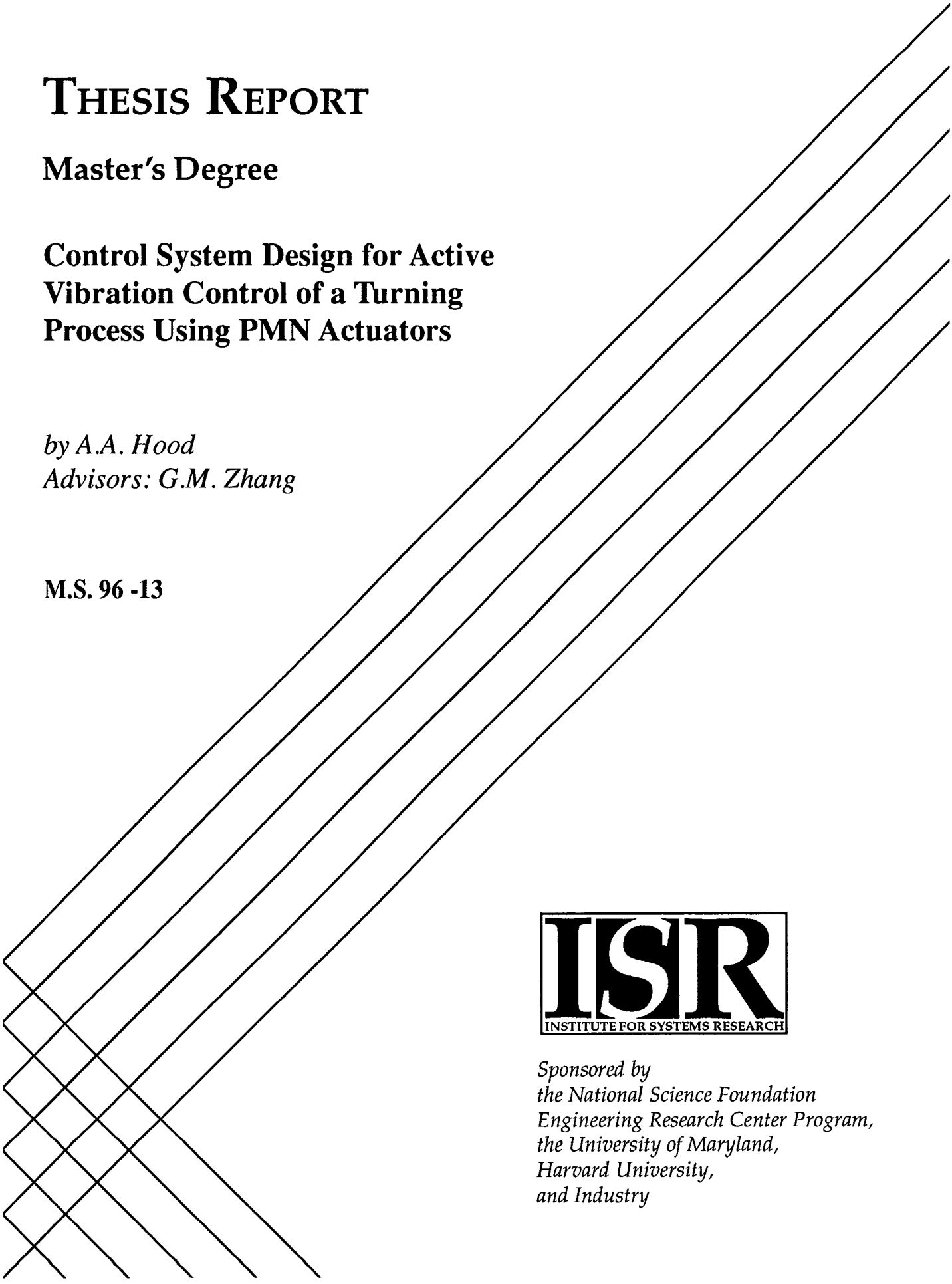
THESIS REPORT

Master's Degree

Control System Design for Active Vibration Control of a Turning Process Using PMN Actuators

by A.A. Hood
Advisors: G.M. Zhang

M.S. 96 -13



Sponsored by
the National Science Foundation
Engineering Research Center Program,
the University of Maryland,
Harvard University,
and Industry

ABSTRACT

Title of Thesis:	CONTROL SYSTEM DESIGN FOR ACTIVE VIBRATION CONTROL OF A TURNING PROCESS USING PMN ACTUATORS
Degree candidate:	Adrian A. Hood
Degree and Year:	Master of Science, 1996
Thesis directed by:	Associate Professor Guangming Zhang, Department of Mechanical Engineering and Institute for Systems Research

In recent years, intensive research has been conducted concerning the use of smart materials for active vibration control and also vibration attenuation for machine tools. This research merges the two research topics and investigates vibration control of a machining operation using actuators made of smart materials.

In this thesis, the effectiveness of using electrostrictive actuators for active vibration control during a turning process on a conventional engine lathe is investigated. The actuators are made of Lead Magnesium Niobate (PMN) and are in a multi-layered configuration. The test bed is a steel structure, called the Smart Toolpost. It is designed to be a key component of a conventional engine lathe machine tool and its purpose is to transmit compensating energy from the actuators to the tool tip during machining. The unique characteristics of PMN actuators to provide accurate displacements ensure the performance of vibration cancellation on the micron scale. The focus of this research is the design of a control system for performance optimization. This research combines both analytical and experimental approaches. In the analytical aspect, optimal, and adaptive control schemes are proposed. Mathematical models of the smart toolpost, based on first

principles, are derived and evaluated. In the experimental aspect, system identification of the toolpost's dynamics as well as the actuator's dynamics is performed. Results from computer simulations are compared with the data obtained from machining experiments, showing good agreements.

Results from this thesis show that PMN actuators are good smart material candidates for active vibration control. Optimal, and adaptive control designs are critical to achieve effective broad band vibration compensation. This thesis gives a systematic presentation of the Smart Toolpost's system modeling, control system design, and real-time microprocessor implementation. But most of all, this thesis illustrates that the use of PMN ceramic material as actuators does indeed have practical applications in precision machining.

Control System Design for Active Vibration Control of a Turning Process using PMN Actuators

by

Adrian A. Hood

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland at College Park in partial fulfillment
of the requirements for the degree of
Master of Science
1996

Advisory Committee:

Associate Professor Guangming Zhang, Chairman/ Advisor
Assistant Professor Balakumar Balachandran
Assistant Professor Gregory Walsh

DEDICATION

To my parents,

Charles and Cynthia Hood

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Guangming Zhang for his guidance and encouragement. I also wish to thank Dr. Balakumar Balachandran and Dr. Gregory Walsh for being especially helpful and agreeing to serve on my committee. I would also like to thank Dr. Darryll Pines, Dr. Miroslav Krstic, Dr. Horace Russell and David Hahn for their technical advice related to vibrations, control systems, and data acquisition.

I am grateful for the financial support provided by the National Science Foundation. I am especially thankful for the help and encouragement from my colleagues in the course of my study: Jason Harley, Huynh Luu, George Dold, Zelalem Eshete, D.T. Le, Melvin Jung, and the other members of the Advanced Design and Manufacturing Lab.

