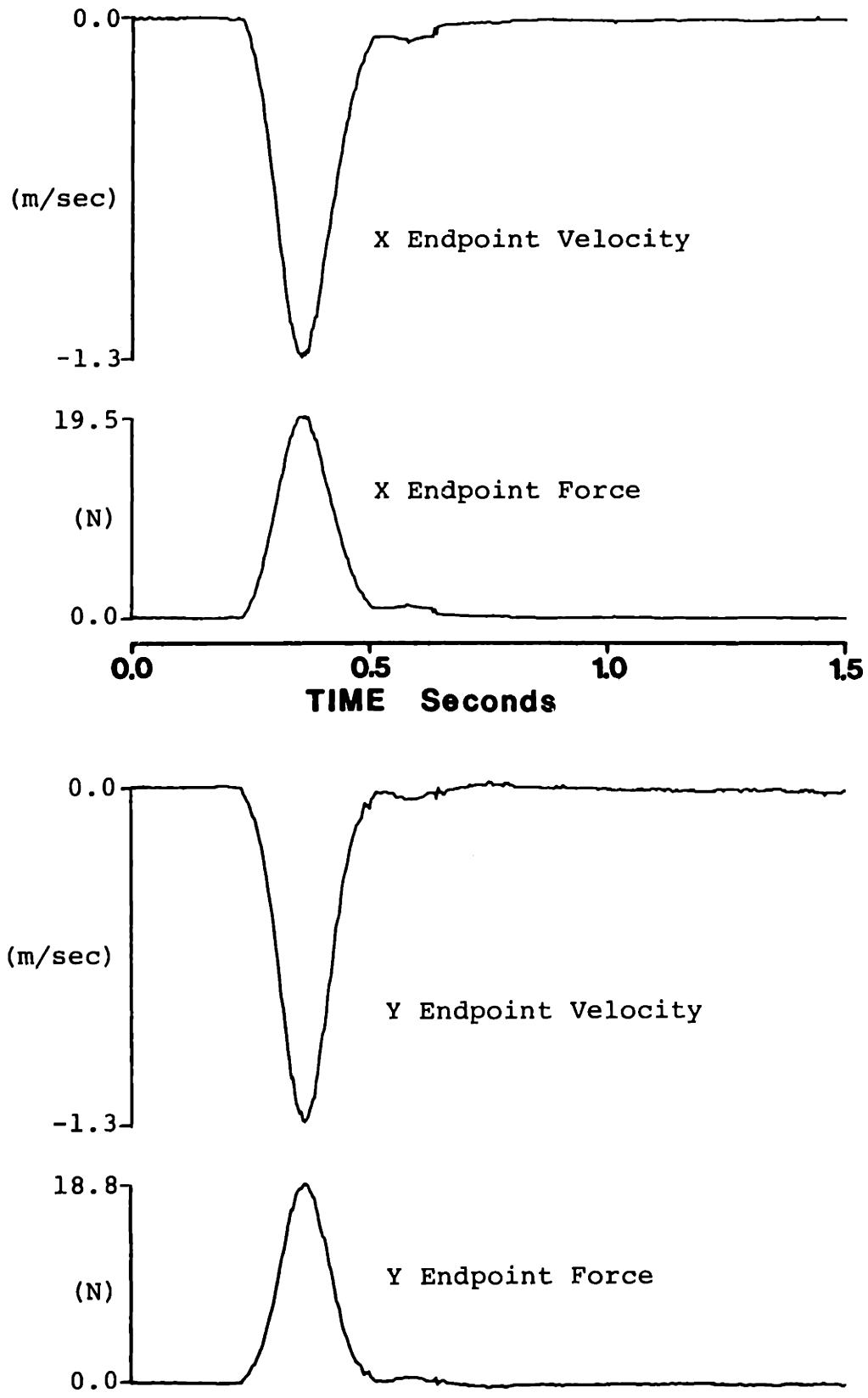


Figure 6.17 Movement with Isotropic Viscous Endpoint Load

commanded endpoint force generated in reponse to this velocity. The time history of velocity and the commanded force both look very smooth and appear to be mirror images of each other (as would be expected).

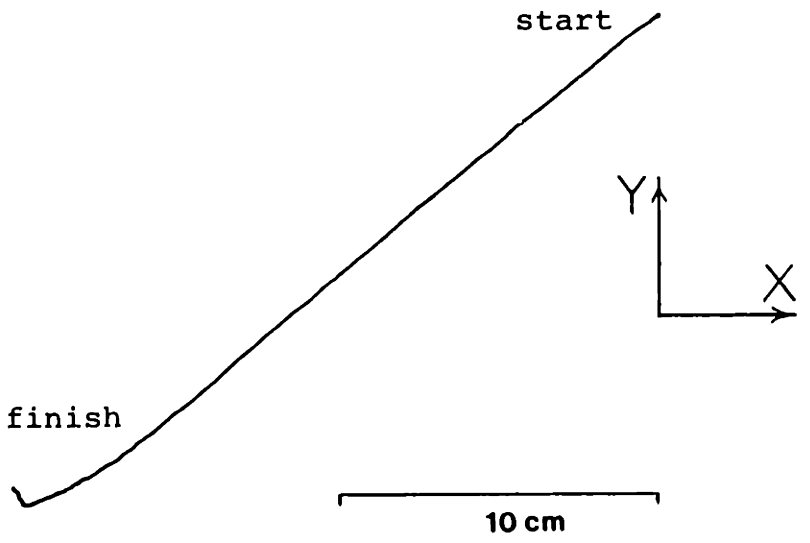


Figure 6.18 XY Plot of Endpoint Trajectory for Isotropic Viscous Load

6.3.5 Large Amplitude Motion

Figure 6.19 shows an XY plot of virtual and actual position of the manipulator endpoint as it moved from one point in the workspace to another. The virtual position was ramped to the final position in 0.350 seconds and the distance traveled was 25.4 centimeters. The ramp was comprised of 100 points and the total distance traveled was divided into 100 equal segments.

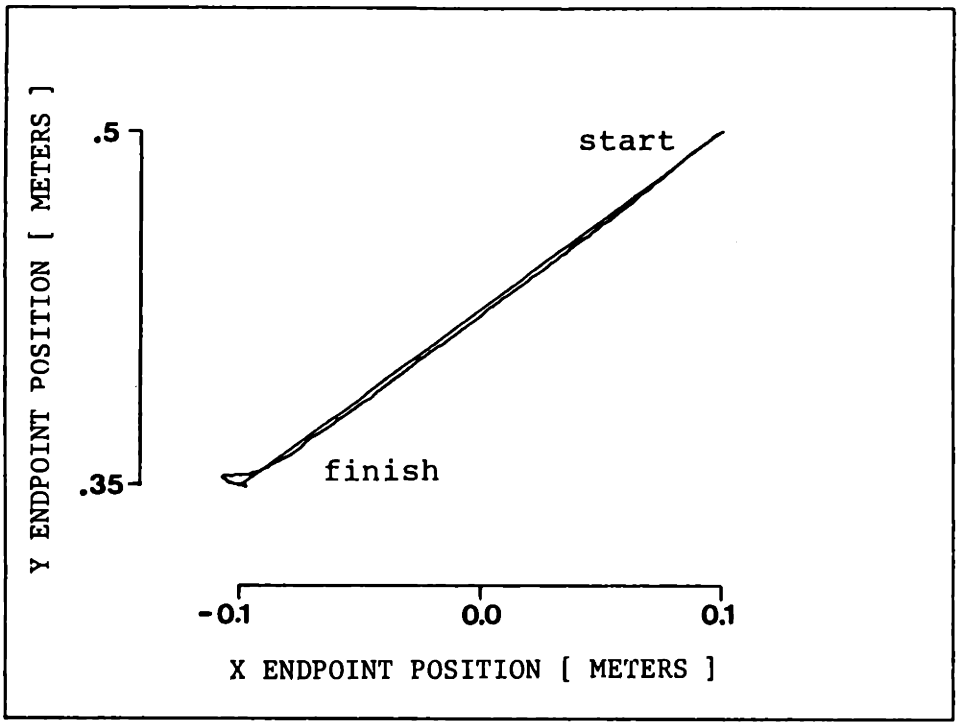


Figure 6.19 Endpoint Trajectory for Large Amplitude Movement

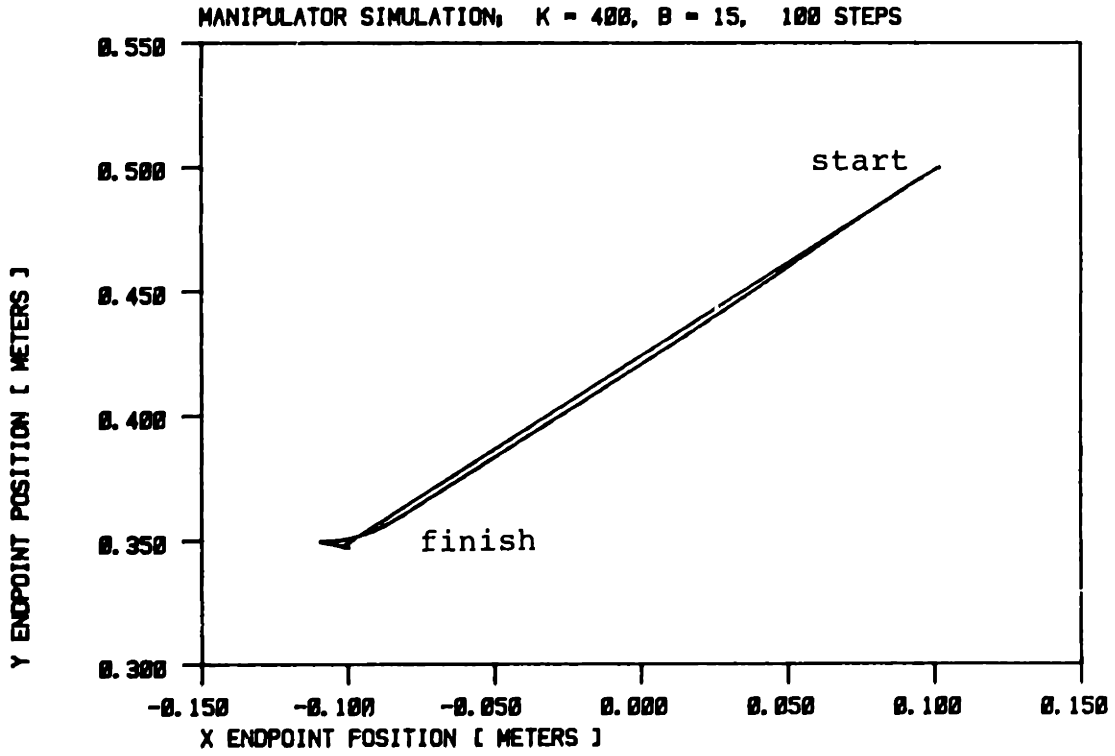


Figure 6.20 Simulated Endpoint Trajectory of the Large Amplitude Movement

The desired endpoint stiffness and damping were:

$$\underset{\sim}{K}_e = \begin{bmatrix} 400 & 0 \\ 0 & 400 \end{bmatrix}$$

$$\underset{\sim}{B}_e = \begin{bmatrix} 15 & 0 \\ 0 & 15 \end{bmatrix}$$

Figure 6.20 shows a simulation of the same move using the model developed in chapter 5 with an EQSIM subroutine in DYSYS. The only difference between the simulation and the actual hardware was the sampling rate. The loop time in impedance control loop was .0035 seconds and the time step in the simulation was dt = .00175 seconds. In order to keep the number of steps in the ramp the same, the position in the simulation was only updated every other time step.

Figure 6.21 shows the time histories of the data from the simulation and the data from the actual manipulator. Not only is the data qualitatively consistent, but the quantitative values are nearly the same for the simulation and the manipulator. The same overshoots in the position traces occur in both sets of data and the velocity profiles are nearly the same for the manipulator and the simulation. This comparison clearly confirms the validity of the model and the performance of the impedance controller.

6.3.6 Parameter Limits for Large Amplitude Motion

The physical limits for $\underset{\sim}{K}_e$ and $\underset{\sim}{B}_e$ were qualitatively evaluated for

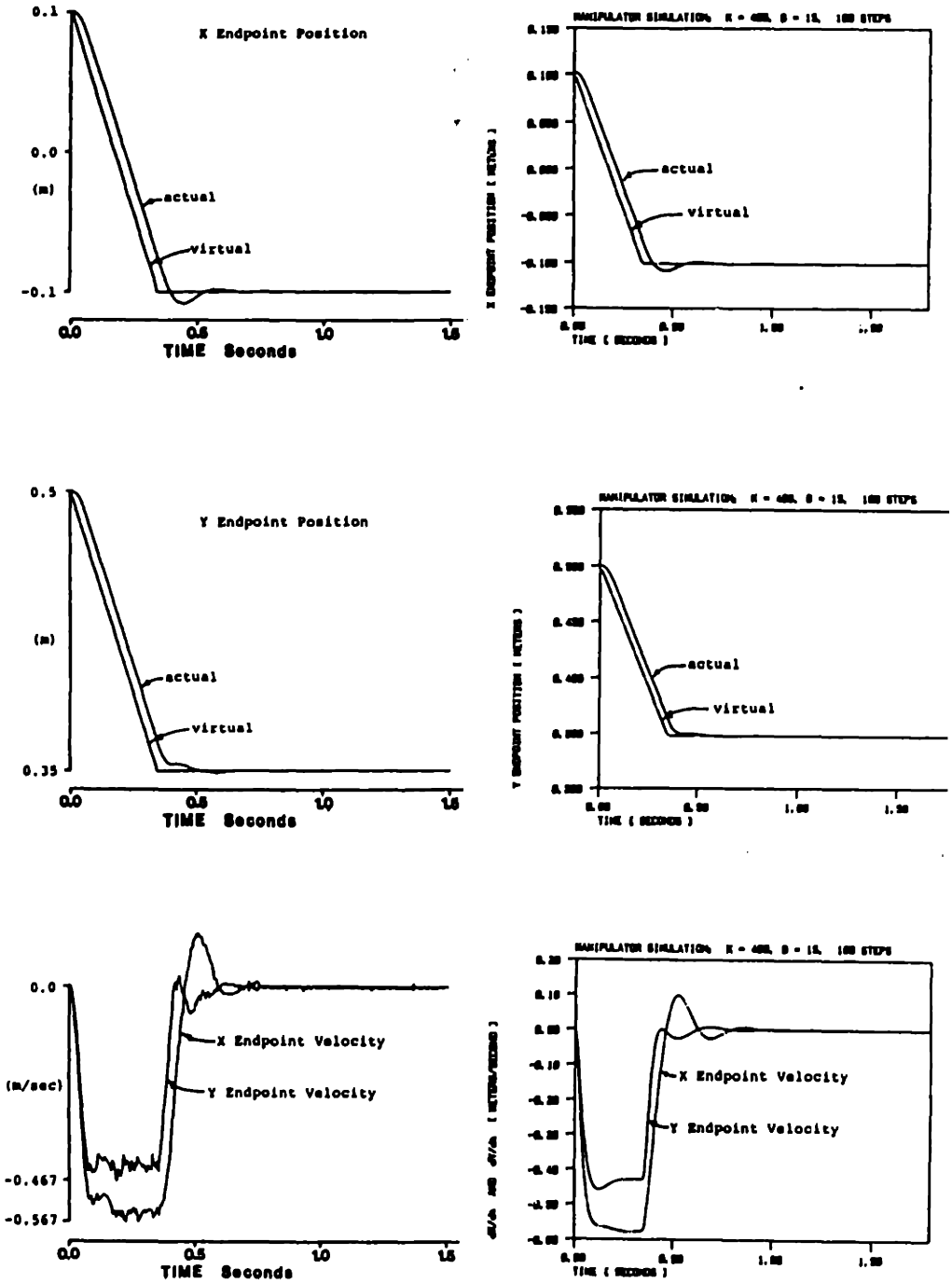


Figure 6.21 Time Response of Actual Manipulator and Simulation

large amplitude motion. These experiments were performed by setting the two parameters ($\underset{\sim}{K}$ and $\underset{\sim}{B}$) and then commanding moves of the manipulator endpoint between four different points in the workspace (at least 20 cm apart). The four points in the workspace were always kept constant and the parameters $\underset{\sim}{K}$ and $\underset{\sim}{B}$ were increased until any one of the moves caused a high frequency instability of approximately 400 hertz. The values of $\underset{\sim}{K}$ and $\underset{\sim}{B}$ were tested in three different ways; holding $\underset{\sim}{B}$ constant and increasing $\underset{\sim}{K}$, holding $\underset{\sim}{K}$ constant and increasing $\underset{\sim}{B}$, and finally increasing both of the parameters at the same time. The qualitative limits for each of these cases are listed in table 6-2. The larger the stiffness, the smaller the allowable damping and vice-versa.

TABLE 6-2 Qualitative Maximums Of Endpoint Stiffness And Viscosity

Increasing K, holding B constant:	
K = 1400 N/m	B = 15 N-sec/m
Increasing B, holding K constant:	
K = 400 N/m	B = 28 N-sec/m
Increasing K and B simultaneously:	
K = 1000 N/m	B = 20 N-sec/m

6.4 Human Subject Experiments

Several tests with human subjects are presented here to demonstrate some of the uses that the apparatus will have in studying the control of

human movement. These tests show that the apparatus is capable of performing the stiffness experiments (described in chapter 1) as well as some new experiments that cannot be done with the apparatus that was originally used to perform the stiffness experiments [18]. These tests are by no means definitve experiments, but are presented here to demonstrate the type of experiments that might be done with the apparatus.

6.4.1 Stiffness Experiment

Figure 6.22 shows a typical test of a subjects endpoint stiffness in a single direction. In this test, the endpoint of the manipulator was commanded to move from its starting point to a position 1 cm in the negative x direction. It was assumed that the subject's desired equilibrium point was still at the starting point and that the interaction force arises as the subject tried to maintain this original equilibrium position. This interaction force and the displacement of the subject's hand from its equilibrium position combine to give an endpoint stiffness of 352 N/m in that direction. This value of stiffness for this isolated experiment is within the range of values reported in previous studies [18].

6.4.2 Instability Compensation

The apparatus that was used for the previous set of stiffness experiments had the ability to change the apparent endpoint stiffness, but did not have the ability to modify the damping matrix as is possible

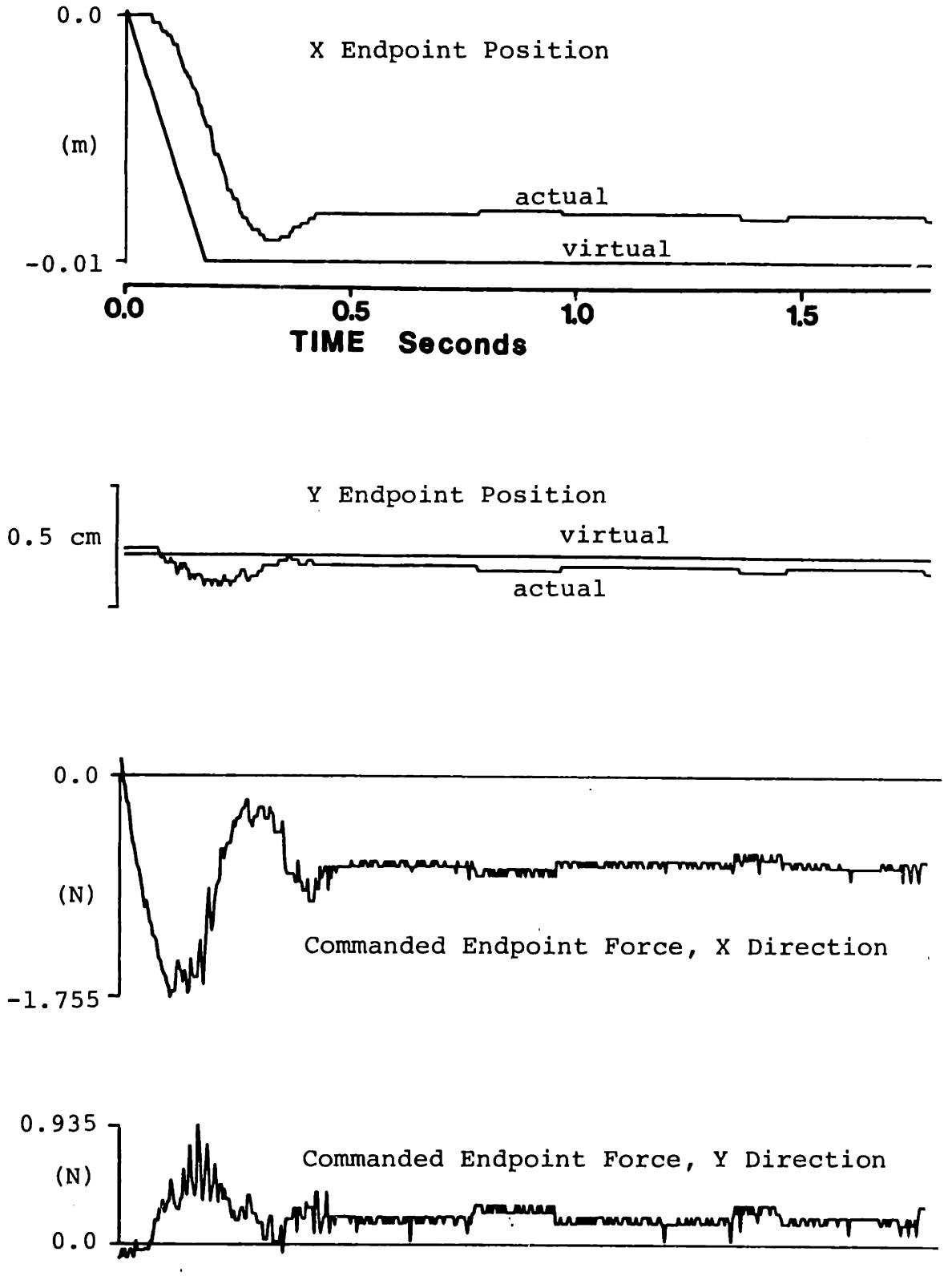


Figure 6.22 Data for the Stiffness Experiment with a Human Subject

with the impedance control strategy used in this work. It is this ability, to modify the endpoint damping, that will be exploited in the next experiment.