I would like to thank my wife Danielle, for her endearing encouragement, patience, and love. Finally, I would like to give a special thanks to my family and friends for their years of support: Charles and Cynthia Hood, Marguerite Hood, Ron Hood, Bernard Hood, Aaron Hood, Andrea Hood, Tim and Armani Johnson, Dawn and William Walker, Donovan Coward, Stefani Walker, Lensey and Ida Hamilton, Norman and Yolanda Pruitt, Dr. Ulysses Glee, Mike Godfrey, and Kevin Monroe.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 Importance of the Machine Tool Industry

Modern manufacturing technology is characterized by demanding requirements concerning accuracy and productivity in a highly automated manufacturing environment. Higher productivity can be achieved by higher spindle speeds, larger feed rates and greater depth of cut. Many researchers agree, special demands on machine tools used for high speed and high accuracy cutting prove the need for advanced control strategies [1]. Various factors affect the machining processes including workpiece deflection, workpiece material properties, machine tool dynamics, and tool conditions. However, these attributes are not directly measurable on-line in real time. Changes in these parameters can adversely affect the performance of the system with a fixed controller. Adaptive control, which can adjust control parameters according to time-varying changes in the system, has emerged as an attractive means of achieving performance goals [2].

The use of adaptive control on machine tools has been studied for about three decades now, but most of the applications involve CNC machines whose cutting parameters are varied (i.e. spindle speed, feedrate, and depth of cut) via the CNC controller to maintain a constant cutting force [3]. But what about the many shop floors that are still using conventional lathes? Will these lathes become obsolete because they cannot compete with the more powerful and more expensive CNC lathes? According to some researchers, the health of the manufacturing industry will be at great risk considering the fact that conventional machine tools and their variants are still dominating the industry [4]. For this reason, a smart toolpost was designed to provide the conventional engine

lathe an inexpensive means to compete against the more expensive CNC machines in the area of machining quality and productivity.

The tool vibration in the radial direction of the workpiece is a major factor affecting the finish quality of a machined surface. As a result, it is critical to control the radial force acting on the cutting tool. Lead-Magnesium-Niobate (PMN) multi-layer smart ceramic actuators were designed and fabricated. These actuators have been used for vibration and noise control [5]. In this thesis research, PMN are placed in the tool post structure to attenuate the tool vibration caused by the cutting force along the workpiece's radial direction to improve surface quality.

1.2 Smart Materials as Actuators

In recent years, applications of smart material have been used for active control of engineering structures. These materials include piezo-ceramics, shape memory alloys, magnetostrictives, electrorheological and magnetorheological fluids.

Piezo-ceramics have received the most attention on the subject of active noise and vibration control [6]. Piezoelectricity is not new. In fact, the first piezoelectric crystals were discovered in 1880 by Pierre and Jacques Cure. It wasn't until around 1954, with the introduction of Lead-Zirconate-Titonate (PZT), that wide spread use piezo electric materials began. They are especially attractive because they are able to produce accurate displacements at high frequencies. Also, compared to other smart materials, they are inexpensive, space efficient, lightweight, and easily embedded into smart structures. The PMN actuators are especially attractive because they are capable of supplying very large forces when compared to their PZT counterpart.

Some current applications of piezo-ceramic materials include surface-mounted piezo-actuators as active dampers in reducing the free vibration decay time of cantilevered beams. [7], cutting error correction mechanism for precision machinery [8], modal analysis of stepped beams [9], and deformable mirrors in an optical communication system

[8]. Some commercial applications include dot-matrix and ink-jet printers, hydrophones, ultrasonic motors, and micropositioners for robots.

1.3 Machine Tool Dynamics

During a turning process, structural elements with low dynamic stiffness are prone to excess vibrations called machine tool chatter [10]. This has adverse effects such as poor surface finish of the work piece, reduced accuracy, and reduced machine life. According to Merrit, forced and self-excited vibrations are the major sources of chatter [10]. These forced vibrations can result from the unbalance of rotating elements. The self-excited vibration presented during machining is due to the relative motion between the work piece and the cutting tool. One of the key features of the smart toolpost is that it is designed to minimize the dynamic variation of this relative motion, thereby maintaining virtually constant depth of cut. This is accomplished by having the actuators provide compensating energy in a manner such that the fluctuation of the cutting force in the radial direction is minimized. In doing so, control of the cutting process can be maintained thereby improving overall surface quality, accuracy and prolonging machine tool life.

The smart toolpost project is an ongoing effort to determine the feasibility of using smart materials to control mechanical vibrations during machining. This project has been sponsored under the Advanced Research Project Agency (ARPA) SMS Partnership Program for Synthesis and Processing of Smart Materials [5]. Industrial partners include AVX, Martin-Marietta, and Lockheed (now Lockheed-Martin).

1.4 Organization of Thesis

This thesis begins with a discussion of smart materials and structures as well as a review of current research that has been done in the area of control of machine tools. Following the review will be a discussion about piezoelectricity and electrostriction. Next,

experimental and theoretical mathematical models are developed and compared for both the smart tool post's mechanical structure and the embedded actuators. The individual models are then combined to form the complete linearized Smart Toolpost model. Then the thesis will discuss the design of an optimal control system utilizing a Kalman filter. Then these optimal poles are used in the development of an adaptive control scheme. Simulations and comparison are shown highlighting the benefits of each. Following will be a discussion on real-time micro-processor implementation. The thesis will conclude with a summary and some suggestions for future work.

2. BACKGROUND

2.1 Passive vs. Active Vibration Control

Initially, vibration control was accomplished by using passive elements in which no power source was utilized to add energy to the system. The level of vibration of the system was controlled by making a mechanical system more stiff or adding damping into the system. The amplitude of vibration of the main mass is reduced in only certain frequency ranges which are a function of relative displacements. In a paper by Batrakov and Tikhomirov [11], they used a tool holder made from powder material with increased porosity that possessed vibration-damping properties. They were able to reduce the vibration in half compared to vibrations observed during operation without using compact material. It should be noted that the frequency ranges were not mentioned in that paper. It is well known, however, that one of the major limitations of applying passive vibration control is that vibration attenuation is only effective in narrow bandwidths, therefore, active control schemes are necessary for vibration attenuation across larger bandwidths.

In papers by Olgac and Holm-Hansen [12,13], active vibration dampening of a spring-mass-damper trio was achieved with a technique they called a delayed resonator. This technique treats feedback gain and time delay as design parameters. The idea is to select a gain and time delay in real time, such that the two dominating roots lie on the

imaginary axis and the rest are in the left half of the s-plane. This pole-placement design is chosen so that the absorber of a spring-mass-damper system displaces 180 degrees out of phase with a harmonic disturbance input. The concept of 180 degrees out of phase was also utilized by Dold for vibration compensation [14]. One of the downsides to this approach is the assumption that the disturbance is harmonic.