This experiment was performed to see if it is possible to determine some the dynamic parameters of the moving human arm. The parameters of interest here are the effective endpoint stiffness, damping and inertia. To date, stiffness has only been sucessfully measured in the static case as described above and the effective endpoint damping has always been an elusive quantity. While the mass and inertance of the limb segments will not change, it is possible for the subject to modify his endpoint inertance by movement of the wrist [12]. None of these parameters has yet been determined under dynamic conditions.

In this experiment, a dynamic condition was created by intentionally destabilizing the the manipulator. Without the subject grasping the handle of manipulator, the oscillations of the manipulator would have grown rapidly beyond the limits of the manipulator workspace. This instability was generated by commanding negative damping (positive velocity feedback), at the manipulator endpoint, which made the real part of the complex pole pair (representing this second order system) positive. Estimates of the negative damping ratio of the manipulator with this control strategy range from -.24 for motion in the direction of greatest inertia, to -.98 for motion in the orthogonal direction.

These preliminary experiments were performed on two subjects. For

one subject, A, only one value of isotropic endpoint stiffness in the manipulandum was used ($\underset{\sim}{K}_e = 600\underset{\sim}{I}$), but for the other subject, B, four different values were used. These stiffness values ranged from $\underset{\sim}{K} = 200\underset{\sim}{I}$ [N/m] to $\underset{\sim}{K} = 800\underset{\sim}{I}$ [N/m] at increments of 200 [N/m]. For both subjects the desired endpoint damping matrix was always $\underset{\sim}{B}_e = -15\underset{\sim}{I}$.

In this experiment, the subject grasped the handle of the manipulator and was instructed to move the handle enough to initiate the instability of the manipulator. The subject then let the oscillations drive him until the control loop ended (after 1.792 seconds). Figure 6.23 shows plots of the oscillation for subject A for one endpoint stiffness. Figure 6.24 shows plots of the oscillations for subject B with four different values of endpoint stiffness. In both cases there was a growing oscillation followed, in some cases, by what appears to be a period of steady state or decaying oscillations. This period of steady oscillations could have been due to either saturation of the actuators, or changes in the physiological system. The subject may be increasing his effective endpoint damping to either balance the unstable manipulator in steady state oscillations, or to reduce the oscillations.

To help in the interpretation of this data, a working hypothesis is postulated that assumes that the human subject does not change or adapt his parameters. If this was the case, then a linearized constant coefficient analysis would be valid with this data. For this analysis a semi-log plot of the peak amplitude of each oscillation versus time will determine the combined effective damping of the subject and the unstable

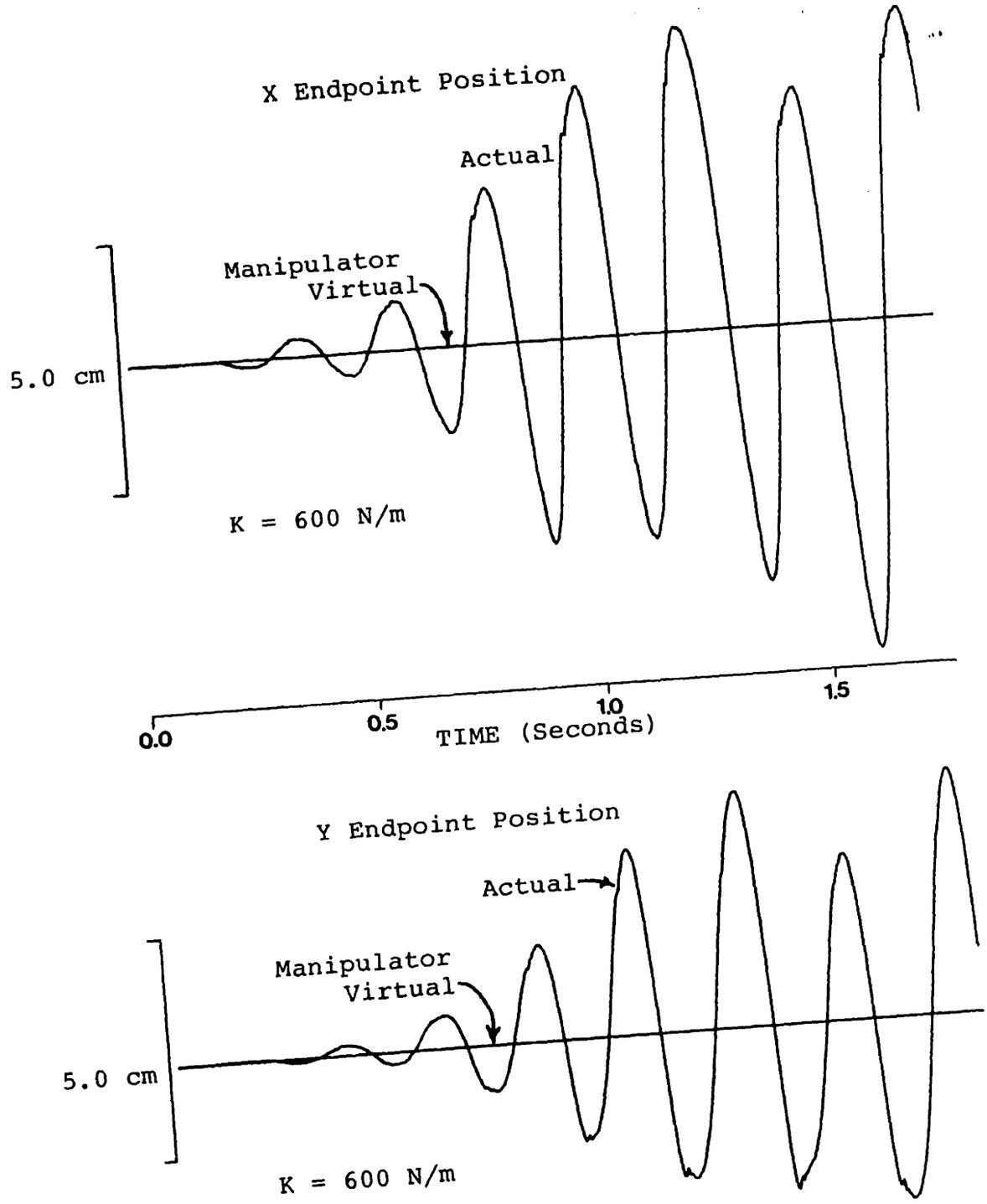


Figure 6.23 Oscillation Plots for Instability Compensation, Subject A

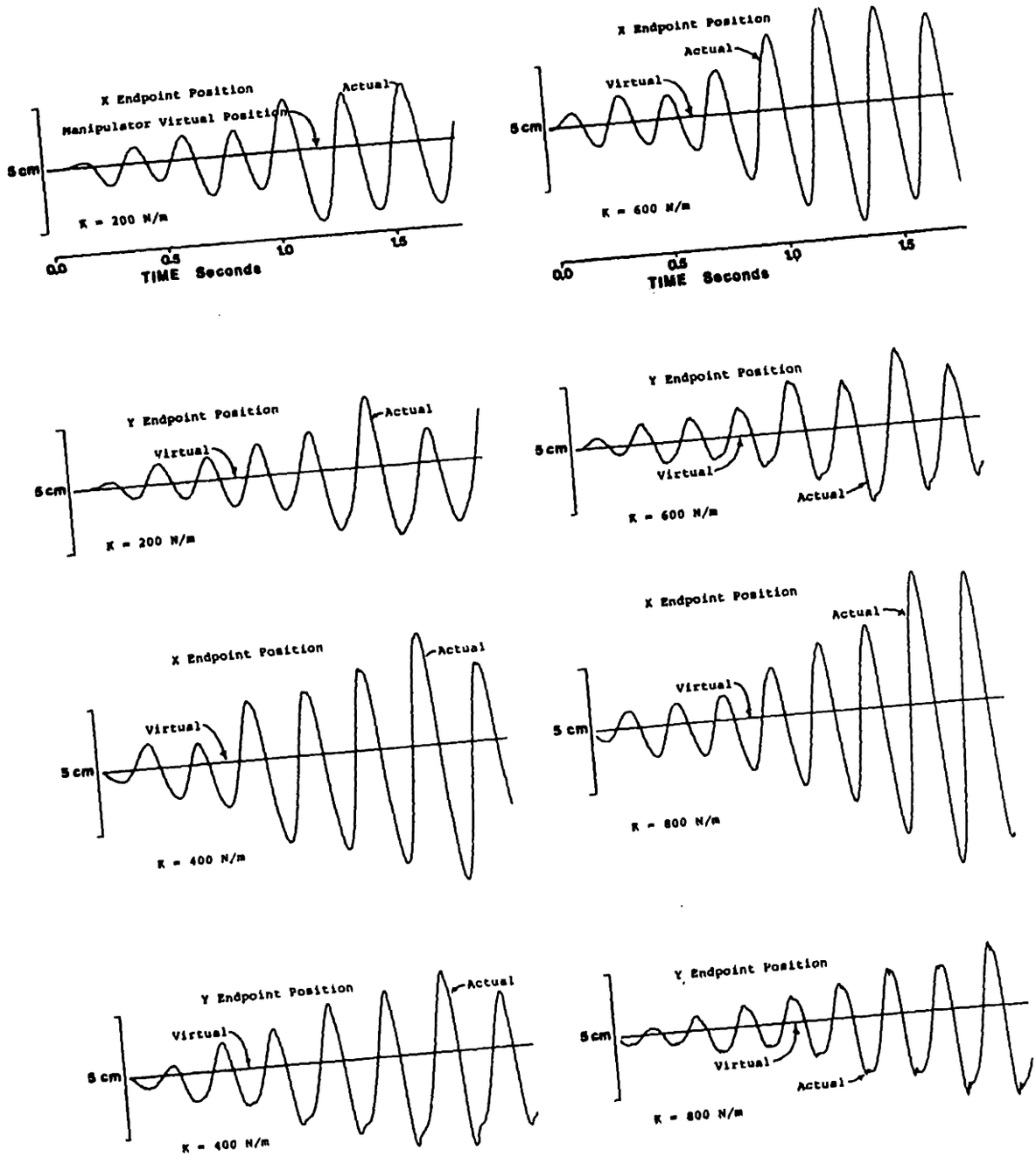


Figure 6.24 Oscillation Plots for Instability Compensation, Subject B

manipulator. A plot of the squared frequency of oscillations versus the commanded manipulator stiffness will determine the the combined effective mass of the subject and the manipulator, and the natural frequency and stiffness of the subject. Although the frequency of these oscillations is considered the damped natural frequency, the damping ratio is small enough so that the damped natural frequency and natural frequency will be considered the same.

For this linear analysis the effective endpoint damping is related to the damping ratio and the natural frequency by the relation:

$$B = 2\zeta\omega_n m \qquad [6\text{-}6]$$

And the natural frequency is related to the effective stiffness and mass by the relation:

$$\omega_n = \sqrt{\frac{K}{m}} \qquad [6\text{-}7]$$

Relation [6-7] can be rewritten to show the relationship between the natural frequencies, w_n, the combined stiffness of the subject and the manipulator ($K_{tot} = K_h + K_a$) and the combined effective mass of the subject and the manipulator ($M_{tot} = M_h + M_a$):

$$\omega_n^2 = \frac{K_{tot}}{M_{tot}} \qquad [6\text{-}8]$$

The difference between two natural frequencies is now computed:

$$\omega_{n2}^2 - \omega_{n1}^2 = \frac{(K_{h2} + K_{a2})}{M_{tot_2}} - \frac{(K_{h1} + K_{a1})}{M_{tot_1}} \qquad [6\text{-}9]$$

The assumption that the system parameters do not change implies that the subject's stiffness does not change ($K_{h1} = K_{h2}$), and that the combined mass of the human and the manipulator does not change ($M_{tot1} = M_{tot2} = M_{tot}$), and gives the following relationship:

$$\omega_{n2}^2 - \omega_{n1}^2 = \frac{K_{a2} - K_{a1}}{M_{tot}} \qquad [6\text{-}10]$$

Equation 6-10 shows that the inverse of the combined mass, M_{tot}, is the slope of the line plotted on this plot of the squared frequency of oscillations versus the commanded manipulator stiffness. The y intercept of the line plotted on this plot corresponds to a manipulator with zero stiffness and predicts the natural frequency of the subject. The x intercept of the line in this plot is the point where the natural frequency in relation 6-8 is zero and represents the stiffness of the human subject.

Using superposition it is possible to estimate the damping for the human subject. It is assumed that the total damping in the manipulator/human system is simply the sum of the damping due the human and the damping due to the manipulator:

$$B_{tot} = B_h + B_a \qquad [6\text{-}11]$$

By substituting relation 6-7 into 6-6 it is possible to estimate the total damping:

$$B_{tot} = 2\zeta\sqrt{K_{tot}M_{tot}} \qquad [6\text{-}12]$$

Since the commanded damping, Ba, in the impedance control algorithm is

known it is possible to compute the damping of the human subject, Bh. Solving relation 6-6 for the damping ratio, gives the following relation for the damping ratio of the subject:

$$\zeta_h = \frac{1}{2} B_h \sqrt{\frac{1}{K_h M_h}} \qquad [6\text{-}13]$$

To determine the total damping, B_{tot}, it is now necessary to determine the combined effective damping ratio of the human subject and the unstable manipulator. A stated previouly, this is done with a semi-log plot of the peak amplitude of each oscillation versus time. Although the stiffness of the manipulator defines an equilibrium point shown in the plots, the oscillations are not symmetric about this point. For this reason, the amplitude (for the semi-log plots) of each peak is measured from the previous peak. These semi-log plots are shown in Figure 6.25. From these plots the growing oscillations are clearly visible.

A linear regression was performed on two sets of data to see how well the data was correlated. This regression was performed for $\underset{\sim}{K} = 200\underset{\sim}{I}$ and $\underset{\sim}{K} = 800\underset{\sim}{I}$. For this regression some of the maverick points at the beginning and end of each data set were neglected. For $\underset{\sim}{K} = 200\underset{\sim}{I}$ the last two points were thrown out for the x direction, and the first and last two points were thrown out in the y direction. For $\underset{\sim}{K} = 800\underset{\sim}{I}$ only the last point was thrown out in the x direction, and the first and last points were thrown out in the y direction. Figure 6.26 shows plots of the data with the best fit lines in each set. For this data the corre-

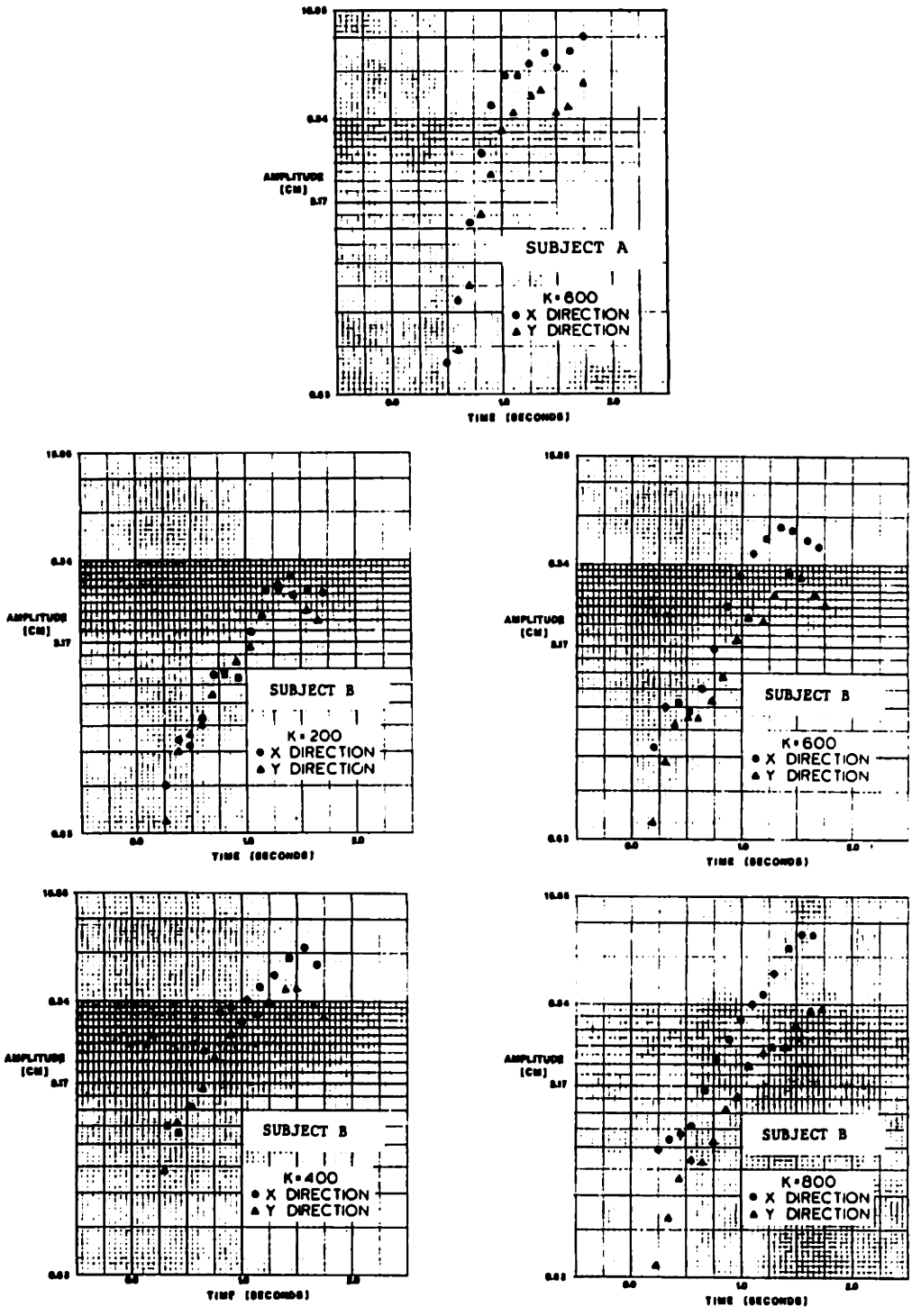
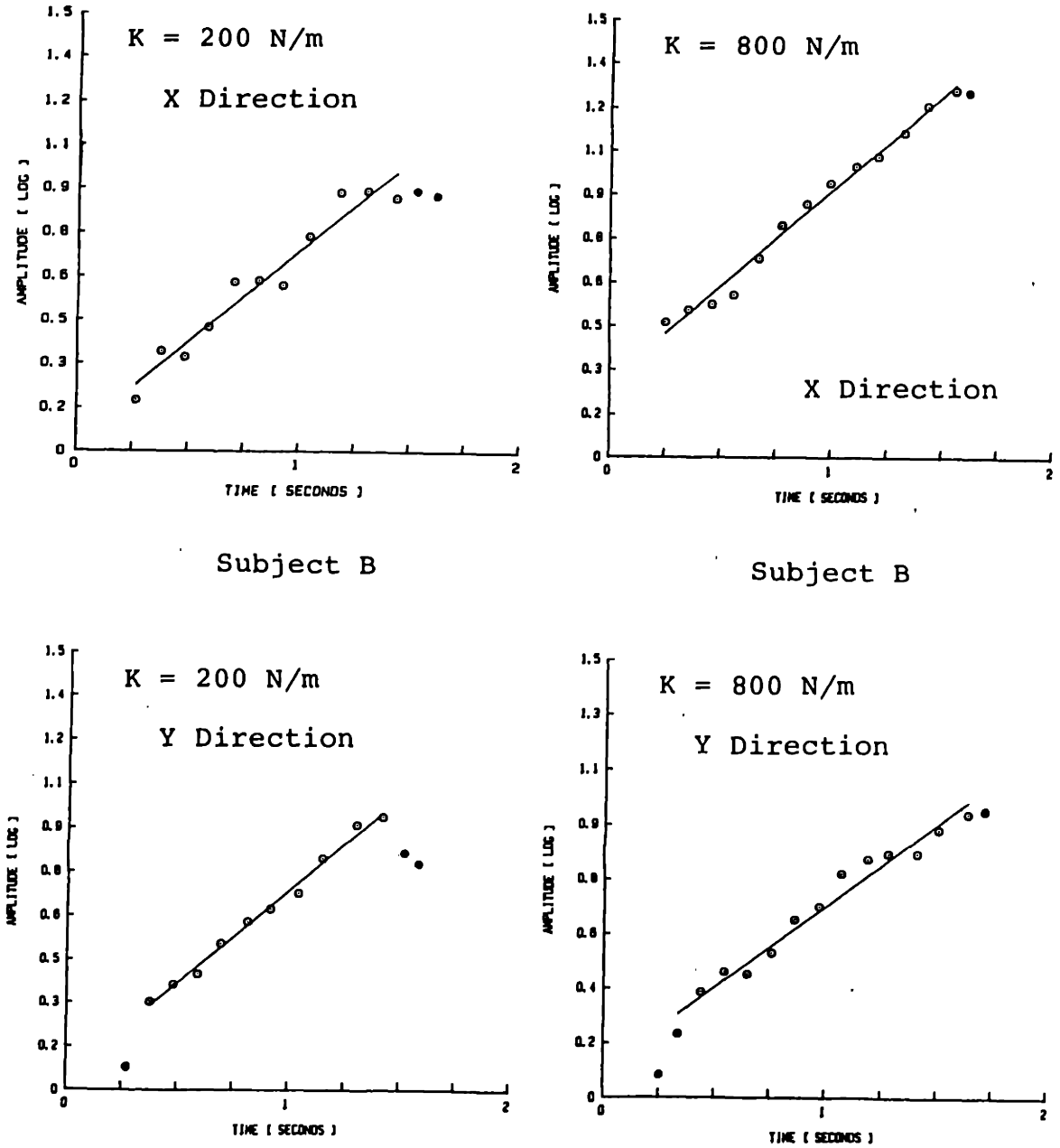


Figure 6.25 Logarithmic Plot of Peak Amplitude Versus Peak Time



● indicates omitted points

Figure 6.26 Linear Regression of Peak Amplitude Plots

lation coefficients range from .9719 to .9954.

Each best fit line has a slope associated with it. In the semi-log plot this slope is normally called the logarithmic decrement, but in this case the slope is more accurately referred to as the logarithmic increment. From this slope/logarithmic increment it is possible to compute the damping ratio with the relation:

$$\zeta^2 = \frac{\delta^2}{4\pi^2 + \delta^2} \qquad [6\text{-}14]$$

For the two sets of data that are best correlated (Figure 6.26c and Figure 6.26d), the negative damping ratios were found to be $\zeta = -.101$ and $\zeta = -.104$. This demonstrates that within the resolution of these particular experiments, the combined damping ratio of human arm and manipulandum is not changing even though the stiffness commands to the manipulator are different.

The next step in determining the total damping, B_{tot} (and ultimately the damping of the subject, Bh), is to determine the total effective mass of the system, and the stiffnes of the subject. Figure 6.27 shows the plot of these squared natural frequencies versus the manipulator stiffness used during the each experimental run. From Figure 6.23 the the natural frequencies were computed from consecutive periods and then averaged together. It is these averages with the computed standard deviation that are plotted in Figure 6.27. Points that deviated significantly from the trends in the data were left out of the average

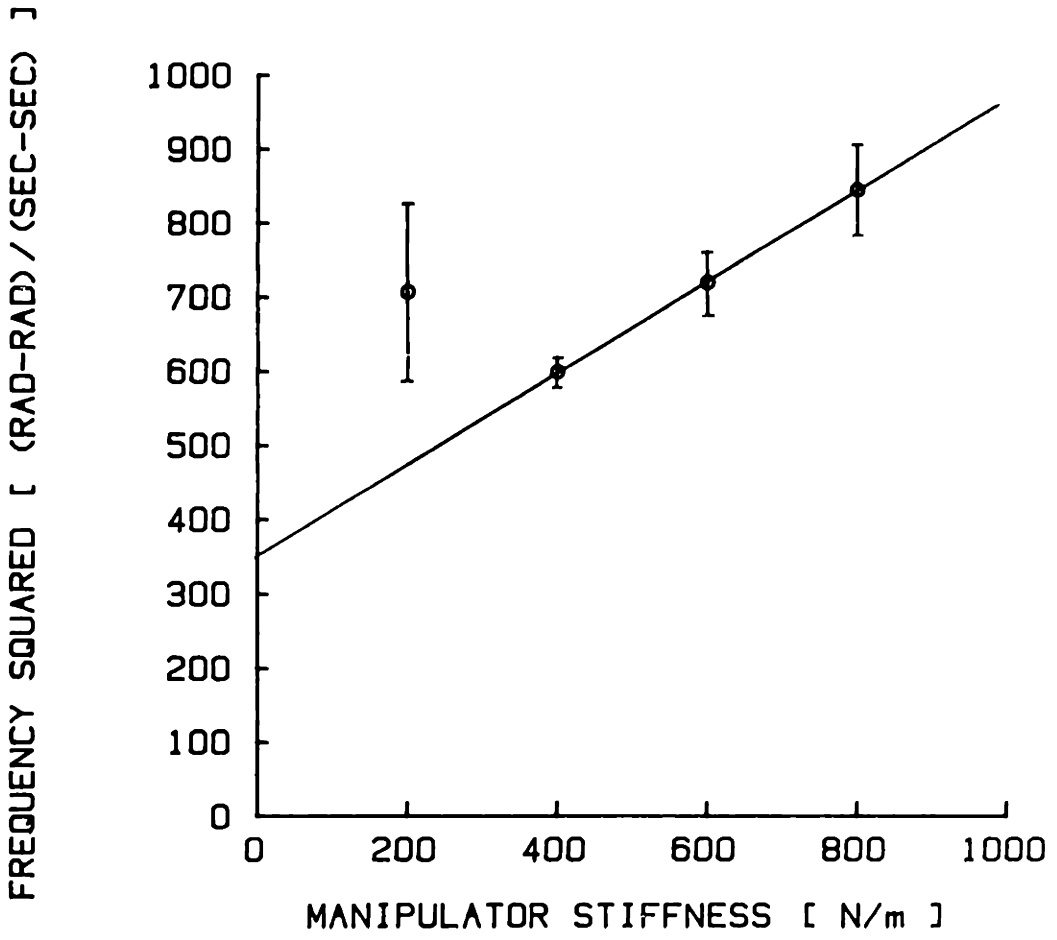


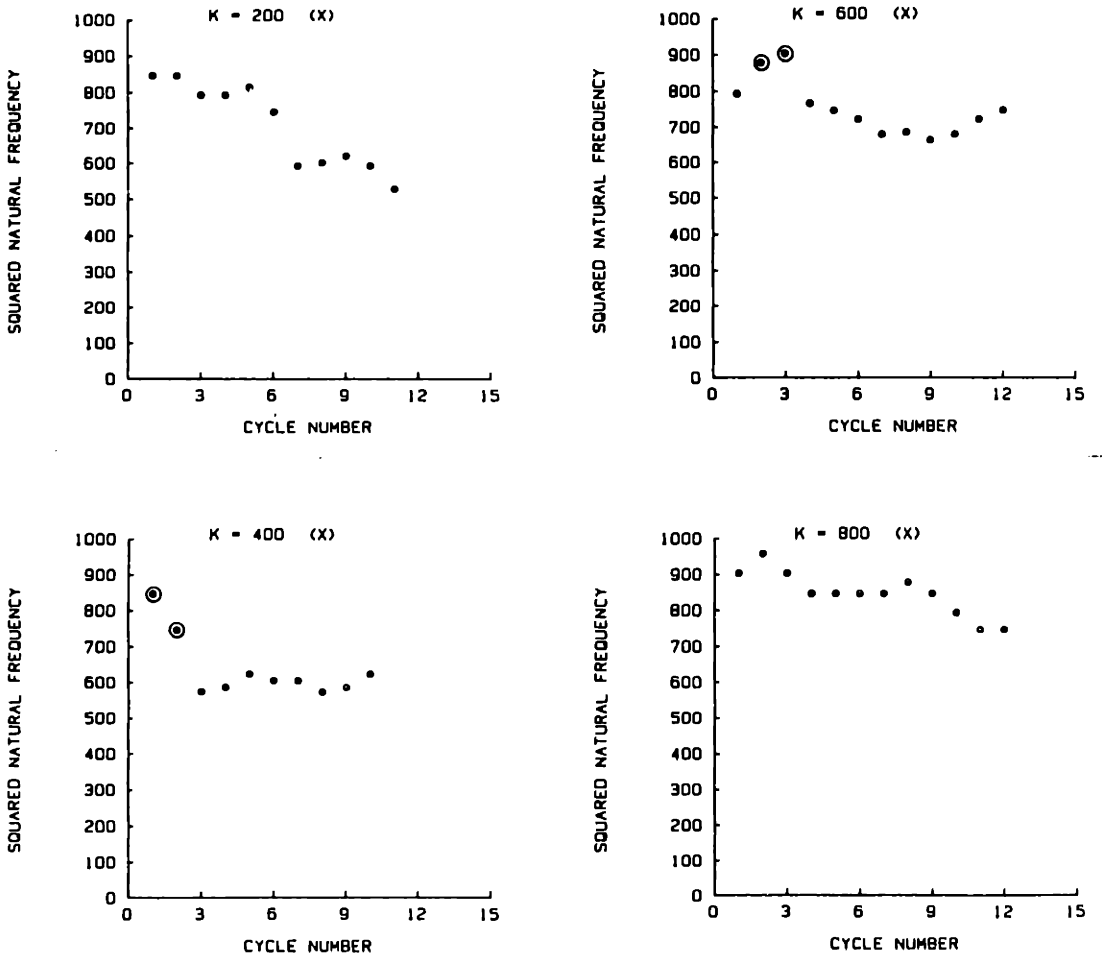
Figure 6.27 Averaged Natural Frequencies Versus Manipulator Stiffness

computation. Only two points for $\underset{\sim}{K} = 400\underset{\sim}{I}$ and two points for $\underset{\sim}{K} = 600\underset{\sim}{I}$ were left out. The justification for the omission of these points is shown in Figure 6.28 which contains plots of the squared frequencies versus cycle number.

For the three greatest manipulator stiffnesses the averaged values of squared natural frequencies fall on the straight line shown in Figure 6.27. The correlation coefficient of these points is 0.9999. Equation 6-10 showed that the inverse of the slope of this line is the combined mass, Mtot. From Figure 6.27 the slope is computed to be .6185, which makes the total mass 1.62 kg.. This is a reasonable value for the total mass, since the maximum apparent mass of the manipulator alone is about .8 kg. For the rest of this analysis it will be assumed that the effective mass of the subject is 1.0 Kg and the effective mass of the manipulator is .62 Kg (less than maximum apparent mass of the manipulator).

The y intercept of the line plotted in Figure 6.27 is the natural frequency of the subject. It was found to be 351 [(rad-rad)/(sec-sec)] or 2.98 hertz. While this number is high for human subjects, it is not unreasonable.

The x intercept of the line plotted in Figure 6.27 is the subject's stiffness and was found to be -568 N/meter. This number is at the high end of the range of stiffnesses previously measured for human subjects. It is possible that these quantities (human subject stiffness and natur-



◎ indicates omitted points

Figure 6.28 Squared Natural Frequency Versus Cycle Number

al frequency) are high because the subject was required to stabilize the unstable manipulator.

Now a value of the damping ratio for the human subject can be estimated. For the following values:

$$\zeta = -0.101$$

$$K_{tot} = 800 + 568 = 1368$$

$$M_{tot} = 1.62$$

The total damping, B_{tot}, computed from relation 6-12 is -9.5 N-sec/meter. The commanded damping in the computer was -15 N-sec/meter, thus the damping due to the subject is computed to be 5.5 N-sec/meter. Substituting this damping of the subject along with the values of the humans effective mass and stiffness into relationship 6-13 gives a damping ratio of $\zeta_h = .120$. While this value seems low, it is a reasonable estimation for this preliminary investigation.

Through this analysis it has been possible to quantify a set of parameters that could not previously be identified. Clearly this apparatus opens up new avenues of experimentation.

CHAPTER 7: DISCUSSION, CONCLUSIONS AND RECOMENDATIONS FOR FURTHER WORK

This chapter discusses some of the experimental results from both the endpoint force generation and the control of endpoint dynamics, stiffness and viscosity. Suggestions for improvements and futher work on the apparatus are made as well as recomendations for possible uses of the apparatus.

7.1 Discussion of Results

7.1.1 Endpoint Force Production

The endpoint force production was evaluated at several different locations in the workspace. The calibration orientation was chosen because that orientation would provide the least errors in actual endpoint position. The other two orientations were chosen to provide some variation in the linkage configuration (namely in the relative angle between the joints).

While the force production appears to be the best in the calibration orientation, deviations in the desired and measured forces are clearly seen in the other orientations. These errors are present in both the magnitude and direction of the measured force. The greatest error is in the 10 newton case and is 8 percent of the maximum force commanded. While this error is acceptable, improved endpoint force production is desired and is left for further work. The variations in Figures 6.4 and 6.5 and Figure 6.3 might be attributed to the fact that

the data in Figure 6.3 was taken 6 months prior to the data taken in Figures 6.4 and 6.5. This assumes that in the course of 6 months, the transducer calibration has been degraded. This could be checked by re-testing the calibration orientation, but is also left here for further work.

It was shown that within the limits of the actuators, it is possible to generate a desired force at the manipulator endpoint in any direction. This was only shown, however, in the static case and further work will be required to show that the same force production is possible when the manipulator endpoint is free to move around in the workspace.

The servo amplifier current monitor was evaluated as means of estimating the endpoint force when the manipulator was free to move around in the workspace. In some cases this estimation was consistent with the commanded force, but it was shown that in most directions the estimation was a less accurate measure of endpoint force. Another drawback to using the servo amplifier current monitor to estimate the endpoint force is that torques that are commanded to be output from the motors (the quantity that the current monitor is measuring) will not represent the actual endpoint force in the dynamic motion of the manipulator. It may be possible to estimate the endpoint force with a model based observer, but the best solution is to measure force directly at the manipulator endpoint.

7.1.2 Impedance Control

The performance of the impedance controller was found to be good for both the control of the endpoint stiffness and the endpoint viscosity. It was possible to evaluate the control of the endpoint stiffness both statically and dynamically. The endpoint viscosity, however, is a function of velocity and could only be evaluated dynamically.

7.1.2.1 Stiffness Control

Allowing for the non-ideal behavior of the transducer, the stiffness control of the static endpoint is very good. In both orientations of the linkage, the outputs from the force transducer are nearly the same. Given the plots for the two different orientations (Figure 6.11) it is not possible to determine which orientation corresponds with which plot.

The ability to orient and shape the endpoint stiffness was clearly demonstrated in Figure 6.12. From this figure it is also possible to see the effects of the non-ideal force transducer used to measure the endpoint forces. In all of these plots it appears as though the offsets in the transducer output are always in the direction of the second and fourth quadrants. This is especially true in Figure 6.12b where the deviation of the force transducer output is towards the major axis of the ellipse formed by the command forces. This supports the claim in the previous chapter, that the force transducer may have preferred directions of displacement.

The undamped oscillation tests were performed to show that it was possible to determine an endpoint stiffness (based on the modelled endpoint inertance) that will produce oscillations with predictable frequencies. In this way, the undamped oscillation tests not only demonstrated the ability to control endpoint stiffness, but they also verify the manipulator model developed in chapter 5. Errors in the measured and predicted frequency are small (.2% to .79%).

It might be of interest in future work to include an algorithm in the control program that modifies the endpoint stiffness matrix based on the mathematical model of the manipulator endpoint inertance. In this program an experimenter would enter an endpoint natural frequency rather than the endpoint stiffness. The algorithm would then maintain this natural frequency regardless of the linkage orientation.

7.1.2.2 Viscosity Control

The control of endpoint viscosity was verified in two different ways. The reponse of the system to a step command in displacement was monitored for various values of the endpoint damping and the manipulator endpoint was moved by a subject with only a viscous load commanded at the endpoint.

For the step response it was shown that increasing the effective endpoint damping decreased the size and number of oscillations. There were problems, however, with steady state offsets in position for large values of damping. These positioning errors were shown to be dependent

on both the static friction in the system, and the amount of torque required to drive the outer link to its final position.

With only a pure viscous load commanded, the commanded endpoint forces should have been a mirror image of the measured velocity (i.e., 1 meter/second equals -1 Newton). This is exactly the case for Figure 6.17, when $\underset{\sim}{B} = 15\underset{\sim}{I}$, and demonstrates that the algorithm is working properly for the commanded viscous load.

7.1.3 Impedance Control Parameter Limits

Although the physical limits of $\underset{\sim}{K}$ and $\underset{\sim}{B}$ were not quantitatively evaluated, several qualitative experiments were performed to determine maximum limits on values for these desired endpoint parameters. These experiments were performed to determine how large these parameters could be before an instability ocurred. The instability in this case was a high frequency oscillation at approximately 400 hertz, that caused undesirable vibration in the manipulator links.

7.2 Suggestions For System Improvements

7.2.1 Modelling of Support Frame

In the tachometer filter section of chapter 4 it was demonstrated that there is an approximately 400 hertz vibration in the support frame. This vibration appears to generate 400 hertz noise on the tachometer

output signal that is detrimental to stable velocity control. A 4^{th} order low pass filter was found to be very effective in eliminating this noise, but is by no means the optimal solution to the problem. This filter has considerable phase lag at low frequencies and may eventually compromise the implementation of the impedance controller [11] proposed for further work.

The solution to this problem is to develop a model of the system that will predict why this 400 hertz vibration causes instability in velocity control of the manipulator. An accurate model of the system will facilitate the design of an improved compensator for this application. This improved compensator would pass a velocity signal to the control algorithm that has less than 5 degrees of phase lag at 40 hertz (i.e., one decade down from 400 hertz).

7.2.2 Servo Amplifier Replacements

There are several problems with the servo amplifiers. The most obvious problem is with the 5000 hertz switching frequency. This 5 Khz switching frequency produces audible sound at this frequency. This is of concern because the intensity of this sound is dependent on the amount of torque being applied by the motors. Although it has not yet been shown that this is a problem, it is possible that subjects may receive audible cues from this sound and make changes in movement strategy based on these cues.

There is also a problem with the servo amplifier feature that allows peak current output for half a second. If a constant force is desired at the endpoint for more than half a second, then this desired force will be limited by the continuous current output capabilities of the servo amplifier. This means that forces that require the peak output current of the amplifiers can only be sustained for half a second.

It was shown earlier that it is not possible to use the peak output current of the amplifiers due to 120 hertz noise in the amplifier. This 120 hertz noise produced undesirable vibrations in output torque and hence the command voltages to the amplifiers had to be limited to a range between ± 8 volts. For this reason, the amplifiers are not being utilized to their maximum capabilities.

It is suggested that these amplifiers be replaced with linear amplifiers. Properly designed linear amplifiers would not exhibit the 120 hertz noise seen on the PMI servo amplifiers and would not produce the high frequency sound due the amplifier switching. In terms of increasing the amount of achievable output force, it would be desirable to purchase amplifiers that do not have a decay from the peak to the continuous current limit. If this is the case, then it will be the responsibility of the control algorithm to ensure that neither the motors nor the amplifiers are overloaded. American Time Products builds linear amplifiers that would be ideally suited for this application, unfortunately, these amplifiers are currently being redesigned and will not be availible for several more months.

7.3 Further Work

7.3.1 Force Feedback

It has become clear that the ability to measure force directly at the manipulator endpoint is extremely useful. Not only would this capability allow the experimenter to monitor subject's interaction force with the manipulator, but the signal can also be fed back so that the implementation of the complete impedance controller [11] is possible. With the implementation of this controller, controlled mechanical interaction with the environment is possible. In this way it is not only possible to control the apparent endpoint stiffness and viscosity, but the apparent inertance as well.

To implement force feedback in this system, a LORD Corporation six axis force-torque wrist sensor, the F/T 15/50, is currently being interfaced. This system is designed to be used in robotic applications as the interface between robot and end-effector. Its application to this project is ideal. The main proceesing unit can be programmed to compute force and torque transformations to any point in the workspace. The output from the main processing unit can be either fed into the computer in serial form, or in parallel form. Application of the serial interface uses a currently existing serial port interface in the computer and does not require the addition or modification of any hardware. The drawback to the serial interface is that the transmission rate is slow and limited to 74 hertz (serial binary data). The parallel interface is faster (a transmission rate that is 104 hertz), but does require some

modification of the interface connections. These modifications will be made so that the force information can be fed into the computer as rapidly as possible.

After this new force transducer is implemented, it is suggested that force production experiments be repeated to confirm the claims that the errors in force generation were actually due to the measurement systems force joystick. It is also suggested that step responses be performed at the endpoint to find the maximum rate of application of forces. In performing this test, however, care must be taken to insure that the manipulandum is being characterized and not the force transducer.

7.3.2 Parallel Processing

One option for improved system performance is the use of two computers. One computer would act as the host computer and run all the primary experimental protocol. The other computer would act as a slave to the host computer and would be entirely dedicated to real time control. The host computer would make decisions about details such as the type of perturbation to the subject and the desired set of parameters for that perturbation. The slave computer would be dedicated to real time control of the manipulandum. Its only function would be to interface with the experimental hardware and implement the desired control algorithm. In some cases it may be possible to divide the workload of computationally compllex algorithms, thus the complexity of a control algorithm would not be a limiting factor in real time control. The

slave computer would operate on time critical computations and the host computer would operate on computations that do require a high computation rate.

7.4 Potential Uses Of The Apparatus

A preliminary experiment was presented at the end of chapter 6. This experiment demonstrates that it may be possible to measure parameters in human subjects that have not been measured before. These quantities are the effective damping and inertance that the subject exhibits at his hand. This was done by destablizing the manipulator with positive velocity feedback and then observing the motion that results when the stable human is coupled to the unstable manipulator. While these experiments were performed without any careful controls or strict experimental protocol, they do show that there are trends in the data from two different subjects that are consistent.

Another experiment that would be interesting, would be to see if it is possible to de-couple mass and weight. This would require some hardware modification that would re-orient plane of motion in the vertical plane In this new plane of motion gravity will have an effect on the manipulator. With impedance control it should be possible to make the endpoint have an apparently low inertance, but then add in a bias force that will result in a higher effective gravitational constant. And likewise, it should be possible to make the endpoint have an apparently high inertance, and then command the interface force to be low. This would be effectively reducing the effective gravitational constant and

thereby simulates a space environment.These experiments could then be potentially verified by similar experiments done in space where gravity is minimal.