In the paper by Regelbrugge *et al* [15], a case study was conducted on the use of smart materials for vibration control. They studied an apparatus that incorporated shape-memory alloy (SMA) springs and electrostrictive multi-layer ceramic actuators for use in spacecraft environments. The SMA springs were used to provide an adaptive-passive isolator stage and the ceramic actuators were used for positioning and canceling transmitted forces. In their research they used a Proportional-Integral-Derivative (PID) feedback controller to produce the commanded actuator force output. They concluded that electrostrictors are practical candidates for micron level positioning devices because they exhibit no electromechanical drift, and provide very high actuation authority over motion of tens of microns.

2.2 Active Vibration Control of Machine Tools

One of the earliest steps towards active control on a machine tool was taken for controlling the relative motion between a cutting tool and workpiece [16,17]. The control scheme was a function of the relative displacement between the cutting tool and the workpiece surface.

Masory and Koren have theoretically and experimentally demonstrated the feasibility of a variable-gain adaptive control system for turning [18,19]. They discussed

how process parameters such as spindle power and cutting force are regulated by varying the feedrate and spindle speed. They used a CNC and a coupled feed forward, variable gain controller to maintain a constant cutting force. The steady state results were satisfactory, but the effects on the transient performance were not discussed.

Tomizuka and Zhang [20] have shown the use of an adaptive PI controller to maintain a constant feed force. In their work, they used a Tree UP-1000 lathe with an LSI 11/23 computer. The depth of cut and material properties were treated as external disturbances. The cutting process was modeled by using least squares identification techniques on measured data conducted by open loop tests. It was found that for unknown depth of cut, its effect on the process gain could be estimated in real time and the controller gain could be adjusted so that the overall open loop gain remained constant. The results of fixed and adaptive controllers were compared showing the superiority of the adaptive one. The adaptive scheme used a self-tuning regulator and a recursive least squares identifier. They attributed the success of the controller to the eighth order model that was able to predict transient dynamics.

Daneshmend and Pak [3], used a Model Reference Adaptive Control (MRAC) scheme to control vibrations by regulating the feed force on a lathe during varying cutting conditions. In their research, they used a first order model to represent the relationship between the feed force and the feedrate. During experimentation, parameter changes were induced by producing step changes in the depth of cut. This resulted in variation in process gain and the results show that although there were large transient spikes at the point of change, the force variation eventually died down.

Rober and Shin [2], applied extended MRAC utilizing zero-phase error tracking control (ZPETC) to cutting force control for an end milling process. The paper compared the results of the extended MRAC to the standard MRAC. The method was an improvement over standard MRAC in that it was capable of meeting the tracking and regulation requirements of the design while maintaining a stable system in the presence of non-minimum phase plant zeros.

Previously mentioned, methods for vibration attenuation were done controlling the feedrate. This could also be done by controlling the depth of cut.

One attempt at using embedded actuators for controlling vibration via the tool post was researched by Rasmussen *et al* [21]. In the paper, the tool holder was made from a hollowed-out steel beam. A cylindrical piezoelectric actuator was placed inside the holder in a manner such that when placed on the lathe, it is oriented along the axial direction of the workpiece. It utilized a flexure pivot to rotate the actuator's motion at a 90^{0} angle to allow the tool tip to attain a maximum displacement of 52 μm. In their research, a 9th order model was developed using a least squares algorithm which was later simplified to a 1st order model. Results from open and closed loop tracking of a 20 Hz reference signal were displayed. A repetitive control scheme was used to achieve good tracking of periodic signals. They found that they were able to generate tool motion within sub-micron precision.

Shiraishi *et al* [22], in their paper, discussed controlling chatter during a turning process by using an optimal control law that incorporated an integral controller (LQI). They based their mathematical model on Merrits' block diagram of a machining system

[10], and used a 2nd order Pade' approximation for the regenerative term. The control variable was the depth of cut which was regulated by a 1 μm/pulse stepper motor. The results show, using a transducer attached to the tool post, they were able to detect chatter and ultimately bring it back to a stable state.

In Tewani *et al* [23], they investigated using active control of a boring bar to suppress machine tool chatter. They used piezoelectric actuators to counteract the disturbances acting on the system. A lumped mass model was created and two control strategies were investigated: acceleration and velocity feedback , and optimal control. Results show that the optimal approach was better and RMS values were decreased by a factor of 4.

As illustrated, there are many different ways to achieve vibration reduction during a turning process. This Smart Toolpost project is similar to that mentioned in [21], in which ceramic actuators are used to control the position of the tool tip during machining. The difference is that for this project, there are 3 PMN electrostrictive actuators embedded inside a mechanical structure in a manner such that it lies perpendicular to the axial direction of the workpiece. Also other active control strategies are investigated.

2.3 Smart Materials and Structures

2.3.1 Background

Although the actual materials are not all new, the field of smart materials and structures (SMS) is quite new (1987). This field pertains to the study of structures that

can be constructed of smart materials and have integrated actuators, sensors, and intelligence. Through the centuries, ages have been characterized by the dominant materials during that particular time period. For example, the Stone Age in which people were able to shape ground stones to their needs. The Bronze Age in which the material was more malleable, but had the drawback of being too soft hence making it inferior to steel which is characteristic of the current Iron Age. Now, a new material age of smart materials is beginning to evolve.

In literature, the terms 'smart', 'intelligent', 'adaptive', 'metamorphic', and 'sensoral', are used rather loosely when discussing SMS . But in general, these terms have been used to describe materials and structures that contain their own sensors, actuators, and control capabilities. To date, there is no internationally excepted definition for smart structures and materials, but there are some consistencies. The structure must be able to sense disturbances, process the information, and through commands to actuators, accomplish some beneficial reaction. Also, these smart materials must exhibit material properties that allow them to sense disturbances such as forces or act as actuators and apply beneficial forces.

One of the main driving forces for the advancement of this technology is the enormous potential for applications in the aerospace programs of NASA and US Air Force. This is because of the great demand for structures with more control capability without the cost of excessive weight and occupied space. Some other areas for applications of smart materials and structures include:

- Buildings that can dampen vibration, control shapes and attitudes, adapt to severe environments, perform orbit transfers and station keeping maneuvers
- Active vibration control, noise reduction for submarines, propeller aircraft, and large flexible structures
- Damage detection of buildings, roads, bridges
- Thermally activated valves, ducts, and switches

Following is a chart of some of the more popular smart materials and their 'smart' properties.

Material	'Smart' Property
Piezoelectric Electrostrictive	Electrically induced strains. Strain induced electric fields.
Magnetostrictives	Magnetic / field induced strains. Strain induced magnetic fields.
Electro-rheological fluids Magnetorheological fluids	Electric / magnetic field dependent dynamic yield stress
Shape Memory Alloy	Temperature dependent 'shape' & stiffness

Table 2-1: Smart materials and their properties

2.3.2 Smart Materials and Vibration Control

Several types of smart materials are being used today for active vibration control. Shape memory alloys (SMA), for example, controls vibration by "Active Modal Modification" [24]. This material has the characteristic of being able to be 'trained' into a particular shape at certain temperatures. The material can be cooled and deformed into other shapes and upon re-heating, will return to its original 'trained' state. This phenomenon is characterized by the crystal structure of the material going through an

austenite to martensite (or vice versa) phase transformation at the phase transformation temperature. This has the effect of changing the material's geometry as well as its stiffness. Typically, SMA is embedded into a composite structure and controls vibration by being actuated in a manner that changes the stiffness of the composite in different directions. During this transformation, the stiffness of SMA changes by a factor of four and the yield strength changes by a factor of 10. An application of SMA is to embed aircraft wings to change their modal characteristics. This material possesses the capability of working at very low bandwidth of about 5Hz.