7.5 Conclusion

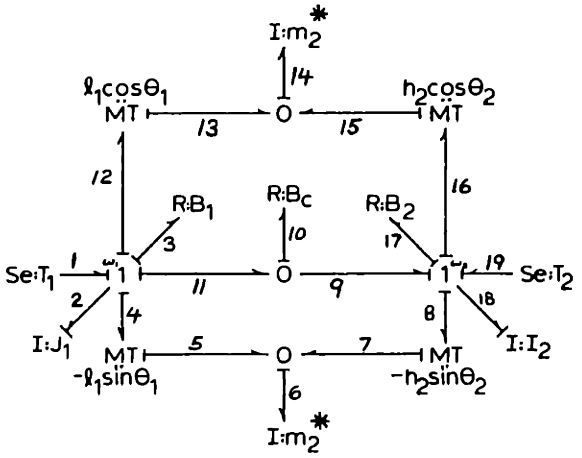
The work has demonstrated the application of an impedance controller to a two serial link manipulator. It has been shown that it is possible achieve target endpoint impedances within the limitations of the actuators. It was then shown that this form of the impedance controller is suitable for continuing and developing the study of human arm movements. The impedance controller not only allows the experimenter to command endpoint position, but by definition it allows the modification of endpoint properties such as stiffness and viscosity. It was the modification of endpoint viscosity that was used in a new experiment to determine physical parameters in the human subject.

This new experiment showed that with a set of simplifying assumptions (a linear system with constant stiffness and constant apparent mass) it is possible to determine the subject's dynamic damping and stiffness. Although the analysis of the results from this experiment produced reasonable results for stiffness and damping, this is by no means the definitive experiment. In order to strengthen the validity of these experiments, a larger field of experimental subjects is needed, as well as careful planning of the experimental protocol. Careful consideration must be given to the commanded endpoint instabilities and the

interpertation of the data collected.

While this instability compensation experiment may eventually provide some insight into the neurological details of movement control, it is only used in this study to determine effective endpoint parameters. It is reasonable to assume that the combined neurological, muscular and skeletal systems provide a set of effective endpoint parameters. Clearly it has been shown that with the impedance controlled manipulandum, it is possible to determine these endpoint parameters. Accurate determination of these parameters and inferences about the neurological system are left for further work.

APPENDIX A: Derivation of Two-link Manipulator Dynamics Using Bond Graph Techniques.



Bond Graph Model for a Two-link Manipulator

Lagrangian State Variables: $f_2 = \omega_1$ (flow) $= \dot{\theta}_1$

$f_{18} = \omega_2$ (flow) $= \dot{\theta}_2$

* Differential Causality:

$$e_6 = m_2\frac{df_6}{dt} \quad \text{(effort on bond 6)} \qquad [A\text{-}1]$$

$$e_{14} = m_2\frac{df_{14}}{dt} \quad \text{(effort on bond 14)} \qquad [A\text{-}2]$$

In terms of Lagrangian State Variables:

$$f_6 = -\ell_1\sin(\theta_1)f_2 - h_2\sin(\theta_2)f_{18} \qquad [A\text{-}3]$$

$$f_{14} = -\ell_1\cos(\theta_1)f_2 - h_2\cos(\theta_2)f_{18} \qquad [A\text{-}4]$$

Differentiate relations [A-2] and [A-3] with respect to time:

$$\frac{df_6}{dt} = -\ell_1\cos(\theta_1)\omega_1 - \ell_1\sin(\theta_1)\frac{d\omega_1}{dt} - h_2\cos(\theta_2)\omega_2 - h_2\sin(\theta_2)\frac{d\omega_2}{dt} \qquad [A\text{-}5]$$

$$\frac{df_{14}}{dt} = -\ell_1\sin(\theta_1)\omega_1 + \ell_1\cos(\theta_1)\frac{d\omega_1}{dt} - h_2\sin(\theta_2)\omega_2 + h_2\cos(\theta_2)\frac{d\omega_2}{dt} \qquad [A\text{-}6]$$

Elemental Equations:

$$\frac{d\omega_1}{dt} = \frac{1}{J_1}e_2 \qquad [A\text{-}7] \qquad\qquad \frac{d\omega_2}{dt} = \frac{1}{I_2}e_{18} \qquad [A\text{-}8]$$

Where junction equations are:

$$e_2 = e_1 - e_3 - e_4 - e_{11} - e_{12}$$

$$e_{18} = e_{19} - e_8 - e_9 - e_{16} - e_{17}$$

In terms of the Lagrangian state variables

$$e_1 = T_1$$

$$e_3 = \omega_1 B_1$$

$$e_4 = -\ell_1 \sin(\theta_1) m_2 \frac{df_6}{dt}$$

$$e_8 = -h_2 \sin(\theta_2) m_2 \frac{df_6}{dt}$$

$$e_9 = B_c(\omega_1 - \omega_2)$$

$$e_{11} = B_c(\omega_1 - \omega_2)$$

$$e_{12} = \ell_1 \cos(\theta_1) m_2 \frac{df_{14}}{dt}$$

$$e_{16} = h_2 \cos(\theta_2) m_2 \frac{df_{14}}{dt}$$

$$e_{17} = \omega_2 B_2$$

$$e_{19} = T_2$$

Substitution into equations [A-7] and [A-8] gives:

$$\dot{\omega}_1 = \frac{1}{J_1}[\ T_1 - \omega_1 B_1 - \ell_1 m_2 \sin(\theta_1)\cos(\theta_1)\omega_1^2 - \ell_1^2 m_2 \sin^2(\theta_1)\dot{\omega}_1$$
$$- \ell_1 h_2 m_2 \sin(\theta_1)\cos(\theta_2)\omega_2^2 - \ell_1 h_2 m_2 \sin(\theta_1)\sin(\theta_2)\dot{\omega}_2$$
$$- B_c(\omega_1 - \omega_2) + \ell_1^2 m_2 \cos(\theta_1)\sin(\theta_1)\omega_1^2 - \ell_1^2 m_2 \cos^2(\theta_1)\dot{\omega}_1$$
$$+ \ell_1 h_2 m_2 \sin(\theta_2)\cos(\theta_1)\omega_2^2 - \ell_1 h_2 m_2 \cos(\theta_1)\cos(\theta_2)\ddot{\theta}_2\ ] \qquad [A\text{-}9]$$

And

$$\dot{\omega}_2 = \frac{1}{I_2}[\ T_2 - \omega_2 B_2 - \ell_1 h_2 m_2 \sin(\theta_2)\cos(\theta_1)\omega_1^2 - h_2^2 m_2 \sin^2(\theta_2)\dot{\omega}_2$$
$$+ h_2^2 m_2 \sin(\theta_2)\cos(\theta_2)\omega_2^2 - \ell_1 h_2 m_2 \sin(\theta_1)\sin(\theta_2)\dot{\omega}_1$$
$$+ B_c(\omega_1 - \omega_2) + \ell_1 h_2 m_2 \cos(\theta_2)\sin(\theta_1)\omega_1^2 + h_2^2 m_2 \cos^2(\theta_2)\dot{\omega}_2$$
$$- h_2^2 m_2 \sin(\theta_2)\cos(\theta_2)\omega_2 - \ell_1 h_2 m_2 \cos(\theta_1)\cos(\theta_2)\ddot{\theta}_1\ ] \qquad [A\text{-}10]$$

Collect the terms:

$$\dot{\omega}_1 = \frac{1}{J_1}[\ T_1 - \omega_1 B_1 - \ell_1^2 m_2(\sin^2(\theta_1) + \cos^2(\theta_1))\dot{\omega}_1 \qquad \text{[A-11]}$$
$$- \ell_1 h_2 m_2(\sin(\theta_1)\cos(\theta_2) - \sin(\theta_2)\cos(\theta_1))\omega_2^2$$
$$- \ell_1 h_2 m_2(\sin(\theta_1)\sin(\theta_2) + \cos(\theta_1)\cos(\theta_2))\dot{\omega}_2 - B_c(\omega_1 - \omega_2)\ ]$$

$$\dot{\omega}_2 = \frac{1}{I_2}[\ T_2 - \omega_2 B_2 - \ell_1 h_2 m_2(\sin(\theta_2)\cos(\theta_1) - \sin(\theta_1)\cos(\theta_2))\omega_1^2$$
$$- h_2^2 m_2(\sin^2(\theta_2) + \cos^2(\theta_2))\dot{\omega}_2 \qquad \text{[A-12]}$$
$$- \ell_1 h_2 m_2(\sin(\theta_1)\sin(\theta_2) + \cos(\theta_1)\cos(\theta_2))\dot{\omega}_1 + B_c(\omega_1 - \omega_2)\ ]$$

Applying Trigonometric Identities

Equation [A-11] becomes:

$$J_1\ \dot{\omega}_1 = T_1 - \omega_1(B_1 + B_c) + B_c\omega_2 - \ell_1 h_2 m_2 \sin(\theta_1 - \theta_2)\omega_2^2$$
$$- \ell_1^2 m_2 \dot{\omega}_1 - \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1)\dot{\omega}_2$$

$$J_1\ \dot{\omega}_1 = T_2 - \omega_2(B_2 + B_c) + B_c\omega_1 - \ell_1 h_2 m_2 \sin(\theta_1 - \theta_2)\omega_1^2$$
$$- h_2^2 m_2 \dot{\omega}_2 - \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1)\dot{\omega}_1$$

Again Collecting Terms:

$$(J_1 + \ell_1^2 m_2)\ \dot{\omega}_1 + \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1)\ \dot{\omega}_2$$
$$= T_1 - \omega_1(B_1 + B_c) + B_c\omega_2 + \ell_1 h_2 m_2 \sin(\theta_2 - \theta_1)\omega_2^2$$

$$(I_2 + h_2^2 m_2)\ \dot{\omega}_2 + \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1)\ \dot{\omega}_1$$
$$= T_2 - \omega_2(B_2 + B_c) + B_c\omega_1 - \ell_1 h_2 m_2 \sin(\theta_2 - \theta_1)\omega_1^2$$

Written in Matrix Form:

$$\begin{vmatrix} J_1 + \ell_1^2 m_2 & \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) \\ \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) & J_2 \end{vmatrix} \begin{vmatrix} \frac{d\omega_1}{dt} \\ \frac{d\omega_2}{dt} \end{vmatrix} =$$

$$\begin{vmatrix} -(B_1 + B_c) & B_c \\ B_c & -(B_2 + B_c) \end{vmatrix} \begin{vmatrix} \omega_1 \\ \omega_2 \end{vmatrix} + \begin{vmatrix} T_1 + \ell_1 h_2 m_2 \omega_2^2 \sin(\theta_2 - \theta_1) \\ T_2 - \ell_1 h_2 m_2 \omega_1^2 \sin(\theta_2 - \theta_1) \end{vmatrix}$$

where: $J_2 = I_2 + h_2^2 m_2$

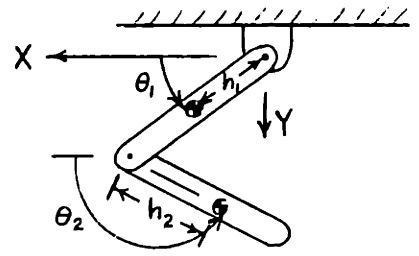
APPENDIX B: Derivation of Two-link Manipulator Dynamics Using Lagrange's Techniques.

Lagrange's Equation:

$$\frac{d}{dt}\left|\frac{\partial \mathcal{L}}{\partial \xi_i}\right| - \left|\frac{\partial \mathcal{L}}{\partial \xi_i}\right| = \chi_i$$

where: ξ_i are the generalized coordinates
χ_i are the generalized forces

For the manipulator:



$\xi_1 = \theta_1$

$\xi_2 = \theta_2$

$\chi_1 = T_1$

$\chi_2 = T_2$

Moment of inertia for each link:

a) About the center of gravity I_1 and I_2

b) About the axis of rotation J_1 and J_2

The parallel axis theorem gives:

$$J_1 = I_1 + m_1 h_1^2$$

$$J_2 = I_2 + m_2 h_2^2$$

The Lagrangian is the combination of kinetic co-energy and potential energy

$$\mathcal{L} = T^* - V$$

The manipulator acts purely in the horizontal plane, hence the contribution from the potential energy term is zero:

$$V = 0$$

Kinetic co-energy:

$$T^* = \tfrac{1}{2} I_1 \dot{\theta}_1^2 + \tfrac{1}{2} I_2 \dot{\theta}_2^2 + \tfrac{1}{2} m_1 (h_1 \dot{\theta}_1)^2 + \tfrac{1}{2} m_2 (u_2^2 + v_2^2)$$

where:

$$u_2 = \ell_1 \dot{\theta}_1 \sin(\theta_1) + h_2 \dot{\theta}_2 \sin(\theta_2)$$

$$v_2 = \ell_1 \dot{\theta}_1 \cos(\theta_1) + h_2 \dot{\theta}_2 \cos(\theta_2)$$

and

$$(u_2^2 + v_2^2) = \ell_1^2\dot{\theta}_1^2 + 2\ell_1 h_2\theta_1\theta_2(\sin(\theta_1)\sin(\theta_2) + \cos(\theta_1)\cos(\theta_2)) + h_2^2\dot{\theta}_2^2$$

$$= \ell_1^2\dot{\theta}_1^2 + 2\ell_1 h_2\theta_1\theta_2\cos(\theta_2 - \theta_1) + h_2^2\dot{\theta}_2^2$$

With substitution and the parallel axis theorem, the kinetic co-energy is:

$$T^* = \tfrac{1}{2} J_1\dot{\theta}_1^2 + \tfrac{1}{2} J_2\dot{\theta}_2^2$$

$$+ \tfrac{1}{2} m_1(\ell_1\dot{\theta}_1)^2 + \ell_1 h_2 m_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_2 - \theta_1)$$

frictional losses are added into the equation as external forces:

for θ_1: $B_c\theta_2 - (B_1 + B_c)\theta_1$

for θ_2: $B_c\theta_1 - (B_2 + B_c)\theta_2$

where:

B_1 and B_2 are the frictional losses in the motors

B_c is the friction between the two links

Apply Lagrange's Equation to each generalized coordinate:

For θ_1: $\frac{d}{dt}\left| J_1\dot{\theta}_1 + m_2\ell_1^2\dot{\theta}_1 + \ell_1 h_2 m_2\dot{\theta}_2\cos(\theta_2 - \theta_1) \right|$

$$- \left| \ell_1 h_2 m_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_2 - \theta_1) \right| = T_1 + B_c\dot{\theta}_2 - (B_1 + B_c)\dot{\theta}_1$$

For θ_2: $\frac{d}{dt}\left| J_2\dot{\theta}_2 + \ell_1 h_2 m_2\dot{\theta}_1\cos(\theta_2 - \theta_1) \right|$

$$- \left| -\ell_1 h_2 m_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_2 - \theta_1) \right| = T_2 + B_c\dot{\theta}_1 - (B_2 + B_c)\dot{\theta}_2$$

Perform the differentiation:

For θ_1: $J_1\ddot{\theta}_1 + m_2\ell_1^2\dot{\theta}_1 + \ell_1 h_2 m_2\ddot{\theta}_2\cos(\theta_2 - \theta_1) - \ell_1 h_2 m_2\dot{\theta}_2(\dot{\theta}_2 - \dot{\theta}_1)\sin(\theta_2 - \theta_1)$

$$- \ell_1 h_2 m_2\dot{\theta}_2\dot{\theta}_1\sin(\theta_2 - \theta_1) = T_1 + B_c\dot{\theta}_2 - (B_1 + B_c)\dot{\theta}_1$$

For θ_2: $J_2\ddot{\theta}_2 + \ell_1 h_2 m_2\ddot{\theta}_1\cos(\theta_2 - \theta_1) - \ell_1 h_2 m_2\dot{\theta}_1(\dot{\theta}_2 - \dot{\theta}_1)\sin(\theta_2 - \theta_1)$

$$+ \ell_1 h_2 m_2\dot{\theta}_1\dot{\theta}_2\sin(\theta_2 - \theta_1) = T_2 + B_c\dot{\theta}_1 - (B_2 + B_c)\dot{\theta}_2$$

Algebraic reduction gives:

$$\text{For } \theta_1: \quad (J_1 + m_2\ell_1^2)\ddot{\theta}_1 + \ell_1 h_2 m_2 \ddot{\theta}_2 \cos(\theta_2 - \theta_1) - \ell_1 h_2 m_2 \dot{\theta}_2^2 \sin(\theta_2 - \theta_1) = T_1 + B_c\dot{\theta}_2 - (B_1 + B_c)\dot{\theta}_1$$

$$\text{For } \theta_2: \quad J_2\ddot{\theta}_2 + \ell_1 h_2 m_2 \ddot{\theta}_1 \cos(\theta_2 - \theta_1) + \ell_1 h_2 m_2 \dot{\theta}_1^2 \sin(\theta_2 - \theta_1) = T_2 + B_c\dot{\theta}_1 - (B_2 + B_c)\dot{\theta}_2$$

This can be written:

$$\text{For } \theta_1: \quad (J_1 + m_2\ell_1^2)\ddot{\theta}_1 + \ell_1 h_2 m_2 \ddot{\theta}_2 \cos(\theta_2 - \theta_1) = T_1 + \ell_1 h_2 m_2 \dot{\theta}_2^2 \sin(\theta_2 - \theta_1) + B_c\dot{\theta}_2 - (B_1 + B_c)\dot{\theta}_1$$

$$\text{For } \theta_2: \quad J_2\ddot{\theta}_2 + \ell_1 h_2 m_2 \ddot{\theta}_1 \cos(\theta_2 - \theta_1) = T_2 - \ell_1 h_2 m_2 \dot{\theta}_1^2 \sin(\theta_2 - \theta_1) + B_c\dot{\theta}_1 - (B_2 + B_c)\dot{\theta}_2$$

Written in Matrix Form:

$$\begin{vmatrix} J_1 + \ell_1 m_2 & \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) \\ \ell_1 h_2 m_2 \cos(\theta_2 - \theta_1) & J_2 \end{vmatrix} \begin{vmatrix} \frac{d\omega_1}{dt} \\ \frac{d\omega_2}{dt} \end{vmatrix} =$$

$$\begin{vmatrix} -(B_1 + B_c) & B_c \\ B_c & -(B_2 + B_c) \end{vmatrix} \begin{vmatrix} \omega_1 \\ \omega_2 \end{vmatrix} + \begin{vmatrix} T_1 + \ell_1 h_2 m_2 \omega_2^2 \sin(\theta_2 - \theta_1) \\ T_2 - \ell_1 h_2 m_2 \omega_1^2 \sin(\theta_2 - \theta_1) \end{vmatrix}$$

[Equation 5-2 Non-linear model of the two-link manipulator]

APPENDIX C: System Parameter Identification

This appendix contains Procedure used in determining the moments of inertia of the inner and outer links of the manipulator.

A)Inner Link

The moment of interia of the inner link was determined by computing the moment of inertia for each component of the inner link and then adding all the moments of inertia together.
The parallel axis theorem is used extensively in this calculation:

$$I_a = I_g + m(\overline{AG})^2$$

The inner link is comprised of four components: one component is an aluminum tube, there are two clevis style hinge blocks (one at each end of the tube) and one coupling flange that attaches a hinge block to the torque motor.

For the tube, the equation for the moment of inertia of a circular cylindrical shell about its center of gravity is:

$$I_g = \frac{1}{2} mr^2 + \frac{1}{12} m\ell^2$$

For the hinge blocks, the equation for the moment of inertia for a rectangular section was used:

$$I_g = \frac{1}{2} m(b^2 + \ell^2)$$

For the coupling flange, the equation of the moment of inertia about its axis of rotation is:

$$I_{zz} = \frac{1}{2} mr^2$$

The moment of inertia about the axis of rotation of the inner link was computed for each component:

1) Tube	$I_{a1} = 0.003740$ N-m-sec^2
2) Inner hinge block	$I_{a2} = 0.000523$ N-m-sec^2
3) Outer hinge block	$I_{a3} = 0.023140$ N-m-sec^2
4) coupling flange	$I_{a4} = 0.000041$ N-m-sec^2

The total moment of inertia is computed:

$$I_a = I_{a1} + I_{a2} + I_{a3} + I_{a4} = 0.02744 \text{ N-m-sec}^2$$

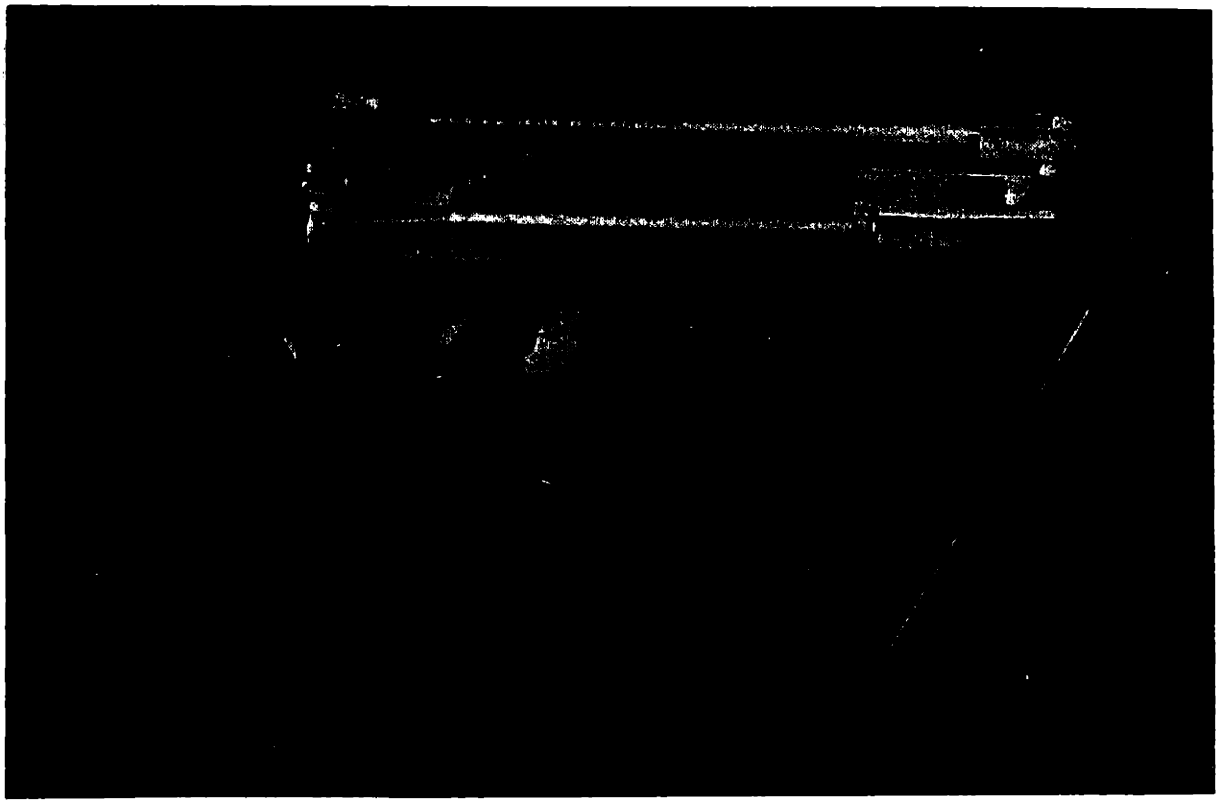
Adding in the torque motor rotor moment of inertia gives the moment of inertia used in the simulation:

$$J_1 = 0.02744 \text{ N-m-sec}^2 + 0.000593\text{N-m-sec}^2$$

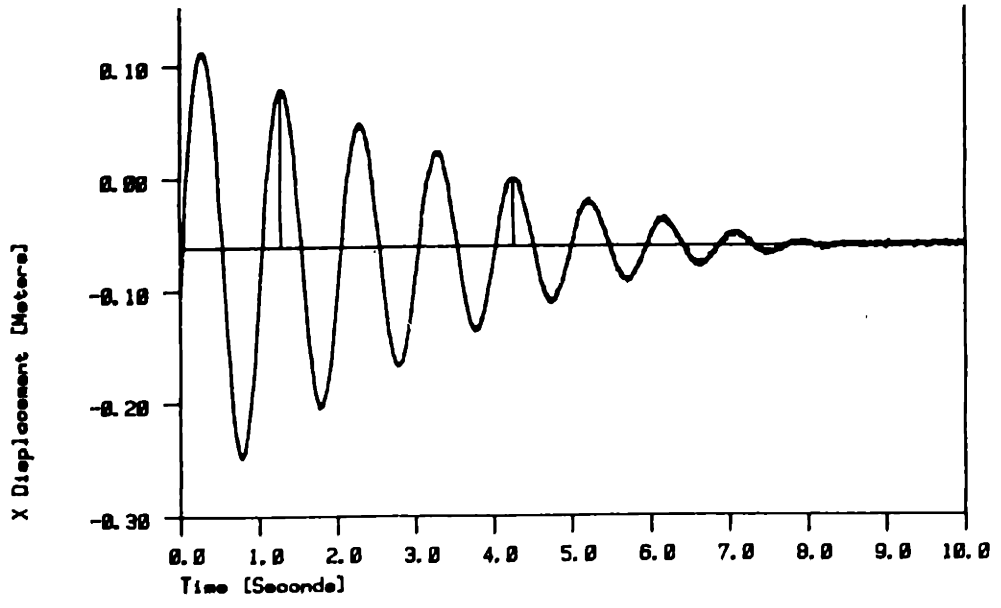
$$= 0.0280 \text{ N-m-sec}^2$$

B) Outer link

The moment of inertia of the outer link was determined by performing a swing test using a system called TRACK. The TRACK system is set up for the automatic accumulation and reduction of three-dimensional kinematic data. Using this system, it is possible to compute both the natural frequency and damping ratio of the outer link swinging free in the gravity field. The photograph below shows the experimental set up.



The figure below shows a typical plot of data from a free swing of the outer link.



From this plot it is possible to compute the logarithmic decrement:

$$\delta = \frac{1}{n} \text{Log}_e \frac{X_0}{X_n}$$

When $n = 3$, $X_0 = 0.137$ meters, $X_3 = 0.0579$ meters

Then $\delta = 0.287$

The damping ratio is computed:

$$\zeta^2 = \frac{\delta^2}{4\pi^2 + \delta^2} = \frac{(0.287)^2}{4\pi^2 + (0.287)^2} = 0.002082$$

$$\zeta = 0.0456$$

The damped natural frequency is computed:

$$\omega_d = 2\pi \text{ (1.0 Hertz)} = 6.28 \text{ rad/sec}$$

The natural frequency is computed:

$$\omega_n = \frac{6.28}{\sqrt{1 - (0.0456)^2}} = 6.29 \text{ rad/sec}$$

The moment of inertia is now computed:

$$I_a = \frac{mgh_2}{\omega_n^2} = \frac{(0.5952 \text{ kg})(9.81 \text{ m/sec}^2)(0.1572 \text{ m})}{(39.56 \text{ 1/sec}^2)}$$

$$I_a = 0.0232 \text{ N-m-sec}^2$$

Adding in the torque motor rotor moment of inertia gives the moment of inertia used in the simulation:

$$J_2 = 0.0232 \text{ N-m-sec}^2 + 0.000593 \text{N-m-sec}^2$$

$$= 0.0238 \text{ N-m-sec}^2$$

APPENDIX D: System Parameter Identification

The endpoint viscosity matrix is:

$$\underset{\sim}{B}_e = \begin{vmatrix} B_1 & B_2 \\ B_3 & B_4 \end{vmatrix}$$

The transformation to joint coordinates is given:

$$\underset{\sim}{B}_\theta = \underset{\sim}{J}^T \underset{\sim}{B}_e \underset{\sim}{J}$$

Where the Jacobian, $\underset{\sim}{J}$, is defined:

$$\underset{\sim}{J} = \begin{vmatrix} -\ell_1 \sin(\theta_1) & -\ell_2 \sin(\theta_2) \\ \ell_1 \cos(\theta_1) & \ell_2 \cos(\theta_2) \end{vmatrix} = \begin{vmatrix} -\ell_1 S_1 & -\ell_2 S_2 \\ \ell_1 C_1 & \ell_2 C_2 \end{vmatrix}$$

Now the matrix multiplication gives:

$$\underset{\sim}{J}^T \underset{\sim}{B}_e \underset{\sim}{J} = \begin{vmatrix} -\ell_1 S_1 & \ell_1 C_1 \\ -\ell_2 S_2 & \ell_2 C_2 \end{vmatrix} \begin{vmatrix} B_1 & B_2 \\ B_3 & B_4 \end{vmatrix} \begin{vmatrix} -\ell_1 S_1 & -\ell_2 S_2 \\ \ell_1 C_1 & \ell_2 C_2 \end{vmatrix}$$

$$= \begin{vmatrix} -\ell_1 S_1 B_1 + \ell_1 C_1 B_3 & -\ell_1 S_1 B_2 + \ell_1 C_1 B_4 \\ -\ell_2 S_2 B_1 + \ell_2 C_2 B_3 & -\ell_2 S_2 B_2 + \ell_2 C_2 B_4 \end{vmatrix} \begin{vmatrix} -\ell_1 S_1 & -\ell_2 S_2 \\ -\ell_1 C_1 & -\ell_2 C_2 \end{vmatrix}$$

$$= \begin{vmatrix} UL & UR \\ LL & LR \end{vmatrix} = \underset{\sim}{B}_\theta$$

Where the diagonal terms are:

$$UL = -\ell_1 S_1 (-\ell_1 S_1 B_1 + \ell_1 C_1 B_3) + \ell_1 C_1 (-\ell_1 S_1 B_2 + \ell_1 C_1 B_4)$$

$$LR = -\ell_2 S_2 (-\ell_2 S_2 B_1 + \ell_2 C_2 B_3) + \ell_2 C_2 (-\ell_2 S_2 B_2 + \ell_2 C_2 B_4)$$

Multiply out the upper left term:

$$UL = \ell_1^2 S_1^2 B_1 - \ell_1^2 S_1 C_1 B_3 - \ell_1^2 S_1 C_1 B_2 + \ell_1^2 C_1^2 B_4$$

If the off-diagonal terms are equal to zero ($B_2 = B_3 = 0$):

$$UL = \ell_1^2 S_1^2 B_1 + \ell_1^2 C_1^2 B_4$$

If the diagonal terms are equal ($B_1 = B_4$)

$$UL = \ell_1^2 B_1 (S_1^2 + C_1^2) = \ell_1^2 B_1 \qquad \text{which is a constant}$$

APPENDIX E: EQSIM listing for Simulation

```
C*****************************************************************'*****
C
C       DYSYS SIMULATION --> SUBROUTINE EQSIM.          2LINKA.FTN
C                     OF
C       A 2-D, 2-LINK PLANAR MECHANISM
C
C       CONTROL of a nonlinear linkage using a "reduced' impedance
C       controller.  F = K(Xv - X) - BV
C       Uniform stiffness and viscosity are desired at the endpoint.
C
C       EQSIM by Ian C. Faye'  21-May-96
C
C*****************************************************************
C
        SUBROUTINE EQSIM
        COMMON T,DT,Y(20),F(20),STIME,FTIME,NEWDT,N,NPNTS,
        1       CONST(20),NVAR,IVARB(20)
C
        DATA CON/57.296/
C
C       FOURTH ORDER SIMULATION FOR LINKAGE INCLUDING CROSS COUPLING
C
        PARAMETER TL1=   .3654,          !LENGTH OF LINK 1
        1         TL2=   .342,           !LENGTH OF LINK 2
        2         TKT1= 0.3726,          !TORQUE SENSITIVITY, MOTOR 1
        3         TKT2= 0.3726,          !TORQUE SENSITIVITY, MOTOR 2
        4         BM1=   .00061,         !MOTOR 1 BEARING LOSSES
        5         BM2=   .00061          !MOTOR 2 BEARING LOSSES
        PARAMETER TIM1= 0.00059,         !MOTOR 1 ROTOR INERTIA
        1         TIM2= 0.00059,         !MOTOR 2 ROTOR INERTIA
        2         H1=    .1927,          !DIST TO CENTROID, LINK 1
        3         H2=    .1572,          !DIST TO CENTROID, LINK 2
        4         TM1=   .6409,          !MASS OF LINK 1
        5         TM2=   .5952           !MASS OF LINK 2
        PARAMETER BEAR=  .0136           !BEARING FRICT BETWEEN LINKS
C       1         AJ11=  .0986,          !LINK 1 INERTIA
C       2         AJ22=  .0238           !LINK 2 INERTIA
        PARAMETER DK1=  400.0,
        1         DK2=    0.0,           ! stiffness
        2         DK3=    0.0,
        3         DK4=  400.0,           !  Desired Endpoint
        4         DB1=   15.0,           !     Conditions
        5         DB2=    0.0,
        6         DB3=    0.0,           ! viscosity
        7         DB4=   15.0
        PARAMETER TCAL = 0.89461,        ! Torque Calibration
        1         AMPG = 3.0             ! Servo-Amp Gain
C
C        F(1) IS ANGULAR ACCELERATION...  LINK 1
C        F(2) IS ANGULAR ACCELERATION...  LINK 2
C        F(3) IS ANGULAR VELOCITY OF LINK 1     !
C        F(4) IS ANGULAR VELOCITY OF LINK 2     !
C                                               ! Angular velocities
C        Y(1) IS ANGULAR VELOCITY OF LINK 1     !      w1 & w2
C        Y(2) IS ANGULAR VELOCITY OF LINK 2     !
C        Y(3) IS ANGULAR POSITION OF LINK 1
C        Y(4) IS ANGULAR POSITION OF LINK 2
C
```

```
C        Y(5) IS X POSITION OF ENDPOINT          ! endpoint location
C        Y(6) IS Y POSITION OF ENDPOINT          !    for link 2
C        Y(7) SERVO AMPLIFIER CURRENT FOR THETA 1
C        Y(8) SERVO AMPLIFIER CURRENT FOR THETA 2
C        Y(9)  IS REFERENCE ANGLE POSITION THETA 1
C        Y(10) IS REFERENCE ANGLE POSITION THETA 2
C        Y(11) IS DESIRED X ENDPOINT POSITION
C        Y(12) IS DESIRED Y ENDPOINT POSITION
C        Y(13) IS X ENDPOINT VELOCITY
C        Y(14) IS Y ENDPOINT VELOCITY
C
C*************************************************************************
C
C  INPUT SOME NEW PARAMETERS
        IF (NEWDT.EQ.-1) THEN
                TYPE 202
 202            FORMAT(' DO YOU WISH TO INPUT REFERENCE ANGLES ? ',$)
                ACCEPT 203,ANREF
 203            FORMAT(A1)
                IF( ANREF.NE.'Y' .AND. ANREF.NE.'y' ) GOTO 204
C
                TYPE 200
 200            FORMAT(' ENTER THE INNER LINK REFERENCE ANGLE ',$)
                READ(5,*) TREF1
                TYPE 201
 201            FORMAT(' ENTER THE OUTER LINK REFERENCE ANGLE ',$)
                READ(5,*) TREF2
        GOTO 205
 204            TREF1 = .8765                   ! 0.3077
                TREF2 = 2.945                   ! 2.3763
 205    CONTINUE
C
C  MASS AT THE HANDLE
                TYPE 206
 206            FORMAT(' DO YOU WISH TO ADD MASS AT THE HANDLE ? ',$)
                ACCEPT 203,ANREF
                IF( ANREF.NE.'Y' .AND. ANREF.NE.'y' ) GOTO 207
C
                TYPE 208
 208            FORMAT(' ENTER THE MASS AT THE HANDLE IN Kg. ',$)
                READ(5,*) TMADD
C
 207    CONTINUE
C
C  SATURATION OF THE SERVO-AMPS
                TYPE 211
 211            FORMAT(' DO YOU WANT THE SERVO-AMPS TO SATURATE ? ',$)
                ACCEPT 203,ANSERV
C
                ICNT1 = 1
C
                TYPE 302
 302    FORMAT(' DO YOU WISH TO CHANGE THE # OF STEPS (100) ? ',$)
                ACCEPT 203,ANREF
                IF( ANREF.NE.'Y' .AND. ANREF.NE.'y' ) GOTO 305
C
                TYPE 300
 300            FORMAT(' ENTER THE # OF STEPS ',$)
```

```
              READ(5,*) NUM
        GOTO 306
 305            NUM = 100
 306    CONTINUE
C
        TFIN = 2*DT*NUM
C
C*******************************************************************************
C
C  CARTESIAN COORDINATE TRANSFORMATIONS
        XSTRT = TL1*COS(Y(3)) + TL2*COS(Y(4))    ! X STARTING POINT
        YSTRT = TL1*SIN(Y(3)) + TL2*SIN(Y(4))    ! Y STARTING POINT
C
        XEND = TL1*COS(TREF1) + TL2*COS(TREF2)   ! X FINAL POINT
        YEND = TL1*SIN(TREF1) + TL2*SIN(TREF2)   ! Y FINAL POINT
C
        DX = (XEND - XSTRT)/NUM
        DY = (YEND - YSTRT)/NUM
        TYPE*,'DX = ',DX,'  DY = ',DY            ! OUTPUT  DX AND DY
C
        XNOM = XSTRT
        YNOM = YSTRT
C
        Y(11) = XSTRT
        Y(12) = YSTRT
C
        XCURR = XNOM
        YCURR = YNOM
C
        TREF1 = Y(3)
        TREF2 = Y(4)
C
        ENDIF
C
C*******************************************************************************
C
C  REFERENCE TRAJECTORY
        IF(NEWDT .EQ. 1)THEN
C
C  UPDATE TRAJECTORY EVERY OTHER TIME STEP
        IF( (ICNT1.AND.'1) .EQ. 1)THEN
                XNOM = XNOM + DX
                YNOM = YNOM + DY
        ENDIF
        ICNT1 = ICNT1 + 1
C
        IF(T .GT. TFIN) XNOM = XEND
        IF(T .GT. TFIN) YNOM = YEND
                XDES = XNOM
                YDES = YNOM
C
C inverse kinematics (not needed for the control in the simulation)
        TP1 = (XDES**2) + (YDES**2) - (TL1**2  - (TL2**2)
        TB1 = 2*TL1*TL2
        PHI = ACOS( TP1/TB1 )   !relative angular displacement
        TO1 = TL2*SIN(PHI)
        TO2 = TL1 + (TL2*COS(PHI))
C
```

```
        TREF1 = ATAN2(YDES,XDES) - ATAN2( TO1,TO2)      !inner link angle
       TREF2 = TREF1 + PHI              ! outer link angle (absolute)
C
        Y(9)  = TREF1
        Y(10) = TREF2
C
        Y(11) = XDES
        Y(12) = YDES
C
        ENDIF
C
C*******************************************************************************
C
C INERTIA MATRIX
        IF(NEWDT .EQ. 1)THEN
        TKG   = TM2 + TMADD
C
        AJ11 = 0.0280 + (TL1**2)*TKG
        AJ12 = (TL1*H2*TM2*COS(Y(4)-Y(3)))
        AJ21 = (TL1*H2*TM2*COS(Y(4)-Y(3)))
        AJ22 = 0.0238 + (TL2**2)*TMADD
C
C MATRIX INVERSION (FOR INERTIA MATRIX)
        D11=AJ22/(AJ11*AJ22-AJ12*AJ21)
        D12=-AJ12/(AJ11*AJ22-AJ12*AJ21)
        D21=-AJ21/(AJ11*AJ22-AJ12*AJ21)
        D22=AJ11/(AJ11*AJ22-AJ12*AJ21)
C
C ANGULAR VELOCITY
        OMGA = Y(1)
        OMGB = Y(2)
C
C FEEDBACK LAW
C
C  DIFFERENCE COMPUTATION
        ALPHA = XDES - XCURR
        BETA  = YDES - YCURR
C
C  TRIG
        S1 = SIN(Y(3))
        S2 = SIN(Y(4))
        C1 = COS(Y(3))
        C2 = COS(Y(4))
C
C  GAIN COMPUTATION
        TOPK = DK1*ALPHA + DK2*BETA
        BOTK = DK2*ALPHA + DK4*BETA
C
        TORK1 = TL1*( BOTK*C1 - TOPK*S1 )
C
        TORK2 = TL2*( BOTK*C2 - TOPK*S2 )
C
C  VELOCITY FEEDBACK CONTRIBUTION
        OFFD = TL1*TL2*(DB1*S1*S2 + DB4*C1*C2 -DB2*((S1*C2)+(S2*C1)) )
C
        TORB1= -(TL1**2)*(DB1*(S1**2) +DB4*(C1**2) -2*DB2*(S1*C1))*OMGA
        1       - OFFD*OMGB
C
```

```
      TORB2= -OFFD*OMGA - (TL2**2)*(DB1*(S2**2) +DB4*(C2**2)
     1

     1       -2*DB2*(S2*C2))*OMGB
C
      AA1 = AMPG*TCAL*(TORK1 + TORB1)
      BB1 = AMPG*TCAL*(TORK2 + TORB2)
C
C SERVO-AMP SATURATION ??
      IF( ANSERV.NE.'Y' .AND. ANSERV.NE.'y' ) GOTO 443
C
      IF(AA1.GT.  24.0) AA1= 24.0
      IF(AA1.LT. -24.0) AA1= -24.0
      IF(BB1.GT.  24.0) BB1= 24.0
      IF(BB1.LT. -24.0) BB1= -24.0
C
 443          CONTINUE
C
C RESTORING TORQUES
      TOR1=TKT1*AA1
      TOR2=TKT2*BB1
C
C SYSTEM ARRAY
      TOP=-(BM1+BEAR)*Y(1)+BEAR*Y(2)+TOR1
      TOP2=TL1*H2*TM2*SIN(Y(4)-Y(3))*(Y(2)**2)
      BOT=+BEAR*Y(1)-(BEAR+BM2)*Y(2)+TOR2
      BOT2=TL1*H2*TM2*SIN(Y(3)-Y(4))*(Y(1)**2
C
C COMPUTE DERIVATIVES
      ENDIF
      F(1)=D11*(TOP+TOP2)+D12*(BOT+BOT2)
      F(2)=D21*(TOP+TOP2)+D22*(BOT+BOT2)
C
      F(3)=Y(1)
      F(4)=Y(2)
C
C CARTESIAN COORDINATE TRANSFORMATION
      Y(5)=TL1*COS(Y(3))+TL2*COS(Y(4))
      Y(6)=TL1*SIN(Y(3))+TL2*SIN(Y(4))
C
      XCURR = Y(5)
      YCURR = Y(6)
C
C  ANGULAR VELOCITIES TO ENDPOINT VELOCITIES
      Y(13) = -TL1*S1*Y(1) -TL2*S2*Y(2)
      Y(14) =  TL1*C1*Y(1) +TL2*C2*Y(2)
C
      Y(7) = AA1       !CURRENT OUTPUT OF SERVO-AMP FOR LINK 1
      Y(8) = BB1       !CURRENT OUTPUT OF SERVO-AMP FOR LINK 2
C
C*****************************************************************
C
      RETURN
      END
C
C*****************************************************************
```