Electrorheological (ER) and Magnetorheological (MR) fluids can increase their dynamic yield stress by applying electric or magnetic fields, respectively. Because of this, they propose great promise by providing a means for active vibration control of structures through active damping [25-27]. Applications include: active engine mounts, shock absorbers, clutches, brakes, actuators, and artificial joints. Unfortunately, application of ER and MR fluids have been limited by their slow (milliseconds) and non-linear behavior.

Magnetostrictive materials have some advantages over SMA , ER, and MR fluids in that they can produce larger strains in a wide bandwidth, higher energy density, and require low excitation voltage. This material is capable of being used as both a sensor and an actuator. Its strains are induced by an applied magnetic field and also, strains can produce a magnetic field. Bi and Anjanappa [28], discussed the feasibility of active vibration damping of a cantilever beam with embedded magnetostrictive mini actuators (MMA). Their simulated results look promising.

Piezoelectric materials (i.e. PZT and polyvinylidene diflouride (PVDF)), like magnetostrictives, have the ability to be used as both sensors as well as actuators. When used as a sensor, deformations of the material can produce an electric field, and when used as an actuator, an electric field can induce a deformation. In 1985, Bailey and Hubbard demonstrated the use PVDF film actuators to damper the free vibration of cantilever beams [7]. In this research, they placed PVDF polymer patches on the top and bottoms of the beam and excited them in a manner that reduced vibration.

Another piezo-ceramic that is not as popular as PZT, but possesses some superior features is lead magnesium niobate $Pb(Mg_{1/3}Nb_{2/3})O_3$ (PMN). This electrostrictive relaxor can produce maximum strains of 1000μm when the temperature is maintained in the region of the order-disorder phase transition of the material [29]. PMN based materials operating above the nominal transition temperature, typically have low electromechanical hysterisis (~5%) over moderate electric fields (~1 Mv/m peak-peak) [30]. The PMN stack is a multi-layer configuration with thin layers (125 to 250 μm) that are diffusion bonded during the manufacturing process. The change in length is proportional to the square of the applied voltage. Unlike PZT, PMN ceramics are not poled. Because of this it is an inherently more stable device without the long term creep associated with PZT (~15% compared to ~3% for PMN). PMN is better by a factor of two over PZT with an expansion rate of $1x10^{6}$/°C. A more detailed comparison between electrostrictive and piezoceramic actuators can be found in the works of Fripp and Hagood [31].

Stacking PMN disk is an effective way to produce relatively larger displacements because the total displacement of the actuator stack is the sum of the individual

contributions from each disk. This method is preferred over increasing the size of the actuator discs. This process involves placing a conductive layer in between each disk as shown in the following figure:

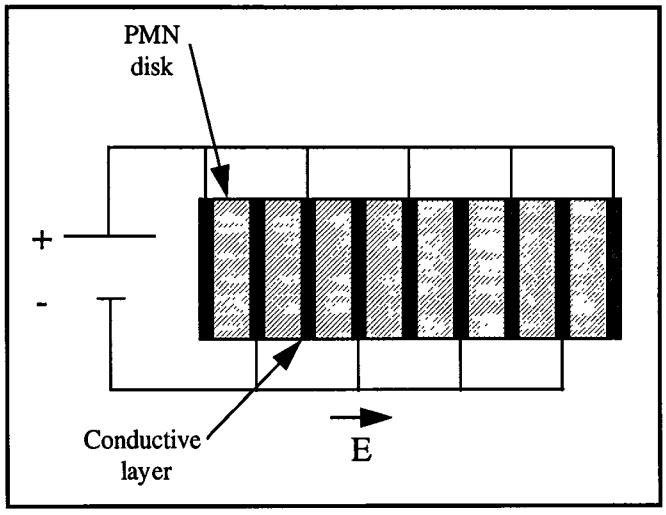


Figure 2-1: Multilayer actuator poling

Each successive layer is oppositley poled producing a strain within each individual disk.

2.3.3 Electrostriction

Electrostriction is a phenomenon in which a polarization produces a change in dimensions of a material [32]. When the material can produce an electric field upon deformation, it is referred to as piezoelectric [32]. Electrostrictives and piezoelectric belong to a class of ionic crystals known as ferroelectrics. These dielectric materials retain a net polarization when an electric field is removed. They are characterized by crystals which consist of sub-volumes called domains that have a uniform permanent re-orientable polarization. This is shown in the following figure:

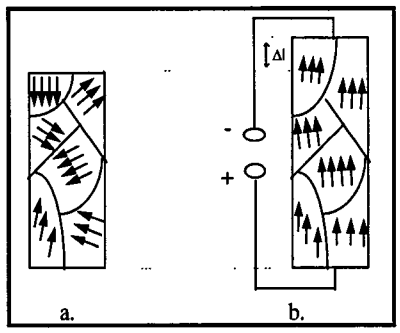


Figure 2-2: Randomly oriented dipoles a) before poling b) after poling

For many dielectric materials, when the field is removed, the dipoles become randomly arranged and the polarization of each dipole is canceled by neighboring dipoles resulting in no net polarization (Figure 2-2a.). However, in ferroelectric materials, the orientation of one dipole influences the surrounding dipoles to have an identical alignment (Figure 2-2b). This is also illustrated in the following diagram:

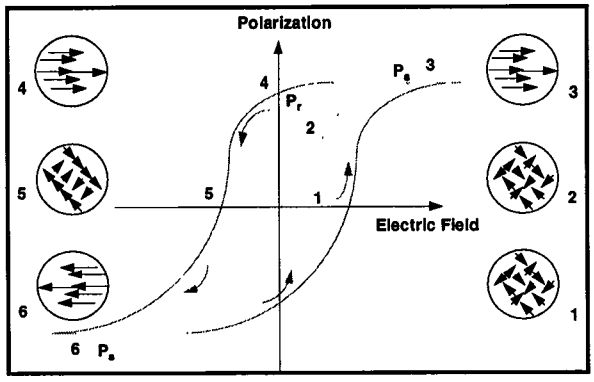


Figure 2-3: Hysterisis loop for ferroelectric materials

When the dipoles are randomly oriented, there is no net polarization (point 1). When a field is applied, the dipoles begin to line up with the field and polarization is

induced (points 1 to 3). This 1-3 curve is characterized by the hyperbolic tangent function[1]

$$|P| = P_s \tanh(k|E|) \qquad (2\text{-}1)$$

Eventually the field aligns all of the dipoles and the maximum polarization is reached (Point 3). This is referred to as the saturation polarization P_s, (also referred to as spontaneous polarization in some texts). When the field is removed, a remnant polarization, P_r, remains (point 4). This remnant polarization is due to coupling between dipoles. When the field is reversed, the dipoles are also reversed. This has the effect of 're-randomizing' the dipoles and producing a zero net polarization again (Point 5). Eventually, if the reversed field is increased, the saturation occurs again (Point 6). As the field continues to fluctuate, a loop is formed. This loop is referred to as the hysterisis loop. It shows how the polarization of the ferroelectric varies with the electric field. The area contained within the hysterisis loop is related to the energy required to cause polarization to switch from one direction to the other. This low hysterisis at small fields is one of the attractive features of PMN. This is illustrated in (Figure 2-4, [8]).