APPENDIX F: Program for Plotting Maximum Force Generation

```
C*******************************************************************************
C  PLOTT.ftn
C
C       PLOTTING ROUTINE
C       FOR PLOTTING MAXIMUM ENDPOINT FORCES    Ian C. Faye'  14-NOV-85
C
C*******************************************************************************
C
        DIMENSION A(2,204),FAR(204),CAL(2,3)
        DIMENSION IPLOT(2),BNDRY(4),IROW(4),ILT(4)
        LOGICAL*1 XLABEL(80),YLABEL(50),TITLE(80)
C
 5      CONTINUE
 110    FORMAT(I1)
C
        IPLT=9
        NROWS=2
        NP = 204
        SCA= 250.       !THE GRAPH SCALING FACTOR
C
14      ITIMPL=0        !Phase Plot
        TYPE*,' THE STATE VARIABLE # FOR THE X-AXIS IS  1'
        IPLOT(1) = 1
        TYPE*,' THE STATE VARIABLE # FOR THE Y-AXIS IS  2'
        IPLOT(2) = 2
C
        NPLT = 2        ! 2 VARIABLES ARE TO BE PLOTTED
        T1MAX= 3.726    !11.178
        T2MAX= 3.726    !5.589  MAXIMUM TORQUES
C
        TYPE*,' ENTER THE INNER LINK ANGLE'
        ACCEPT*,THT1
        TYPE*,' ENTER THE OUTER LINK ANGLE'
        ACCEPT*,THT2
C
        TRIG = (3.1416/180.)
        THT1 = THT1*TRIG
        THT2 = THT2*TRIG
C
        TL1  = .3654    ! INNER LINK LENGTH
        TL2  = .3420    ! OUTER LINK LENGTH
C
        TJ1 = -TL1*SIN(THT1)
        TJ2 = -TL2*SIN(THT2)
        TJ3 =  TL1*COS(THT1)
        TJ4 =  TL2*COS(THT2)
C
        XH  =  TL1*COS(THT1) + TL2*COS(THT2)
        YH  =  TL1*SIN(THT1) + TL2*SIN(THT2)
C
        XE  =  TL1*COS(THT1)
        YE  =  TL1*SIN(THT1)
C
C  DRAW THE MANIPULATOR
        A(1,1) = 0
        A(2,1) = 0
C
        A(1,2) = XE
```

```
      A(2,2) = YF

      A(1,3) = XH
      A(2,3) = YH

      A(1,4) = -9999.0
      A(2,4) = -9999.0
C
C CHECK THE ENDPOINT LOCATION
C
      Q1  =  1.
      IF( XF .GE. 0.0 ) THEN
            Q1 = -1.
      ENDIF
C
C CHECK THE OUTER ANGLE
C
      Q2  = 1.
      IF( THT2 .GE. 180.*TRIG ) THEN
            Q2 = -1.
      ENDIF
C
      Q3  = 1.
      IF( THT2 .LT. 90.*TRIG ) THEN
            Q3 = -1.
      ENDIF
C
      THETA = 0.0
      DTHTF = 1.8
      DTHTR = 0.031415
      EX    = 0.000001
C
C FIRST HALF
      DO 99 I = 1,100
      THETA = (I-1)*DTHTR
      FACTOR = ( (TJ2+(TJ4*TAN(THETA))) / (TJ1+(TJ3*TAN(THETA))+EX) )
C
C FIRST QUADRANT
      IF( THETA .LT. 3.1415/2. ) THEN
      IF( FACTOR .GT. 1.0)THEN
            T2 = -T2MAX
            T1 = T2/FACTOR
      ELSE IF( (FACTOR .LT. 1.0).AND.(FACTOR.GT. 0.0) )THEN
            T1 = -T1MAX*Q3
            T2 = T1*FACTOR
      ELSE IF( (FACTOR .LE. 0.0).AND.(FACTOR.GT. -1.0))THEN
            T1 =  T1MAX*Q2
            T2 =  T1*FACTOR
      ELSE IF( FACTOR .LE. -1.0 )THEN
            T2 = -T2MAX
            T1 =  T2/FACTOR
      ELSE
            T1 = T1MAX
            T2 = T2MAX
      ENDIF
C
C SECOND QUADRANT
      ELSE IF( THETA .GE. 3.1415/2. ) THEN
```

```
        IF( FACTOR .GT. 1.0)THEN
                T2 = -T2MAX*Q1
                T1 = T2/FACTOR
        ELSF IF( (FACTOR .LT. 1.0).AND.(FACTOR.GT. 0.0) )THEN
                T1 = T1MAX
                T2 = T1*FACTOR
        ELSE IF( (FACTOR .LE. 0.0).AND.(FACTOR.GT. -1.0))THEN
                T1 =  T1MAX
                T2 =  T1*FACTOR
        ELSE IF( FACTOR .LE. -1.0 )THEN
                T2 = -T2MAX
                T1 =  T2/FACTOR
        ELSF
                T1 = T1MAX
                T2 = T2MAX
        ENDIF
        ENDIF
C
        XTMP = T1/(TJ1+(TJ3*TAN(THETA))+EX)
        YTMP = XTMP*TAN(THETA)
C
        XF = XTMP/250. + XH
        YF = YTMP/250. + YH
C
        INDEX = I+4
        A(1,INDEX) = XF
        A(2,INDEX) = YF
        FAR(INDEX) = FACTOR

 99     CONTINUE
C
C SECOND HALF
        DO 97 I = 101,200
                THETA = (I-1)*DTHTR
        FACTOR = ( (TJ2+(TJ4*TAN(THETA))) / (TJ1+(TJ3*TAN(THETA))+EX) )
C
C THIRD QUADRANT
C
        IF( THETA .LT. 3.*3.1415/2. ) THEN
C
        IF( FACTOR .GT. 1.0)THEN
                T2 = T2MAX
                T1 = T2/FACTOR
        ELSF IF( (FACTOR .LT. 1.0).AND.(FACTOR.GT. 0.0) )THEN
                T1 = T1MAX*Q3
                T2 = T1*FACTOR
        ELSE IF( (FACTOR .LE. 0.0).AND.(FACTOR.GT. -1.0))THEN
                T1 =  -T1MAX*Q2
                T2 =  T1*FACTOR
        ELSE IF( FACTOR .LE. -1.0 )THEN
                T2 = T2MAX
                T1 =  T2/FACTOR
        ELSE
                T1 = T1MAX
                T2 = T2MAX
        ENDIF
C
```

```
C FOURTH QUADRANT
C
        ELSE IF( THETA .GE. 3.*3.1415/2. ) THEN
        IF( FACTOR .GT. 1.0)THEN
                T2 = T2MAX*Q1
                T1 = T2/FACTOR
        ELSE IF( (FACTOR .LT. 1.0).AND.(FACTOR.GT. 0.0) )THEN
                T1 = -T1MAX
                T2 = T1*FACTOR
        ELSE IF( (FACTOR .LE. 0.0).AND.(FACTOR.GT. -1.0))THEN
                T1 =  -T1MAX
                T2 =  T1*FACTOR
        ELSE IF( FACTOR .LE. -1.0 )THEN
                T2 = T2MAX
                T1 =  T2/FACTOR
        ELSE
                T1 = T1MAX
                T2 = T2MAX
        ENDIF
        ENDIF

C
        XTMP = T1/(TJ1+(TJ3*TAN(THETA))+EX)
        YTMP = XTMP*TAN(THETA)
C
        XF = XTMP/250. + XH
        YF = YTMP/250. + YH
C
        INDEX = I+4
        A(1,INDEX) = XF
        A(2,INDEX) = YF
        FAR(INDEX) = FACTOR
 97     CONTINUE
C
        DO 399 I = 1,204
 399    CONTINUE
C
        IROW(1) = 1
        IROW(2) = 2
        ILT(1)=4
        ILT(2)=-1
        ILT(3)=6
        TYPE*,' DO YOU WANT MANUALLY CHANGE THE LINE TYPE ?'
        READ(5,141) ANSW
        IF(ANSW.NE.'Y'.AND.ANSW.NE.'y') GO TO 61
        WRITE(5,198)
198     FORMAT(' Enter line types; ILT(2) will be the phase plot line,'
     1          /,' ILT(1), ILT(2), ILT(3) ( I2 format) :')
        READ(5,92)(ILT(J),J=1,3)
 92     FORMAT(3I2)
 61     CONTINUE
C
        TYPE*,' X_AXIS LABEL: '
        READ(5,140)(XLABEL(I),I=1,80)
140     FORMAT(80A1)
        TYPE*,' Y-AXIS LABEL (UP TO 46 CHARACTERS):'
        READ(5,150) (YLABEL(I),I=1,46)
150     FORMAT(46A1)
```

```
        TYPE*,' PLOT TITLE:'
        READ(5,140) (TITLE(I),I=1,80)
C
        IAUTSC=1
        TYPE*,' DO YOU WANT MANUALLY SCALE THE PIOT?'
        READ(5,141) ANSW
141     FORMAT(A1)
        IF(ANSW.NE.'Y'.AND.ANSW.NE.'y') GO TO 60
        IAUTSC=0
        WRITE(5,197)
197     FORMAT(' Enter scale values XMIN,XMAX,YMIN,YMAX :')
        READ(5,*)(BNDRY(J),J=1,4)
C
60      ITMDT=1
        ISPEED=15
        CALL CGRAPH(A,NROWS,NP,IROW,ITIMPL,STIME,FTIME,,,ILT,XLABEL,
        1           YLABEL,TITLE,IAUTSC,BNDRY,1,,ISPEED,ITIMDT)
C
C PLOT A CALIBRATION MARK
        CAL(1,1) = (10./250.) - .4
        CAL(2,1) = .1
        CAL(1,2) = -.4
        CAL(2,2) = .1
        CAL(1,3) = -.4
        CAL(2,3) = (10./250.) + .1
C
        CALL CGRAPH(CAL,NPOWS,3,IROW,ITIMPL,STIME,FTIME,,,ILT,XLABEL,
        1           YLABEL,TITLE,IAUTSC,BNDRY,1,,ISPEED,ITIMIT)
C
        DO 199 I=1,NROWS
199     IROW(I)=0
C
        WRITE(5,200)
200     FORMAT(' DO YOU WANT ANY MORE PLOT?'/
        1       ' 1=TIME PIOT',/,' 2=PHASE PLOT',/,' ELSE=NONE',/,
        2       ' YOUR CHOICE: ',$)
        READ(5,110)IREPLY
        IF(IREPLY.NE.1.AND.IREPLY.NE.2) GO TO 80
        GO TO 5
80      CONTINUE
        END
C
C*************************************************************************
```