[1] **E** is the applied electric field, **P** is the polarization, ***k*** is a new material constant, and $\mathbf{P_s}$ is the spontaneous polarization. These constants are determined from experiments and a line fitting routine. The constants ***k*** and $\mathbf{P_s}$ are temperature dependent.

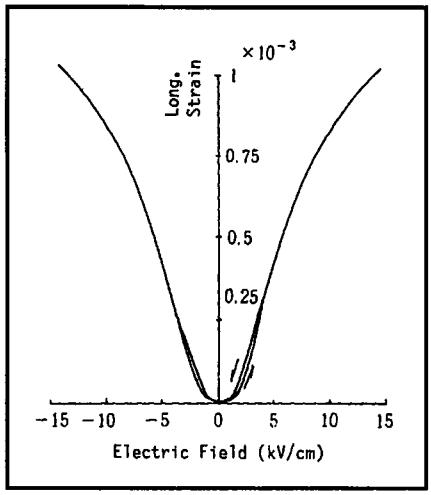


Figure 2-4: Electric field induced strains for (0.9 PMN-.1 PT) under stress free conditions.

In nature, ferroelectrics are found with zero net polarization. Piezoelectricity is induced in a ferroelectric ceramic by raising the temperature of the material above the curie temperature and applying a high electric field. This is referred to as poling. At this temperature, the ferroelectric undergoes a transition from a pyroelectric (polar) to a paraelectric (non-polar) state and the polarization disappears [8]. Applying the high electric field has the effect of realigning the polar axis of the domains to produce a net bulk polarization in the crystal. This poled piezoelectric will now deform when subjected to an electric field and polarize when mechanically stressed. This allows this material to be used both as an actuator and a sensor. The piezoelectric materials can described by the following equivalent electric circuits [33].

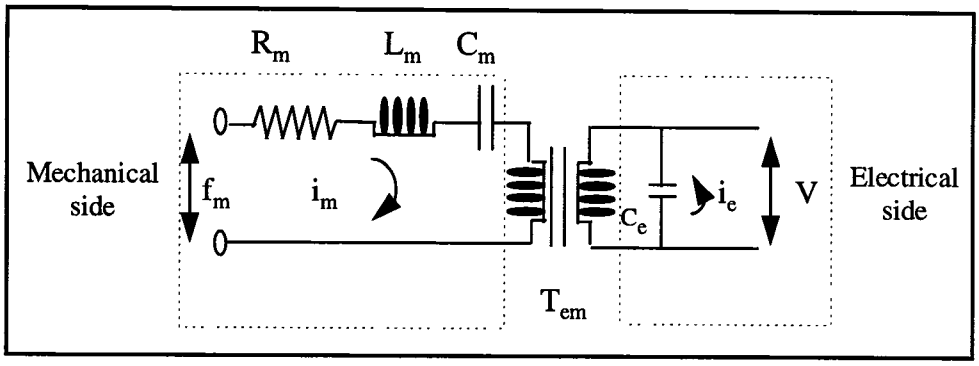


Figure 2-5: Equivalent electric circuit of a piezo-actuator

Where R_m, L_m, C_m, f_m, and i_m , represent the mechanical damping, mass, stiffness, mechanical force and 'mechanical vibration velocity', respectively. On the right side, C_e, i_e, and V represent the piezoelectric capacitance, electric current and electric voltage. T_{em} represents the electro-mechanical interlocking behavior of piezoelectric behavior which is determined by the material's properties.

PMN belongs to a special group of ferroelectric material known as relaxors. These relaxors exhibit an electrostrictive effect. This is similar to that of piezoelectricity, but instead of the strain being linearly dependent on the electric field, it is non-linearly dependent on the square of the polarization (Figure 2-4). The anions and cations of the material can be thought of as being connected by tiny springs that produce the crystal lattice (Figure 2-6, [8]).

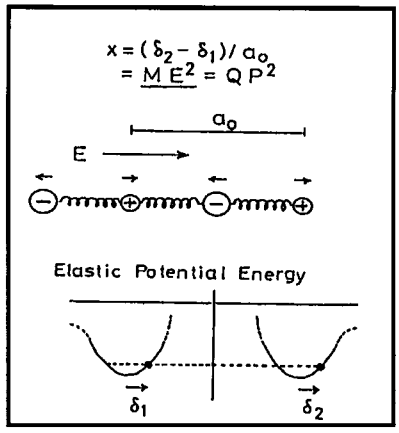


Figure 2-6 : Microscopic origin of electrostriction

Upon applying an electric field, these particles separate, producing a change in the crystal lattice. Electrostrictive materials have centrosymmetric crystals, therefore, anharmonicity of the springs is essential [8]. This is the reason for the second order effect shown in Figure 2-4. Because of this phenomenon, the same deformation occurs even when the electric field is reversed. The polarization is approximately proportional to the applied electric field, but at high electric fields, the polarization begins to saturate (spontaneous polarization). Looking again at Figure 2-4, there is a small 'linear operating range' for small electric fields. As with any other transducer, it is desirable to operate in this linear range. Therefore, for a sinusoidal input voltage with the form:

$$V(t) = V_{dc} + A\sin(\omega t) \qquad (2\text{-}2)$$

where V_{dc} is the biasing DC voltage, and A is the amplitude, linear fluctuations can be produced.

Three serially linked, embedded actuators were used for this research. The DC bias was 165 V and the maximum attainable linear displacement was frequency dependent. This is illustrated in the following plot:

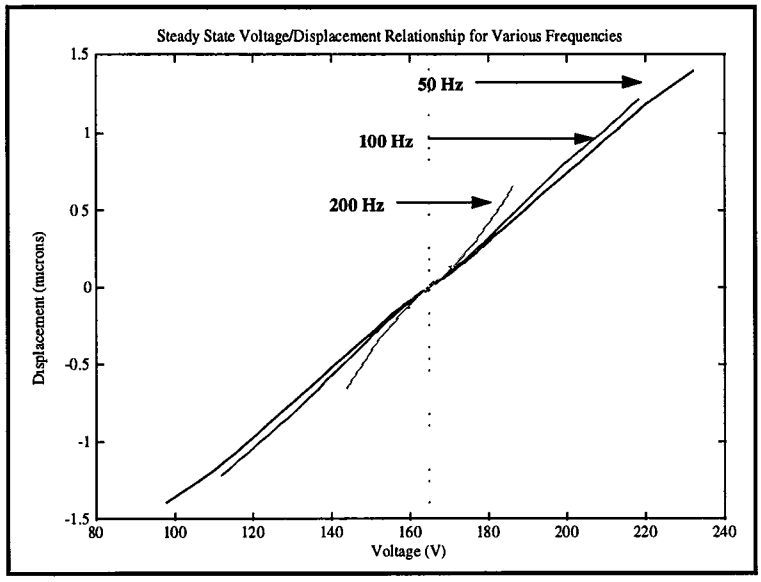


Figure 2-7: Steady state voltage/displacement relationship for 3 embedded actuators

For low frequencies, larger strains were attained for a maximum harmonic input of about 160 Vpp.

The following chart shows some typical Actuator Parameters.[2]

Material	Electrostrictive	Piezoelectric
Max. Displacement (μm)	18	16
Capacitance (μF)	6	1
Dissipation Factor(%)	4.5	3
Maximum Frequency(Hz)	100	100
Force (N)	3,000	1,200
Hysterisis (%)	2	15
Creep (Max. after 24hrs	2%	15%
Operation temperature (°C)	10-50	-20 to 80
Modules of Elasticity (Pa)	$9.6x10^{10}$	$6x10^{10}$
Tensile (Kg)	3	3
Response time (μs)	less than 100	less that 10
Thermal Coeff. of expansion	less than 1 ppm/°C	less than 2 ppm/°C

Table 2-2: Typical electrostrictive and piezoelectric material properties

[2] Information provided by AVX Corporation for a 6mm diameter, 20 mm length, cylindrical stack

3. MATHEMATICAL MODELING

3.1 Mechanical Design

The smart toolpost is a steel based mechanical structure designed to transmit vibrational energy from PMN actuators to the cutting tool. The system consists of four major parts: cutting tool holder, vibration absorber, membranes, and the toolpost housing. An exploded view of the toolpost system is shown in the following figure [4]:

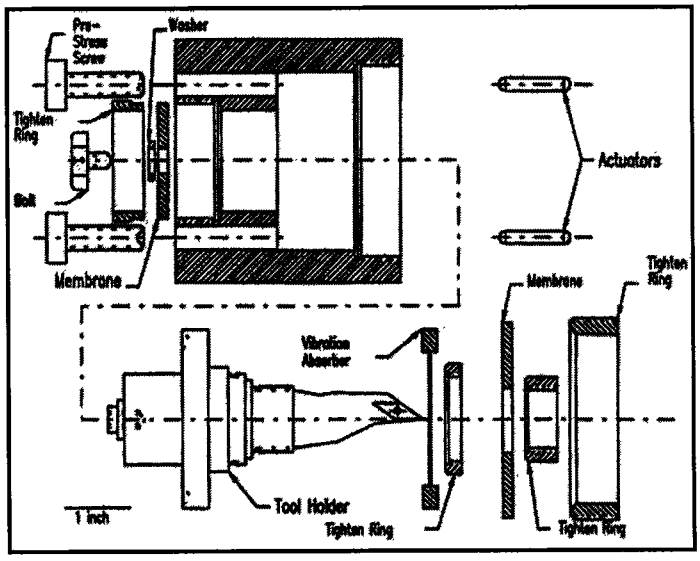


Figure 3-1: Smart Toolpost: exploded view

The tool holder is designed to firmly rest on two circular membranes. These membranes are fixed in place inside the toolpost housing via tightening rings. The actuators are placed inside special cavities of the tool housing in a manner such that they are embedded between the tool holder and tightening screws. Since the actuator

placement is off-centered, multiple actuators are needed to eliminate any induced moments. For this reason, the system is designed to handle 3 different configurations: 2, 3, or 6 actuators equally spaced along a set radius about the base of the tool holder.

To assist in dampening out high frequency vibrations, a passive vibration absorber is also placed onto the tool holder. A cross-section of the final installation is shown in the following figure [4].

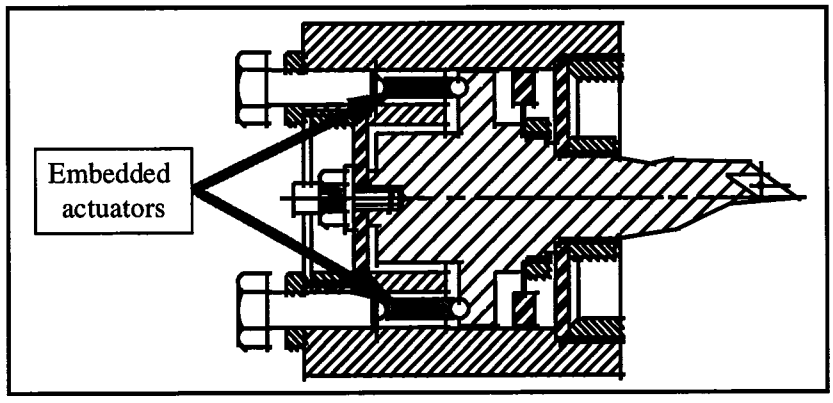


Figure 3-2 : Cross-section of the Smart Toolpost structure

The methodology of the operation of the toolpost is as follows: The system is designed to be installed onto a 15" Clausing Colechester engine lathe. During the cutting process, the cutting tool holder vibrates and its motion is detected by a linear displacement sensor located at the opposite end of the tool holder. The sensed signal is then sent through a digital control algorithm which produces the necessary compensating signal. When this voltage is applied across the actuators, they produce a compensating force against the tool holder which in turn, transmits the force to the cutting tool. The ideal condition is one in which the cutting force remains constant. This would minimize surface aberrations due to fluctuating forces.

3.2 Mathematical Model of the Toolpost's Mechanical Structure:

In this thesis research, an assumption is made that the toolpost's mechanical structure can be thought of as a single degree of freedom, spring-mass-damper system as shown in the following diagram:

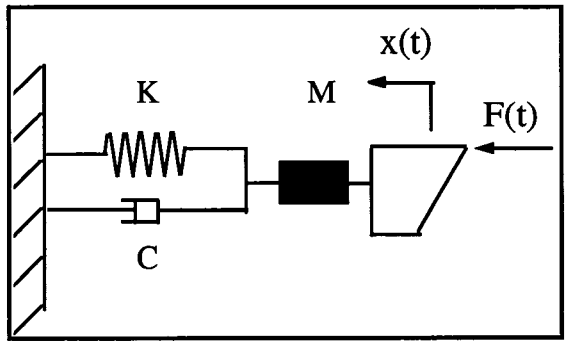


Figure 3-3: 1-D spring-mass-damper representation of the toolpost's mechanical structure

This model produces the following transfer function that maps the force excitation to the output displacement of the tool tip:

$$\frac{X(s)}{F(s)} = \frac{1}{Ms^2 + Cs + K} = \frac{1}{s^2 + 2\varsigma\omega_n s + \omega_n^2}\left(\frac{1}{M}\right) \qquad (3\text{-}1)$$

where M, C and K are the effective mass, damping, and stiffness values. It will be shown in section 4.4 how the values for the natural frequency and damping coefficient were determined.

3.3 Mathematical Model of the Embedded Actuator[3]

Now that the mechanical structure has been discussed, the following section will give the derivation of the mathematical model of an embedded actuator. A single degree of

---

[3] This derivation follows closely with the works of Liang, Fun, and Rogers. [35]

freedom, spring-mass-damper system will be used in the following derivation to represent the actuator/mechanical structure dynamics. This is illustrated in the following figure for a harmonic disturbance input.