APPENDIX G: Control Programs

```
/* ********************************************************************** */
/*      SMOVEF.C
This program is intended to implement both position and velocity feedback
on the apparatus.  This will be a completely digital implementation
This program includes the torque command output in the output data file.
This torque command is transformed to give the commanded endpoint force.

The input to the computer is the velocity signal from the tachometer.
The input of position to the computer is from the encoder decoder.

The command Torque signal is then generated based on:

                Tc = [J'*K]*(Xv-L(O)) - [J'*B*J]*W

        where:
                K   is the stiffness in endpoint coordinates
                B   is the viscosity in endpoint coordinates
                Xv  is the virtual trajectory vector
                O   is the absolute angular position vector
                J   is the Jacobian transformation matrix

Limit the torque command output so that we do not ask for more than 24 amps

The structure of the data file for storing endpoint postion (desired and
actual) and the endpoint velocity is:

     [ x_des | y_des | x_act | y_act | x_vel | y_vel | x_for | y_for ]

Each column is 512 points long the program to plot this data is PLOT4B.C

Version 0:
The path from one point to another is a ramp that is 100 points.
The duration of the ramp is .350 seconds.

        Ian C. Faye
*/

#include <all.h>
#define BELL    07
#define PI      3.1415926
#define SCA_TRG (4000/PI)
#define SCA_POS 20000    /* position scale */
#define SCA_VEL 1000     /* velocity scale  */
#define SCA_FOR 200      /* force scale */
#define FDX     100      /* the number of steps during a movement  */
#define FDX1    101      /* FDX + 1 */
#define MSIZ    4096     /* was 3072  for 6x512 :: NOW  8x512  */
#define CSIZ    8192     /* 2 bytes per integer 2x4096   */
#define PNTS    512
#define PNTS1   511      /* PNTS-1 */

float   end_x[FDX1] = 0.0, end_y[FDX1] = 0.0;   /* endpoint location */
int     end_xi[FDX1] = 0,  end_yi[FDX1] = 0;    /* integer values */
int     meas[MSIZ] = 0;          /* buffer to store all the data */
float   stor[8] = 0.0;           /* absolute endpoint target locations */
float   dsp = .01;               /* displacement 1 cm */

float   a_link = .3654;          /* inner link length */
float   b_link = .342;           /* outer link length */
int     c_size = CSIZ;  /* number of bytes to store */
int     save = 0, prnt = 0;
```

```
        /* endpoint dynamics */
float   km1= 0.0, km2= 0.0, km4= 0.0, bm1= 0.0, bm2= 0.0, bm4= 0.0;

#define a2_link (.3654*.3654)    /* inner link length squared */
#define b2_link (.342*.342)      /* outer link length squared */
#define ab_link (.3654*.342)     /* inner and outer link lengths */

/* coefficients for impedance control */
float   cof1k= 0., cof2k= 0., cof3k= 0., cof4k= 0., cofk= 0.;
float   cof5k= 0., cof6k= 0., cof7k= 0., cof8k= 0., cof9k= 0.;
float   cof1b= 0., cof2b= 0., cof3b= 0., cof4b= 0., cofb= 0.;
float   cof5b= 0., cof6b= 0., cof7b= 0., cof8b= 0., cof9b= 0.;

main()
{
        char    dummy;
        char    *file_name = "d11:cosalf.dat";  /* the lookup file */
        int     base, count, j, ans;
        float   dt, dx, t, tf_n, x, cn_x, cn_y, tht_a, tht_b, a0, b0;
        float   x_strt, y_strt;
        float   a_des, b_des;
        float   save_tf_n;

/* set the endpoint dynamic conditions */
        km1 = 400.;
        km2 = 0.0;
        km4 = 400.;

        bm1 = 15.0;
        bm2 = 0.0;
        bm4 = 15.0;

/* initialize everything */
        printf(" Running SMOVEK.C;  version 0: 100 point ramp\n");
        str_set();      /* store endpoint locations */

        csr_set();      /* csr set to enable the relays */
        dac_put(0,0);   /* zero the D/A converters      */

/* initialize the extended memory */
        xb_init();
        xb_rfile(file_name);

/* pause to allow the user to turn on the relays */
        printf(" enter an [s] to start: \n");
        scanf(" %c",&dummy);
        if(dummy == 's')
                {
        off_out();
        off_com(&a0,&b0);   /* find angular offsets */

/* The time base */
        base = 4;
        count = 35;        /* this combination makes dt = .00350 sec */
        dt = .00350;
        t  = 0.0;
        tf_n = FDX*dt;          /* final time in seconds */
        ck_set(base,count);     /* set the clock */

/* start the program */
```

```
while(ans != 'e')
        {
        ang_com(a0, b0, &tht_a, &tht_b );  /* find current position */
        jconv(tht_a, tht_b, &x_strt, &y_strt); /* cartesian endpoint*/
                                              /* location */
        if( ans == 'm' )
                menu();

        if( ans == 'p' )
                prnt = 1;

        if( ans == 'n' )
                prnt = 0;

        if( ans == 'x' )
                {
                save_tf_n = tf_n;
                tf_n = dt,
                printf("    in step mode now!\n");
                }

        if( ans == 'v' )
                {
                tf_n = save_tf_n;
                printf("    Out of step mode now!\n");
                }

        if( ans == 's' )
                show_enc(a0, b0);

        if( ans == 'z' )
                {
                fileit(),
                save = 1;
                }
        if( ans == 'c' )
                {
                endfile();
                save = 0;
                }

        if( ans == 'q' )
                distance();

        if( (ans >= '1') && (ans <= '4'))
        dynamics(FDX, km1, km2, km4, bm1, bm2, bm4,
        x_strt,y_strt,stor[(ans-1)*2-96],stor[ans*2-97], tf_n,dt);

        if( ans == 'l' )
        dynamics(FDX, km1, km2, km4, bm1, bm2, bm4,
        x_strt, y_strt, (x_strt + dsp ), (y_strt+0.0), tf_n, dt),

        if( ans == 'r' )
        dynamics(FDX, km1, km2, km4, bm1, bm2, bm4,
        x_strt, y_strt, (x_strt - dsp ), (y_strt+0.0), tf_n, dt);

        if( ans == 'i' )
        dynamics(FDX, km1, km2, km4, bm1, bm2, bm4,
        x_strt, y_strt, (x_strt+0.0), (y_strt - dsp ), tf_n, dt);

        if( ans == 'o' )
```

```
                dynamics(FDX, km1, km2, km4, bm1, bm2, bm4,
                x_strt, y_strt, (x_strt+0.0), (y_strt + dsp ), tf_n, dt);

                if( ans == 'd' )
                        {
                        change();
                        printf(
  "k11= %6.2f, k21= %6.2f, k22= %6.2f, b11= %6.2f, b21= %6.2f, b22= %6.2f\n",
                        km1, km2, km4, bm1, bm2, bm4 );
                        }

                ans = ttyi();
                }
                }
        printf("%c",BELL);

        puts(" Bye");
}
/* SUBROUTINES ************************************************************ */

menu()
{
        printf("  \n");
        printf("(m) will print this menu.\n");
        printf("  \n");
        printf("(p) will enable the print of stored data.\n");
        printf("(n) will disable the print of stored data.\n");
        printf("(x) puts the program in 'step' mode.\n");
        printf("(v) takes the program out of 'step' mode.\n");
        printf("(s) shows the current encoder values (offset).\n");
        printf("(z) will initialize a data file.\n");
        printf("(c) will close the data file.\n");
        printf("(1-4) will move the arm positions 1-4.\n");
        printf("(l) will move the arm endpoint to the left.\n");
        printf("(r) will move the arm endpoint to the right.\n");
        printf("(i) will move the arm endpoint inward.\n");
        printf("(o) will move the arm endpoint outward.\n");
        printf("(d) will allow the user to change dynamics.\n");
        printf("(q) will allow the user to change displacement distance.\n");
        printf("(e) will exit the program.\n");
}

/* stiffness and viscosity subroutine */
dynamics(num, k1, k2, k4, b1, b2, b4, x_strt,y_strt,x_end,y_end,tt,dt)
int     num;
float   k1, k2, k4, b1, b2, b4;
float x_strt,y_strt,x_end,y_end,tt,dt;
{
        char    dummy;
        float   cos1, sin1, cos12, sin12, both1;
        float   cos2, sin2, cos22, sin22, both2;
        float   sin1x2, cos1x2, both1x2, both2x1;

        float   k_top, k_bot, a_des, b_des, x_des, y_des, a_tmp, b_tmp;
        float   ta_tmp, tb_tmp, den_j;
        int     icnt = 0, q = 0, val = 0, val_tmp = 0, acnt = 0;

        int     tor_ai, tor_bi, ang_ai, ang_bi, off_ai, off_bi;
        float   tor_ka, tor_kb, tor_ba, tor_bb;
        float   omga_a, omga_b, vel_cal, tor_cal, off_diag, aw_tmp, bw_tmp;
        long int cnn_at = 0, cnn_bt = 0;
```

```
/* what we need to make a movement */
        int     k, maa, idx= 0, tmpi;
        int     base, count, i, j;
        float   t, tht_a, tht_b, dx, dy, tfin, x_pnt, y_pnt, alpha, beta;

/* zero the D/A converters */
        dac_put(0,0);

        dx = (x_end - x_strt)/num;
        dy = (y_end - y_strt)/num;
        t  = 0.0;
        tfin = tt;

                end_x[0] = x_strt;
                end_y[0] = y_strt;
                end_xi[0] = end_x[0]*SCA_POS;
                end_yi[0] = end_y[0]*SCA_POS;

/* precomputation of endpoint locations for the control loop */
        for(i=1; i < FDX1 ;i++)
                {
                end_x[i] = end_x[(i-1)] + dx;
                end_y[i] = end_y[(i-1)] + dy;
                end_xi[i] = end_x[i]*SCA_POS;
                end_yi[i] = end_y[i]*SCA_POS;
                }

                i = 0;
/* precomputation of some of the parameters for the contol loop */
        vel_cal = (20./4096.)*(3.2258); /* rad/sec per A/D unit */
        tor_cal = (4096./20.)*(.89461); /* A/D unit per N-m     */

        coeff(k1, k2, k4, b1, b2, b4);

                off_ai = 2044;  /* offset of A/D channel 0 */
                off_bi = 2045;  /* offset of A/D channel 1 */

        printf(" \n");
        printf("             WORKING \n"),
        printf("%c",BELL);

/* CONTROL LOOP ************************************************************ */
/* real time loop */

/*      for( ; ; )      */

        ck_clear(),
        ck_go();
        while(ck_wait())
        {
/* find the joint angles and do the TRIG lookup */
        cos_a_find( &cos1, &sin1, &cos12, &sin12, &both1, &ang_ai);
        cos_b_find( &cos2, &sin2, &cos22, &sin22, &both2, &ang_bi);

/* endpoint trajectory assignment, store the desired trajectory */
/*      if( icnt & 01 ) acnt++;          move every other dt */
        if( (t += dt) <= tfin )
                {
                x_des = end_x[ icnt ];
                y_des = end_y[ icnt ];
```

```
                meas[idx++] = end_x1[icnt];
                meas[idx++] = end_y1[icnt];
                }
        else
                {
                x_des = x_end;
                y_des = y_end;
                meas[idx++] = end_x1[FDX];
                meas[idx++] = end_y1[FDX];
                }

/* compute the common TRIG multiplies */
        sin1x2  = sin1*sin2;
        cos1x2  = cos1*cos2;
        both1x2 = cos1*sin2;
        both2x1 = cos2*sin1;

/* Forward kinematics, compute endpoint location from joint angles */
        x_pnt = a_link*cos1 + b_link*cos2;
        y_pnt = a_link*sin1 + b_link*sin2;

/* store the joint angles;  measure and store the angular velocities */
        meas[idx++] = ang_a1;
        meas[idx++] = ang_b1;
                adc_in(0,&val);          /* A/D channel 0 */
                val_tmp = val-off_a1;
                omga_a = (val_tmp)*vel_cal;
        meas[idx++] = val_tmp;

                adc_in(1,&val);          /* A/D channel 1 */
                val_tmp = val-off_b1;
                omga_b = (val_tmp)*vel_cal;
        meas[idx++] = val_tmp;

/* Error computation; desired minus actual endpoint position */
        alpha = x_des - x_pnt;
        beta  = y_des - y_pnt;

/* The gain computation */
        k_top = (k1*alpha + k2*beta);
        k_bot = (k2*alpha + k4*beta);

/* Torque contribution from stiffness */
        tor_ka = a_link*( (cos1*k_bot) - (sin1*k_top) );
        tor_kb = b_link*( (cos2*k_bot) - (sin2*k_top) );

/* viscosity */
        off_diag = cof4b*sin1x2 + cof5b*cos1x2 - cofb*(both1x2 + both2x1);

        tor_ba = -(cof1b*sin12 + cof2b*cos12 - cof3b*both1)*omga_a
                        - off_diag*omga_b;

        tor_bb =          - off_diag*omga_a
                -(cof7b*sin22 + cof8b*cos22 - cof9b*both2)*omga_b;

        tor_a1 = (tor_cal*(tor_ka + tor_ba));
        tor_b1 = (tor_cal*(tor_kb + tor_bb));

/* limit torque command output so that we do not ask for more than 24 amps */
        if(tor_a1 > 1638) tor_a1 = 1638;
        if(tor_a1 < -1638) tor_a1 = -1638;
```

```
        if(tor_bi > 1638) tor_bi = 1638;
        if(tor_bi < -1638) tor_bi = -1638,

/* store the command torques */
        meas[idx++] = tor_ai;
        meas[idx++] = tor_bi;

        dac_put(tor_ai,tor_bi);

/* update the loop counter */
                if( icnt++ >= PNTS1 ) break,
        }
        dac_put(0,0);

/* LOOP END ************************************************************** */

        printf("%c",BELL),
        printf("%c",BELL);
        printf("            DONE \n");


if( save == 1 ) /* store the buffer only if a file is open */
        {
        /* Transformation from joint coordinates to endpoint coordinates
                for the storage matrix  meas[ ] */

        for( i= 0; i<MSIZ; i+=8)
                {
        /* position */
                a_tmp = (meas[ i+2 ])/SCA_TRG;  /* theta 1  in radians*/
                b_tmp = (meas[ i+3 ])/SCA_TRG;  /* theta 2  in radians*/
                        /* Forward kinematics */
                meas[ i+2 ] = SCA_POS*(a_link*cos(a_tmp) + b_link*cos(b_tmp)),
                meas[ i+3 ] = SCA_POS*(a_link*sin(a_tmp) + b_link*sin(b_tmp));
        /* velocity */
                aw_tmp = (meas[ i+4 ]*vel_cal);          /* omega 1 */
                bw_tmp = (meas[ i+5 ]*vel_cal);          /* omega 2 */
                        /* Transformation */
                meas[ i+4 ] = -SCA_VEL*(aw_tmp*a_link*sin(a_tmp)
                                        + bw_tmp*b_link*sin(b_tmp));
                meas[ i+5 ] = SCA_VEL*(aw_tmp*a_link*cos(a_tmp)
                                        + bw_tmp*b_link*cos(b_tmp));
        /* command torque => endpoint force */
                ta_tmp = (meas[ i+6 ]/tor_cal);
                tb_tmp = (meas[ i+7 ]/tor_cal);
                        /* determinant of the Jacobian */
                        den_j = a_link*b_link*((cos(a_tmp))*sin(b_tmp)
                                - (cos(b_tmp))*sin(a_tmp) );
                        /* Transformation */
                meas[ i+6 ] = SCA_FOR*( (1./den_j)*(ta_tmp*b_link*cos(b_tmp)
                                        - tb_tmp*a_link*cos(a_tmp)) );
                meas[ i+7 ] = SCA_FOR*( (1./den_j)*(ta_tmp*b_link*sin(b_tmp)
                                        - tb_tmp*a_link*sin(a_tmp)) );
                }

        /* store the meas[] buffer, only if the real time is completed */
                if( icnt >= PNTS1 )
                {
                printf(" Store the buffer\n");
                        istore(meas, c_size);
                printf("%c",BELL);
```

```
                )
        )

if( prnt == 1 ) /* print the data to store only if we want to */
        {
        for( q=0; q< MSIZ; q+= 8)
        printf("idx= %d   a= %d, b= %d,  va= %d, vb= %d, da= %d, db= %d\n",
        (q/8),meas[q], meas[q+1], meas[q+2], meas[q+3], meas[q+4], meas[q+5]);
        scanf(" %c",&dummy);
        }

}

/* compute the quantities that are constant in the real time loop */
coeff(k1, k2, k4, b1, b2, b4)
float   k1, k2, k4, b1, b2, b4;
{
        cof1k = a2_link*k1;
        cof2k = a2_link*k4;
        cof3k = 2*a2_link*k2;
        cof4k = ab_link*k1;
        cofk  = ab_link*k2;
        cof5k = ab_link*k4;
        cof6k = 2*ab_link*k2;
        cof7k = b2_link*k1;
        cof8k = b2_link*k4;
        cof9k = 2*b2_link*k2;

        cof1b = a2_link*b1;
        cof2b = a2_link*b4;
        cof3b = 2*a2_link*b2;
        cof4b = ab_link*b1;
        cofb  = ab_link*b2;
        cof5b = ab_link*b4;
        cof6b = 2*ab_link*b2;
        cof7b = b2_link*b1;
        cof8b = b2_link*b4;
        cof9b = 2*b2_link*b2;
}


/* store the 4 locations for the endpoint */
str_set()
{
        stor[0] = 0.1016;       /* position 1 */
        stor[1] = 0.3476;

        stor[2] = -0.1016;      /* position 2 */
        stor[3] =  0.3476;

        stor[4] = -0.1016;      /* position 3 */
        stor[5] =  0.5000;

        stor[6] = 0.1016;       /* position 4 */
        stor[7] = 0.5000;
}

/* allow the user to change the effective endpoint dynamics */
change()
{
        printf("k11 was %6.2f, enter new k11: \n", km1);
```

```
        scanf("%f", &km1);

        printf("k21 was %6.2f, enter new k21 (same as k12): \n", km2);
        scanf("%f", &km2);

        printf("k22 was %6.2f, enter new k22: \n", km4);
        scanf("%f", &km4);

        printf("b11 was %6.2f, enter new b11: \n", bm1);
        scanf("%f", &bm1);

        printf("b21 was %6.2f, enter new b21 (same as b12): \n", bm2);
        scanf("%f", &bm2);

        printf("b22 was %6.2f, enter new b22: \n", bm4);
        scanf("%f", &bm4);
}

/* allow the user to change the relative displacement distance */
distance()
{
        printf("relative displacement was %4.3f meters\n", dsp);
        printf("enter the new displacement in meters:\n");
        scanf("%f", &dsp);

        if(dsp >= .1) dsp = .1;
        printf("relative displacement is now %4.3f meters\n", dsp);
}

show_enc(a_0, b_0)
float   a_0, b_0;
{
        float tht_a, tht_b;
        int ans, ang_ai, ang_bi;
        while(ans != 'q')
                {
                ang_com(a_0, b_0, &tht_a, &tht_b );

                ang_ai = SCA_TRG*tht_a;
                ang_bi = SCA_TRG*tht_b;

        printf("offset scaled angles a= %d units, b= %d units    q(uit)\r",
                ang_ai, ang_bi);

                ans = ttyi();
                }
}

/* ************************************************************************** */
```

```
/*      INTFAC.C
This is the program that contains all the subroutines
that computer uses to INTERFACE with the analog world.

Configured to work with the PDP 11/73 in Emilio Bizzi's Lab.

******* ROUTINES ***************************************************************

tty1()                              returns keyboard input.
off_out()                           outputs offsets to the digital interface.
csr_set()                           sets the csr bit on the parallel interface,
                                    so that we can turn on the relays.
csr_off()                           sets the csr bit on the parallel interface,
                                    to turn off the relays.
dac_out(value_a,value_b)            outputs values on corresponding DACs.
dac_put(value_a,value_b)            outputs offset values on corresponding DACs.
dac_rad(value_a,value_b)            outputs RADIAN values on corresponding DACs.
                                    (0 rad = -10 volts,  PI rad = +10 volts).
adc_in(chann,value)                 returns a value from the A/D converter
                                    for channel chann.
off_com(a0,b0)                      finds angle offsets based on orientation
ang_com(a0,b0,fa,fb)                finds angles (radians) adjusted by the offset
ang_cim(fa,fb)                      finds angles (integers) adjusted by the offset
end_find(x_end,y_end)               finds link angles, then computes cartesian
                                    endpoint coordinates.
cos_a_find(cos, sin, cos2, sin2, both, ang)     extended memory read to find
                                    trig values for theta 1
cos_b_find(cos, sin, cos2, sin2, both, ang)     extended memory read to find
                                    trig values for theta 2
jconv(tht_a,tht_b,cn_x,cn_y)        converts joint angles to endpoint coordiantes

*******************************************************************************

Ian Faye    25 October 1985
            18 December 1985
*/

#include        "stdio.h"
#include        "std.h2"
#include        "ttyrt.h1"
#include        "rt11.h"
#include        "hardwa.h"

float pos_a = 0.9, pos_b = 91.634;      /* offset angles for calibration   */
float lnk_a = .3654, lnk_b = .342;
GLOBAL int  a0i = 0, b0i = 0;
GLOBAL float tht_a[2] = 0.0, tht_b[2] = 0.0, out[5] = 0.0;

/* define some scaling conversion factors */
#define PI              3.1415926
#define TRIG_SCALE      (PI/4000)       /* for the encoders */
#define DEGR_SCALE      (180.0/4000)    /* for the encoders */
#define RAD_SCALE       (PI/180.0)
#define DAC_CON         (4096./PI)
#define DAC_CR          (4000./PI)

/* the keyboard input */
METACH  tty1()
{
        register int c, hold;
```

```
        nold = JSW;
        JSW |= SPECIAL;
        if((c = emt(TTYIN,0))<0)
                c = EOF;
        JSW = nold;
        return(c);
}

off_out()
{
        int a,b,a0,b0;

        a       =       017777 & *v11a_in;      /* read theta 1 */
        b       =       017777 & *v11b_in;      /* read theta 2 */

        a0      =       (pos_a/DEGR_SCALE) - a;
        b0      =       (pos_b/DEGR_SCALE) - b;

        printf(" \n");
        printf(" Angle offsets sent out a: %d,  b: %d\n",a0,b0);

        *v11a_out = a0;
        *v11b_out = b0;
}

csr_set()
{
        *csra_out = 2;  /* set csr1 on theta 1
                                        to allow the relays to be active */
}

csr_off()
{
        *csra_out = 0;  /* set csr1 on theta 1
                                        to turn the relays off */
}

dac_out(des_a, des_b)
float des_a, des_b;
{
        int     out_a, out_b;
        out_a   =  DAC_CON*(des_a - .00 ) + 2048; /* 20 bits per PI rad */
        out_b   =  DAC_CON*(des_b + .00 ) + 2048; /* 4096. = 20*2048/10 */
        *dac_buf =  (out_a ) &   07777;  /* MASK with AND, DAC X */
        *dac_buf =  (out_b ) | 0100000;  /* MASK with OR,  DAC Y */
}

dac_rad(des_a, des_b)
float des_a, des_b;
{
        int     out_a, out_b;
        out_a   =  DAC_CR*(des_a - .00 ); /* 20 bits per PI rad */
        out_b   =  DAC_CR*(des_b + .00 ); /* 4096. = 20*2048/10 */
        *dac_buf =  (out_a ) &   07777;  /* MASK with AND, DAC X */
        *dac_buf =  (out_b ) | 0100000;  /* MASK with OR,  DAC Y */
}

dac_put(des_ai, des_bi)
int des_ai, des_bi;
{
        /* Normally the offset should be about 2048, but DAC A is off
```

```
            so it has an offset of about 12 units (to get zero volts)  */

        *dac_buf =  (des_a1 + 2060) &   07777;  /* MASK with AND, DAC X */
        *dac_buf =  (des_b1 + 2049) | 0100000;  /* MASK with OR,  DAC Y */
}

adc_in(chann,value)
int     *value, chann;
{
        *ad_csr = (chann << 8);
        while( ! (*ad_csr & AD_DONE ) )
                ;
                *value = (*ad_buf & 07777);
}

/* computes the offset angles (in RADIANS) */
off_com(a0,b0)
float *a0, *b0;
{
        int     a,b;
        a       =       017777 & *v11a_in;      /* read theta 1 */
        b       =       017777 & *v11b_in,      /* read theta 2 */

        *a0     =       pos_a*RAD_SCALE - a*TRIG_SCALE;
        *b0     =       pos_b*RAD_SCALE - b*TRIG_SCALE;

        a01     =       *a0 / TRIG_SCALE;
        b01     =       *b0 / TRIG_SCALE,

}

/* finds the angles (in radians) adjusted by the computed offset angles */
ang_com(a0,b0,fa,fb)
float a0, b0, *fa, *fb;
{
        int     a,b;
        a       =       017777 & *v11a_in;      /* read theta 1 */
        b       =       017777 & *v11b_in;      /* read theta 2 */

        *fa     =       TRIG_SCALE * a + a0;
        *fb     =       TRIG_SCALE * b + b0;
}

/* finds the angles (integers) adjusted by the computed offset angles */
ang_cim(fa,fb)
int *fa, *fb;
{
        *fa = ((017777 & *v11a_in) + a01);
        *fb = ((017777 & *v11b_in) + b01),
}

/* based on the current joint angles, find the endpoint coordinates
   this routine requires off_com(a0,b0) to be used first. */
end_find(x_end,y_end)
float *x_end,*y_end;
{
        xb_read( &lnt_a[0], 8, ((017777 & *v11a_in) + a01)*8L );
        xb_read( &lnt_b[0], 8, ((017777 & *v11b_in) + b01)*8L );

        *x_end = lnk_a*lnt_a[0] + lnk_b*lnt_b[0];
        *y_end = lnk_a*lnt_a[1] + lnk_b*lnt_b[1];
```

```
}

/* based on the current joint angles, find and return to the calling
   program the cosine and sine values for the angles.
   this routine requires off_com(a0,b0) to be used first. */
cos_find( cosin1, cosin2, sine1, sine2)
float *cosin1, *cosin2, *sine1,*sine2;
{
        xb_read( &tht_a[0], 8, ((017777 & *v11a_in) + a0i)*8L );
        xb_read( &tht_b[0], 8, ((017777 & *v11b_in) + b0i)*8L );

        *cosin1  = tht_a[0];
        *cosin2  = tht_b[0];
        *sine1   = tht_a[1];
        *sine2   = tht_b[1];
}

/* based on the current joint angles, find and return to the calling
   program the cosine and sine values and their various combinations
   cosine*cosine, sine*sine and cosine*sine  for the angles.
   This routine requires off_com(a0,b0) to be used first. */
cos_a_find(cosine, sine, cosin2, sine2, both, ang)
int     *ang;
float   *cosine, *cosin2, *sine,*sine2, *both;
{
        *ang = ((017777 & *v11a_in) + a0i);
        xb_read( &out[0], 20, (*ang)*20L );

        *cosine  = out[0];
        *sine    = out[1];
        *cosin2  = out[2];
        *sine2   = out[3];
        *both    = out[4];
}
cos_b_find(cosine, sine, cosin2, sine2, both, ang)
int     *ang;
float   *cosine, *cosin2, *sine,*sine2, *both;
{
        *ang = ((017777 & *v11b_in) + b0i);
        xb_read( &out[0], 20, (*ang)*20L );

        *cosine  = out[0];
        *sine    = out[1];
        *cosin2  = out[2];
        *sine2   = out[3];
        *both    = out[4];
}

/* this routine determines the endpoint position given link angles */
jconv(tht_a, tht_b, cn_x, cn_y)
float tht_a, tht_b, *cn_x, *cn_y;
{
        float x, y;
        *cn_x = lk_a*cos(tht_a) + lk_b*cos(tht_b);
        *cn_y = lk_a*sin(tht_a) + lk_b*sin(tht_b);
}
```

```
/*              CLOCK.C

                Real Time Clock functions

                    SMI 31-JAN-85

*/


#define CKCSR 0170420           /* RT Clock CSR address */
#define CKBPR CKCSR+2           /* RT Clock Buffer-Preset address*/

/* Clock RATE  ( + repeated interval mode) */
#define MHZ1  012               /* 1 MHz */
#define KHZ100 022              /* 100 KHz */
#define KHZ10 032               /* 10 KHz */
#define KHZ1 042                /* 1 KHz */
#define HZ100 052               /* 100 Hz */

#define GO 1                    /* Start clock */
#define OVF 0200                /* Overflow set */

static unsigned *ck_status = CKCSR;
static unsigned *ck_bpr = CKBPR;

ck_set (base_rate,ncounts)
/*
    FUNCTION  Set real time clock rate (Repeated interval mode)
*/
int base_rate,ncounts;
{
        int rate;

        switch (base_rate)
                {
                case 2:
                        rate = HZ100;
                        break;
                case 3:
                        rate = KHZ1;
                        break;
                case 4:
                        rate = KHZ10;
                        break;
                case 5:
                        rate = KHZ100;
                        break;
                case 6:
                        rate = MHZ1;
                        break;
                default:
                        puts(" incorrect clock rate ");
                        rate = 0;
                        break;
                }
        *ck_status = rate;
        if(ncounts < 0)
                {
                puts("incorrect clock preset");
                *ck_bpr = 0;
                }
```

```
        else
                *ck_bpr = - ncounts;
        return;
}

ck_go ()
/*
    FUNCTION : Start the clock.
*/
{
        *ck_status |= GO;
        return;
}

ck_stop()
/*
    FUNCTION : Stop the clock
*/
{
        *ck_status &= ~GO;
        return;
}

ck_clear()
/*
    FUNCTION : Clear the clock overflow
*/
{
        *ck_status &= ~OVF;      /*clear overflow*/
}


int ck_wait ()
/*
    FUNCTION : Wait for a clock pulse. Signal if rate is too fast
    RETURNS :  0 if rate is too fast
               1 if rate is OK
*/
{
        if( *ck_status & OVF )
                {
                puts(" Sampling too fast !");
                return(0),
                }
        else
                {
                while( !(*ck_status & OVF))
                        ;
                *ck_status &= ~OVF;      /*clear overflow*/
                return(1);
                }
}

delay(n)
/*
    FUNCTION : Wait n clock ticks.
    CALLS ck_go(),ck_wait(),ck_stop
*/
int n;
{
        int count;
```

```
/*****************************************************************************

                File operations          FILEOP.C

                Operations that are common to almost all
                operations that we would ever do with a file:
                        A. Create the data file
                        B. Write the data out to the file
                        C. Close the data file
                        D. Open the data file
                        E. Read the data file
                C_SIZE  (SAMPLES*2)  for integer numbers
                C_SIZE  (SAMPLES*4)  for floating point numbers

                                                11/29/85

*****************************************************************************/

#include "std.h"
#include "useful.h"
#include "rt11.h"

int fd = 3, ij = 0, ik = 0;

char *file_name = "dl2:testio.dat";

/*** ROUTINES ***************************************************************/

/*Create the data file*/
BOOL fileit()
        {
        char *fname;

        putfmt("Store data in file named: \n");
        getlin(file_name, 80);
        fname = file_name;
        if((fd = create(fname, WRITE, 1)) < 0)
                errfmt("ERROR: can't create %p: %i\n", fname, fd);
        else
                errfmt("created: %p: %i\n", fname, fd);
        return(SUCCESS);
        }

/* Write the data out, floating point */
BOOL fstore(buffer,c_size)
float   *buffer;
int     c_size;
        {
        if((ij = write(fd, buffer, c_size)) != c_size)
                errfmt("ERROR: write(%i, %ui, %i): %i\n", fd, buffer,
                                                        c_size, ij);
        return(SUCCESS);
        }

/* Write the data out, integer */
BOOL istore(buffer,c_size)
int     c_size, *buffer;
        {
        if((ik = write(fd, buffer, c_size)) != c_size)
                errfmt("ERROR: write(%i, %ui, %i): %i\n", fd, buffer,
                                                        c_size, ik);
```

```
        return(SUCCESS);
        }


 /*Close the data file*/
BOOL endfile()
        {
        if((ij = close(fd)) < 0)
                errfmt("ERROR: close( %l ): %l\n", fd, ij);
        else
                errfmt("close( %l ): %l\n", fd, ij);
        return(SUCCESS);
        }

/* Open old file */
BOOL get_file()
{
        char *fname;

        putfmt("Get data stored in file named: \n");
        getlin(file_name, 80);

        fname = file_name;
        if((fd = open(fname, READ, 1)) < 0)
                errfmt("ERROR: can't open %p: %l\n", fname, fd);
        else
                errfmt("open: %p: %l\n", fname, fd);
        return(SUCCESS);
}

/* Read the data in, floating point */
BOOL fr_data(buffer,c_size)
float   buffer;
int     c_size;
{
        if((ij = read(fd, buffer, c_size)) != c_size)
                errfmt("ERROR: read(%l, %ul, %l): %l\n", fd, buffer,
                                                        c_size, ij);
        return(SUCCESS);
}

/* Read the data in, integer */
BOOL ir_data(buffer,c_size)
int     c_size, buffer;
{
        if((ik = read(fd, buffer, c_size)) != c_size)
                errfmt("ERROR: read(%l, %ul, %l): %l\n", fd, buffer,
                                                        c_size, ik);
        return(SUCCESS);
}

/*************************************************************************/
```

```
/* ********************************************************************
                    HARDWA.H
This program includes all the hardware address locations and
address vectors for the interfaces in the PDP 11/73 computer
in Emilio Bizzi's Lab.

The parallel interfaces are the DEC DRV-11 equivalent

The A/D board is a Data Translation DT1761 SE
The D/A board is a Data Translation DT1761 SE

The real time clock is a MODEL 306 Programmable Real Time Clock
made by Grant Technology Systems Corporation.

Ian Faye, October 25 1985
*/


/* parallel interface */
#define DRV11A_IN       0767764         /* inner link encoder */
#define DRV11B_IN       0767774         /* outer link encoder */

#define DRV11A_OUT      0767762         /* inner link output offset */
#define DRV11B_OUT      0767772         /* outer link output offset */

#define CSRA_OUT        0767760
#define CSRB_OUT        0767770

#define DRV11_OUT       0767752         /* digital output for MDACs */

#define CSR1_OUT        0767750

/*
  define the vectors that point to the hardware locations
*/

int     *v11a_in        =       DRV11A_IN
int     *v11b_in        =       DRV11B_IN

int     *v11a_out       =       DRV11A_OUT
int     *v11b_out       =       DRV11B_OUT

int     *csra_out       =       CSRA_OUT
int     *csrb_out       =       CSRB_OUT

int     *v11_out        =       DRV11_OUT

int     *sr1_out        =       CSR1_OUT


/* a-to-d converter */
#define AD_CSR          0777000
#define AD_CHN          AD_CSR + 1
#define AD_BUF          AD_CSR + 2
#define AD_IVEC         0130
#define AD_PRI          AD_IVEC + 2
#define AD_GO           1
#define AD_DONE         0200
#define AD_ERROR        0100000

/*
```

```
  define the vectors that point to the hardware locations
*/

int     *ad_csr         =       AD_CSR;
int     *ad_buf         =       AD_BUF,
int     *ad_ivec        =       AD_IVEC;
int     *ad_pri         =       AD_PRI,


/* d-to-a converter */
#define DAC_BUF         AD_CSR + 2

/*
 define the vectors that point to the hardware locations
*/
int     *dac_buf        =       DAC_BUF;


/* logical (bit-wise) i/o locations */
#define DR_CSR          0767770
#define DR_OUT          DR_CSR + 2
#define DR_IN           DR_CSR + 4

/*
 define the vectors that point to the hardware locations
*/
int     *dr_csr         =       DR_CSR;
int     *dr_out         =       DR_OUT,
int     *dr_in          =       DR_IN;

/* Real time clock */
#define KV_CSR 0170420
#define KV_BPR KV_CSR+2
#define KV_OVF 0200
#define KV_ERR 010000
#define KV_RATE 001 /*1 MHz*/
#define KV_MODE 01
#define KV_GO 01

/*
 define the vectors that point to the hardware locations
*/
static unsigned *kv_csr = KV_CSR;
static unsigned *kv_bpr = KV_BPR,


/* the pseudo storage classes */
#define FAST register

#define OFF 0
#define ON 1


/* ************************************************************************ */
```

```
/*****************************************************************************

        EXTEND.C        4-NOV-85

                        From Ted Milner used to be called--
        tstexb.c:       test exbuff: extended memory buffer routines.
        Provides:
        XB_INIT():
                initialize the extended memory buffer. Call this routine
                 first.
        int XB_WRITE( data_pointer, number_of_bytes,
                                        extended_memory_buffer_pointer ):
                int *data_pointer;
                int number_of_bytes;
                long int extended_memory_buffer_pointer;
                Write number_of_bytes bytes from data_pointer to the
                extended memory buffer.
                Returns the number of bytes actually written.
        int XB_READ( data_pointer, number_of_bytes,
                                        extended_memory_buffer_pointer ):
                int *data_pointer;
                int number_of_bytes;
                long int extended_memory_buffer_pointer;
                Read number_of_bytes bytes from the extended memory buffer
                to data_pointer.
                Returns the number of bytes actually read.
        NOTES:  .) An extended memory buffer pointer is a long byte pointer
                The beginning of the buffer is 0L. The end of the buffer is
                xb_max_pointer;

*****************************************************************************/

#include "std.h"

/*****************************************************************************/

#define PAGE_SHIFT      6       /* How much to shift an extended memory
                buffer address to turn it into a page address. */
#define PAGE_MASK       (~077)  /* Make page addresses point to page
                                        boundaries. */
#define START_PAGE      01600   /* Starting page address. */


/*****************************************************************************/

GLOBAL long int xb_max_pointer = 196608L;
        /* Maximum extended memory buffer pointer:
        A pointer >= xb_max_pointer is illegal. */

/*****************************************************************************/

GLOBAL xb_init()
        {
        xbfini();       /* Just call the MACRO initialize routine. */
        putfmt( "initialized.\n" );
        }

/*****************************************************************************/

GLOBAL xb_write( p_data, n_bytes, p_xbf )
        int *p_data;
```

```
        int n_bytes;
        long int p_xbf;
        {
        int page_address;
        int w_pointer[ 2 ];     /* 2 word extended buffer pointer. */
        int return_value;

        if ( ( ( n_bytes / 2 ) * 2 ) != n_bytes )
                {
                errfmt( "Odd n_bytes in XB_WRITE: %l\n", n_bytes );
                exit( NO );
                }

        page_address = (p_xbf >> PAGE_SHIFT) & PAGE_MASK;
        w_pointer[ 0 ] = page_address + START_PAGE;
        w_pointer[ 1 ] = p_xbf - (((long)page_address) << PAGE_SHIFT);
        return_value = xbfwrt( p_data, n_bytes, w_pointer );
        return( return_value );
        }

/*****************************************************************************/

GLOBAL xb_read( p_data, n_bytes, p_xbf )
        int *p_data;
        int n_bytes;
        long int p_xbf;
        {
        int page_address;
        int r_pointer[ 2 ];     /* 2 word extended buffer pointer. */
        int return_value;

        if ( ( ( n_bytes / 2 ) * 2 ) != n_bytes )
                {
                errfmt( "Odd n_bytes in XB_READ: %l\n", n_bytes );
                exit( NO );
                }

        page_address = (p_xbf >> PAGE_SHIFT) & PAGE_MASK;
        r_pointer[ 0 ] = page_address + START_PAGE;
        r_pointer[ 1 ] = p_xbf - (((long)page_address) << PAGE_SHIFT);
        return_value = xbfread( p_data, n_bytes, r_pointer );
        return( return_value );
        }

/*****************************************************************************/

GLOBAL long int xb_rfile( file_name )
        TEXT *file_name;
        {
        int n_bytes;
        FILE inf;
        TEXT buf[ BUFSIZE ];
        long int total_bytes = 0L;

        inf = open( file_name, READ, 1 );
        while ( 0 < ( n_bytes = read( inf, buf, BUFSIZE ) ) )
                {
                /* Desperately need error checking here. */
                xb_write( buf, n_bytes, total_bytes );
                total_bytes += n_bytes;
                }

        close( inf );
        return( total_bytes );
        }

/*****************************************************************************/
```

```
/* ********************************************************************** */
/*      FORCE2.C
This program is intended to generate a force of desired direction and
magnitude at the handle.  This force will be ramped up to the final
value.  It will also store the transducer output and current monitor.

This version will take 100 samples of signal from the force transducer
and use that as the offset to subtract from the actual readings.

        Ian C. Faye'    5-Jan-86
*/
/* ********************************************************************** */

#include <all.h>
#define BELL 07
#define PI 3.1415926
#define RAD (PI/180.0)

float   len_1 = .3654, len_2 = .3420,
float   sines[ 512 ] = 0.0, force[ 256 ] = 0.0;
int     i_force[ 256 ] = 0;
int     save[ 768 ] = 0;          /* save it in integer form */
int     c_size = 1536;            /* 2 bytes per integer */
int     x_trans[100] = 0, y_trans[100] = 0;
int     fx_off = 0, fy_off = 0;

main()
{
        char    dummy, dum, dumy, ans;
        char    *file_name = "dl1:cossin.dat",
        float   fmag, fair, f_x, f_y, trise, tfin, dt;
        float   off_a, off_b;
        float   xp_end, yp_end;
        int     base, count, tr_cnt,
        int     tx_total, ty_total, i, val;

        csr_set();      /* set to turn on the relays */
        dac_put( 0, 0); /* zero the DAC to start */

/* set up the extended memory */
        xb_init();
        xb_rfile(file_name);

/* set the real time clock */
        base  = 4,
        count = 50;
        ck_set(base,count);

/* the time conditions */
        dt    = .0050;  /* dependent on base and count above */
        trise = 64*dt;
        tfin  = 128*dt;

/* wait for the user to set the relays */
        printf(" enter an [s] to start: \n");
        scanf(" %c",&dumy),
        off_com(&off_a,&off_b); /* find the encoder offsets */

                ans = 'y';
        while( ans != 'e')
        {
```

```
        if(dumv == 's')
                {

        /* sample the transducer to get offsets */
                tr_cnt = 0;
                ck_clear();
                /* Real Time Loop for the force output */
                        ck_go();
                        while(ck_wait())
                        {
                                adc_in(0,&val);
                        x_trans[ tr_cnt ] = val;

                                adc_in(1,&val);
                        y_trans[ tr_cnt ] = val;

                        tr_cnt = tr_cnt + 1;
                        if( tr_cnt>99 ) break;
                        }
                        ck_stop();

/*              for( i = 0; i<100; i++ )
printf("x_trans= %d,  y_trans= %d \n", x_trans[i]-2048, y_trans[i]-2048);
*/
                        /* find the average of transducer signals */
                        tx_total = ty_total = 0;
                        for( i = 0; i<100; i++ )
                        {
                        tx_total = (x_trans[ i ]-2048) + tx_total;
                        ty_total = (y_trans[ i ]-2048) + ty_total;
                        }

                        fx_off = tx_total/100;
                        fy_off = ty_total/100;

                printf("Transducer offsets computed; \n");
                printf("for x ==> %d,  y ==> %d \n", fx_off, fy_off);
                printf("  \n");

        end_find(&xp_end, &yp_end); /* find the endpoint position */
        printf(" in meters x = %7.4f,  y = %7.4f\n", xp_end, yp_end);

                printf(" enter the magnitude of the force:\n");
                scanf(" %f", &fmag);

                printf(" enter the direction of the force in degrees:\n");
                scanf(" %f", &fdir);

                f_x = fmag*sin((fdir+90.)*RAD);
                f_y = fmag*cos((fdir+90.)*RAD);

                printf(" FORCES: fx = %6.3f,  fy = %6.3f\n", f_x, f_y);
                printf(" Delay ----> type [g] to go\n");
                scanf(" %c",&dum);
                scanf(' %c",&dummy);

                if(dummy == 'g')
                        push_me(f_x, f_y, trise, tfin, dt);

                printf(" Stop ----> type [e] to exit\n");
                scanf(" %c",&ans);
```

```
                }
        }
        ans = 'e';
                printf("%c",BELL);
        puts(" Bye");
}

/* ********************************************************************** */
/* routines */

push_me(x_for, v_for, trise, tfin, dt)
float trise, tfin, dt, x_for, y_for;
{
        char    dum;
        int tor1, tor2, i, jj, j, inax, jnax, snax, knax, fnax;
        float cos1, cos2, sin1, sin2, t, x_step, y_step, det,
        int tmp_x, tmp_y;
        float x_force, v_force, x_mul, y_mul, for_x, for_v;
        int xi_force, yi_force, val;
        int fx_total, fx_avg, c1_total, c1_avg;
        int fy_total, fy_avg, c2_total, c2_avg,

        float   f_tor1, f_tor2;

        scanf(" %c",&dum),
                printf("%c",BELL);
        fileit();
                scanf(" %c",&dum);
                printf("%c",BELL),

        snax = 0;
        inax = 0;
        t = 0.0;
        x_force = 0.0;
        y_force = 0.0;

        if(trise != 0.0 )
                {
                x_step = (x_for)*(dt/trise);
                y_step = (y_for)*(dt/trise);
                }
        else
                {
                x_step = x_for;
                y_step = y_for;
                }

        for( j = 0; j < 128; j++)
        {
        i_force[ j*2 ] = (1024*x_force);
        force[ j*2 ] = x_force;

        if( x_for >= 0.0 )
                {
                if( (x_force += x_step) >= x_for ) x_force = x_for;
                }
        else
                {
                if( (x_force += x_step) <= x_for ) x_force = x_for;
                }
```

```
        i_force[ (j*2)+1 ] = (1024*y_force);
        force[ (j*2)+1 ] = y_force;

        if( v_for >= 0.0 )
                {
                if( (v_force += v_step) >= y_for ) v_force = v_for;
                }
        else
                {
                if( (y_force += v_step) <= y_for ) v_force = v_for;
                }
        }
/*
        indx = 0;
        for( j = 0; j < 128; j++)
        {
        printf(" fx = %7.4f, fy = %7.4f\n", force[ indx++], force[ indx++]);
        }

        indx = 0;
        for( j = 0; j < 128; j++)
        {
        printf(" fx = %d, fy = %d\n", i_force[ indx++], i_force[ indx++]);
        }
*/
        /* 184 D/A units per Newton-meter; page 58 (31) of book 3 */
        x_mul = 184.*len_1;
        y_mul = 184.*len_2;

/* Real Time Loop for the force output */
        indx = jndx =  knax = fndx = 0;
        ck_clear();
        ck_go();
        while(ck_wait())
/*      for( jj = 0; jj< 5000; jj++)    */
        {
                cos_find(&cos1,&cos2,&sin1,&sin2);

                x_force = force[ fndx++ ];
                y_force = force[ fndx++ ];

                tor1 = (x_mul*(cos1*y_force - sin1*x_force));
                tor2 = (y_mul*(cos2*y_force - sin2*x_force));

                dac_put( tor1, tor2);

                /* Read the transducer and current monitor */
                for( i = 0; i < 4; i++)
                {
                adc_in(i,&val);
                save[ indx ++ ] = val;
                }
                save[ indx ++ ] = i_force[ jndx++ ];
                save[ indx ++ ] = i_force[ jndx++ ];

                /* Store the cosines and sines */
                sines[ snax ++ ] = cos1;
                sines[ snax ++ ] = cos2;
                sines[ snax ++ ] = sin1;
                sines[ snax ++ ] = sin2;
```

```
/*              if( (t += dt) >= tfin ) break;
*/
                if( index >= 767 ) break;
        }
        ck_stop();

/*      for(i = 0; i<512; i++)
                printf(" index %d, value= %6.4f\n", i, sines[i]);
*/
        printf("  \n");
        printf(" index is %d \n", index);
        printf("%c",BELL);
        dac_put( 0, 0); /* zero the force after we are done */

        /* find the average of each channel
           this is only really useful when we expect a channel to be zero */
        fx_total = fy_total = c1_total = c2_total = 0;
        for( i = 0; i<128; i++ )
        {
        /* the Force Transducer */
        fx_total = (save[ (i*6)+0 ]-2048) + fx_total - fx_off;
        fy_total = (save[ (i*6)+1 ]-2048) + fy_total - fy_off;

        /* the Current Monitor */
        /* this computation is with Newton-meters per A/D units */
        c1_total = (save[ (i*6)+2 ]-2048) + c1_total + 12;
        c2_total = (save[ (i*6)+3 ]-2048) + c2_total + 12;
        }

        fx_avg = fx_total/128;
        fy_avg = fy_total/128;
        c1_avg = c1_total/128;
        c2_avg = c2_total/128;

printf(" The averages are fx_avg: %d, fy_avg: %d, c1_avg: %d, c2_avg: %d\n",
                        fx_avg, fy_avg, c1_avg, c2_avg );

/* transform the measured joint torques to endpoint forces */
        for( i = 0; i<128; i++ )
        {
        /* see page 26 of book 3 for scaling factors    */

        /* the Force Transducer */
        tmp_x = save[ (i*6)+0 ] + 36;
        tmp_y = save[ (i*6)+1 ] + 16;
        /* this computation is with Newtons per scaled A/D units
           scaled means that the number is scaled up by 1024    */
        save[ (i*6)+0 ] = 30*(tmp_x - 2048);
        save[ (i*6)+1 ] = 30*(tmp_y - 2048);

        /* the Current Monitor */
        /* this computation is with Newton-meters per A/D units */
        f_tor1 = 0.005458*(save[ (i*6)+2 ]-2048 + 12);
        f_tor2 = 0.005458*(save[ (i*6)+3 ]-2048 + 12);

        cos1 = sines[ (i*4) ];
        cos2 = sines[ (i*4) +1 ];
        sin1 = sines[ (i*4) +2 ];
        sin2 = sines[ (i*4) +3 ];
        det = len_1*len_2*((cos1*sin2)-(sin1*cos2));
```

```
        for_x = ( (f_tor1*len_2*cos2) - (f_tor2*len_1*cos1) )/det;
        for_y = ( (f_tor1*len_2*sin2) - (f_tor2*len_1*sin1) )/det;

        save[ (i*6)+2 ] = (1024*for_x);
        save[ (i*6)+3 ] = (1024*for_y);
        }

        for(i = 0; i<768; j++)
printf(" index %d, value to store = %d, %d, %d, %d, %d, %d \n", (i-6),
        save[i++], save[i++], save[i++], save[i++], save[i++], save[i++] ),

        istore(save, c_size);    /* store the save buffer */
        endfile();
}
/* ********************************************************************** */



/*************************************************************************

        TTYRT.h1        RT11 Terminal Handling for concurrent C

*************************************************************************/

/* JSW bits, see RT11 Advanced Programmers Guide  */
#define NOWAIT  0100            /* Don't wait for terminal bit */
#define SPECIAL 010000          /* Special I/O mode bit */
#define LOWER   040000          /* Lower case enabled bit */

#define TTYBITS (~( NOWAIT | SPECIAL | LOWER ))
/* TTY bits of JSW are zero here  (TTYBITS & JSW) gives status of terminal
        bits in JSW */

#define DEFAULT ( NOWAIT | LOWER )      /* regular mode of terminal I/O */

/* EMT calls */
#define TTYIN   0340
#define TTYOUT  0341
```

REFERENCES

[1] W. Abend, E. Bizzi and P. Morasso, "Human Arm Trajectory Formation", Brain, 1982.

[2] C.J. Abul-Haj, "The Design of an Upper-arm Prosthesis Simulator with Variable Mechanical Impedance.", SMME Thesis, Massachusetts Institute of Technology, 1981.

[3] J.E. Colgate, "The Design of a Dynamics Measuring Device.", SMME Thesis, Massachusetts Institute of Technology, 1986.

[4] S.L. Cotter, "Nonlinear Feedback Control of Manipulator Endpoint Impedance.", SMME Thesis, Massachusetts Institute of Technology, September 1982.

[5] Data Technology Incorporated, Application Note 1010, Woburn, MA.

[6] T. Flash,"Organizing Principles Underlying The Formation of Arm Trajectories", PhD Thesis, Massachusetts Institute of Technology, May 1983.

[7] G.F. Franklin and J.D. Powell, Digital Control of Dynamic Systems, Addison Wesley, 1980.

[8] N.J. Hogan, "Mechanical Impedance Control in Assistive Devices and Manipulators.", In Proceedings of the Joint Automatic Control Conference, Joint Automatic Control, 1980.

[9] N.J. Hogan, "Prostheses Should Have Adaptively Controllable Impedance", Presented at the IFAC Symposium Control Aspects of Prosthetics and Orthotics Columbus, Ohio, May 7-9, 1982.

[10] N.J. Hogan, "Impedance Control: An Approach to Manipulation: Part 1 - Theory," Journal of Dynamic Systems, Measurement, and Control, Vol 107, March 1985.

[11] N.J. Hogan, "Impedance Control: An Approach to Manipulation: Part 2 - Implementation," Journal of Dynamic Systems, Measurement, and Control, Vol 107, March 1985.

[12] N.J. Hogan, "The Mechanics of Multi-Joint Posture and Movement Control," Biological Cybernetics, 52,315-331, 1985.

[13] N. Hogan, B. McKeon and I. Faye, "Kinematically Constrained Movements", work in progress.

[14] J.M. Hollerbach and T. Flash, "Dynamic Interactions Between Limb Segments During Planar Arm Movement," Biological Cybernetics, 1982.

[15] R. Johnson, "Designing With Optical Shaft Encoders," Robotics Age, May/June 1983.

[16] D.C. Karnopp and R.C. Rosenberg, System Dynanics: A Unified Approach, Wiley, 1975.

[17] M.A. Kleidon, "Modeling And Performance Of A Pnuematic/Hydraulic Hybrid Actuator With Tunable Mechanical Impedance.",SMME Thesis, Massachusetts Institute of Technology, 1983.

[18] F.A. Mussa-ivaldi, N. Hogan, and E. Bizzi,"Neural, Mechanical, and Geometric Factors Subserving Arm Posture in Humans," The Journal of Neuroscience, Vol.5, No.10, pp. 2732-2743, October 1985

[19] K. Ogata, Modern Control Engineering, Prentice-Hall, 1970.

[20] R.P. Paul, Robot Manipulators: Mathematics, Programming and Control, M.I.T. Press, Cambridge MA, 1981.

[21] H. Paynter, Analysis and Design of Engineering Systems, M.I.T. Press, Cambridge MA, 1961.

[22] PMI Switching Servo Amplifiers Instruction Manual, July 1981

[23] R.C. Rosenberg and D.C. Karnopp, Introduction To Physical System Dynamics, McGraw-Hill, 1983.

[24] Texas Instruments, The TTL Data Book for Design Engineers, Second Edition, 1981.

[25] J. Wlassich, "Nonlinear Force Feedback Impedance Control.", SMME Thesis, Massachusetts Institute of Technology, 1986.