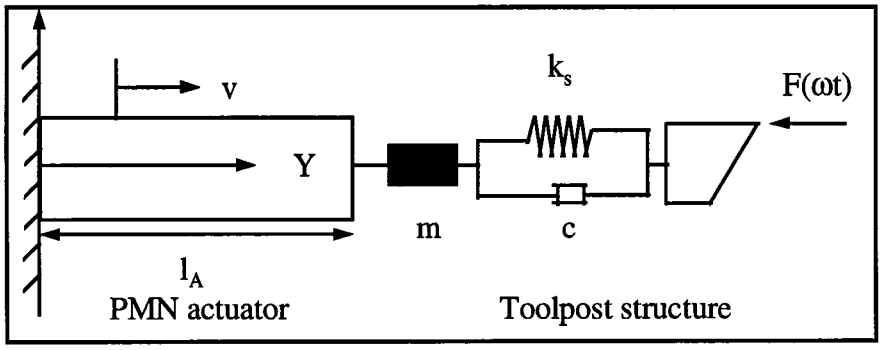


Figure 3-4 : 1-D spring-mass-damper system to model structural impedance on a PMN stack

The actuator is modeled as a bar, fixed at one end (tightening screw) and acted upon by a single frequency cutting force on the other end (via tool holder). The force-displacement relations for the PMN actuator can be expressed as:

$$F=-K_A(y-y_{in}) \tag{3-2}$$

where y is the displacement and y_{in} is the free induced displacement equal to:

$$y_{in} = Q_{3333} P_3 P_3 l_A \tag{3-3}$$

where Q is the electrostrictive coefficient, P is the polarity in the 3 direction[4], and K_A is the static stiffness of the PMN which is given by

$$K_A = EA_A / l_A \tag{3-4}$$

where E is the modulus of elasticity, A_A , is the cross-sectional area of the actuator , and l_A is the length.

[4] The 3 direction is perpendicular to the face of the actuator disks

In this derivation, the following relation, which is based on the concept of mechanical impedance, is being utilized:

$$F = -Z\dot{y} \tag{3-5}$$

where Z represents the mechanical impedance of the structure, which is represented by the spring-mass-damper system and is given by the following formula:

$$Z = m\left(\frac{\omega^2 - \omega_n{}^2}{\omega}\right)i + c \tag{3-6}$$

Where ω_n, is the natural frequency of the structure, ω is the forcing frequency, m is the effective mass, and c is the damping coefficient. Therefore, the force acting on the actuator is given by the following formula:

$$F = -Z\dot{y} = -Z(i\omega)y = -\left(c + m\left(\frac{\omega^2 - \omega_n{}^2}{\omega}\right)i\right)(i\omega)y \tag{3-7}$$

which reduces to:

$$F = -\left(c(i\omega) - m(\omega^2 - \omega_n{}^2)\right)y = -K_D y \tag{3-8}$$

where K_D is termed as the dynamic stiffness.

From equations *(A-30)* and *(A-31)* in appendix A, modified to reflect one dimension, where the axial direction is represented by the subscripts *33,* we have:

$$E_3 = -2Q_{3333}\sigma_{33}P_3 + \frac{1}{k}\arctan h(\frac{|\mathbf{P}|}{P_s})\frac{P_3}{|\mathbf{P}|} \tag{3-9}$$

and

$$\varepsilon_{33} = s^P_{3333}\sigma_{33} + Q_{3333}P_3P_3 \tag{3-10}$$

The governing equation that describes the axial vibration of a bar is:

$$\frac{EA_A}{\rho}\frac{\partial^2 V(y)}{\partial y^2} = \frac{\partial^2 V(t)}{\partial t^2} \tag{3-11}$$

where ρ, is the mass density of the PMN.

Using separation of variables, $V(y,t)=v(y)F(t)$, the solutions to this differential equation have the following form:

$$\begin{aligned} v(y) &= C_1 \sin(\hat{k}y) + C_2 \cos(\hat{k}y) \\ F(t) &= C_3 \sin(\omega t) + C_4 \cos(\omega t) \end{aligned} \tag{3-12}$$

where C_1 and C_2 are constants determined by boundary conditions, C_3 and C_4 are determined from initial conditions and

$$\hat{k}^2 = \frac{\omega^2 \rho}{EA_A} \tag{3-13}$$

where ω is the forcing frequency.

Considering only the spacial part, the two boundary conditions are as follows:

1.) Displacement at $y=0$ is zero, i.e. $v(0)=0$

2.) The force at the end of the actuator is equal to the force exerted by the structure.

$$F_{str} = A_A \sigma_{33} = A_A T_{str}\Big|_{y=l_a} = -Z\dot{v}\Big|_{y=l_a} = -Zv\Big|_{y=l_a}(i\omega) \tag{3-14}$$

Inserting the 1st boundary condition into equation (3-12), we find:

$$v(0) = C_2 = 0 \tag{3-15}$$

This relationship reduces to $C_2 = 0$

Using equations (3-7) and (3-10) and inserting the second boundary condition into equation (3-12), we find the induced strain to be:

$$\varepsilon_{33} = \frac{\partial v}{\partial y}\Big|_{y=l_a} = s^{p}_{3333}\frac{Zv\big|_{y=l_a}(i\omega)}{A_A} + Q_{3333}P_3P_3 \tag{3-16}$$

where

$$\frac{\partial v}{\partial y}\Big|_{y=l_A} = C_1\hat{k}\cos(\hat{k}l_A) \tag{3-17}$$

and

$$v\big|_{y=l_A} = C_1\sin(\hat{k}l_a) \tag{3-18}$$

substituting equations (3-17) , and (3-18) , into equation (3-16) yields:

$$\varepsilon_{33} = C_1\hat{k}\cos(\hat{k}l_A) = s^{p}_{3333}\frac{ZC_1\sin(\hat{k}l_A)(i\omega)}{A_A} + Q_{3333}P_3P_3 \tag{3-19}$$

Solving for C_1 yields:

$$C_1 = \frac{Q_{3333}P_3P_3}{\left(\hat{k}\cos(\hat{k}l_A) - s^{P}_{3333}\dfrac{Z\sin(\hat{k}l_A)(i\omega)}{A_A}\right)} \tag{3-20}$$

The impedance of the PMN actuator at $y=l_a$ is as follows:

From equation (3-5):

$$Z_A = \frac{F}{\dot{y}} \tag{3-21}$$

where:

$$F = EA_A\frac{\partial v}{\partial y}\Big|y=l_a \tag{3-22}$$

and

$$\dot{y}_A = \dot{v}\big|_{y=l_A} = (i\omega)v\big|_{y=l_A} \tag{3-23}$$

Therefore, combining equations (3-22) and (3-23) and substituting them into equation (3-21) yields:

$$Z_A = \frac{F}{(i\omega)v\big|_{y=l_A}} = \frac{\frac{\partial v}{\partial y}\big|_{y=l_a} EA_A}{(i\omega)v\big|_{y=l_A}} = \frac{C_1 \hat{k}\cos(\hat{k}l_a)EA_A}{(i\omega)C_1\sin(\hat{k}l_A)} = -\frac{\hat{k}EA_A}{\omega\tan(\hat{k}l_A)}i \qquad (3\text{-}24)$$

Recalling that:

$$K_A = \frac{EA_A}{l_A} \qquad (3\text{-}25)$$

and substituting equation (3-25) into equation (3-24) yields:

$$Z_A = \frac{-K_A\hat{k}l_A}{\omega\tan(\hat{k}l_A)}i \qquad (3\text{-}26)$$

Therefore, the coefficient C_1 can be re-written as:

$$C_1 = \frac{Z_A Q_{3333} P_3 P_3}{\hat{k}\cos(\hat{k}l_A)(Z_A + Z)} \qquad (3\text{-}27)$$

Now the actuator's displacement and the exerted force at $x=l_A$ can be determined as:

Actuator's tip displacement:

$$x = v\big|_{y=l_A} = \frac{Z_A Q_{3333} P_3^2}{\hat{k}l_A(Z_A + Z)}\tan(\hat{k}l_A) \qquad (3\text{-}28)$$

Force: at $x=l_A$

$$F = \sigma_{33}A_A = (\varepsilon_{33} - Q_{3333}P_3P_3)\frac{1}{s^P_{3333}}A_A = \frac{Q_{3333}P_3^2}{s^P_{3333}}\left(\frac{Z_A}{(Z_A+Z)} - 1\right)A_A \qquad (3\text{-}29)$$

where the material constants are in Table 3-3 [34]:

Temperature (°C)	P_s (C/m^2)	k (m/MV)	Q_{3333} (m^4/C^2)	$E = \frac{1}{s_{3333}^{\ p}}$ (Gpa)	Curie Temp. (°C)
5	.259	1.16	.0133	97	18

Table 3-3: Table: Electromechanical material properties for b250077, PMN-PT-BT[5]

Equations (3-28) and (3-29), show the displacement and force relationships at the tip of the actuator, but it fails to determine the actual displacements and forces experienced at the tool tip. Also, this model was derived using a harmonic input. For a random input, the complexity of the relationships would increase. Because of these reasons, alternate means for deriving a simpler model that adapted nicely for use with modern control principles was sought. In fact, as will be shown in the next chapter, system identification was able to capture the system's dynamics taking into consideration boundary conditions and produce a linearized model in a form that was conducive to various control design algorithms.

[5] b250077, PMN-PT-BT corresponds to 7.7% $PbTiO_3$ (PT) with PMN as the base and 2.5% $BaTiO_3$ (BT) as the a dopant. (percentages are by the volume).

4. SYSTEM IDENTIFICATION

4.1 System Identification Theory

In order to design an optimal or adaptive controller, it is best to have as accurate a model of the controlled system as possible. For this toolpost project, two transfer functions had to be determined, one that mapped the excitation force to the displacement of the tool's tip and the other that mapped the applied actuator voltage to tool's tip displacement. This was done for three embedded actuators. This was accomplished by utilizing a least squares scheme using measured input /output data and a predetermined linear Box-Jenkins model [36]. This model has the following form:

$$y(z) = G(z)u(z) \tag{4-1}$$

where:

$$G(z) = \frac{B(z)}{A(z)} = \frac{b_1 z^{-nk} + b_2 z^{-nk-1} + \cdots + b_{nb} z^{-nk-nb+1}}{1 + a_1 z^{-1} + \cdots + a_{nf} z^{-nf}} \tag{4-2}$$

and $u(z)$ is the discrete time input signal and z^{-1} is the delay operator. The coefficients b_i and a_i of the numerator and denominator are the parameters to be determined. Also, the orders and delays are described by *nb, nf,* and *nk.* The objective is to fit the measured input /output relationships to the predetermined model structure. Various model orders and delays are tried and there are various criteria, which will be discussed, that govern

which model is actually chosen. Before presenting the results of the system identification, a little background is in order.

4.1.1 Least-Squares

The basic concept of least-squares method is to fit a series of measured data to a mathematical model in a manner that minimizes the mean square error of the difference between the observed and computed data.

Let the model output be:

$$\hat{y}(t) = \theta^T \varphi(t) \tag{4-3}$$

where $\hat{y}(t)$ is the estimated output, θ is the estimated parameter vector given by

$$\theta = [a_1 \quad a_2 \quad \cdots \quad a_{na} \quad b_1 \quad b_2 \quad \cdots \quad b_{nb}] \tag{4-4}$$

and $\varphi(t)$ is the I/O signal vector given by:

$$\varphi(t) = \left[-y(t-1) \quad \cdots \quad -y(t-na) \quad u(\text{t - nk}) \quad \cdots \quad u(t-nk-nb+1)\right]^T \tag{4-5}$$

The objective is to minimize the prediction error by minimizing the mean square error:

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\left[y(t) - \hat{y}(t)\right]^2 \tag{4-6}$$

Expanding this equation yields:

$$\begin{aligned} V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\left[y(t) - \hat{y}(t)\right]^2 = \frac{1}{N}\sum_{t=1}^{N}\left[y(t) - \theta^T\varphi\right]^2 = \frac{1}{N}\sum_{t=1}^{N} y^2(t) - \frac{1}{N}\sum_{t=1}^{N} 2\theta^T\varphi(t)y(t) \\ + \frac{1}{N}\sum_{t=1}^{N}\theta^T\varphi(t)\varphi^T(t)\theta = \frac{1}{N}\sum_{t=1}^{N} y^2(t) - 2\theta^T f_N + \theta^T R_N\theta \end{aligned} \tag{4-7}$$

where:

$$f_N = \frac{1}{N}\sum_{t=1}^{N}\varphi(t)y(t) \text{ and } R_N = \frac{1}{N}\sum_{t=1}^{N}\varphi(t)\varphi^T(t) \tag{4-8}$$

If R_N is invertable, then:

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} y^2(t) - f_N^T R_N^{-1} f_N + (\theta - R_N^{-1} f_N)^T R_N (\theta - R_N^{-1} f_N) \quad (4\text{-}9)$$

By inspection, it can be seen that the last term is always positive, therefore equation (4-9) can be minimized by making the last term equal to zero. This is done by choosing:

$$\theta = \hat{\theta}_N = R_N^{-1} f_N \quad (4\text{-}10)$$

4.1.2 Comparing model structures

One of the best ways to determine the best model is to compare residuals when confronted with new sets of data. This can be done by comparing prediction error variances. One problem is that the size of the model is inversely proportional to the prediction error. This is true, even though the 'correct' model may have been passed. This is due to the fact that there are unnecessary parameters being used to fit the model. To avoid an unnecessarily large order model, there are two popular methods to balance the cost of too many parameters vs. the cost of error which are:

Akaike's information criterion method:

$$\min_{d,\theta} (1 + \frac{2d}{N}) \sum_{t=1}^{N} \varepsilon^2(t) \quad (4\text{-}11)$$

and

Final prediction error method:

$$\min_{d,\theta} (\frac{1 + d/N}{N(1 - d/N)}) \sum_{t=1}^{N} \varepsilon^2(t) \quad (4\text{-}12)$$

For this research, Akaike's criterion was used.

4.2 Experimental Setup

In order to eliminate the need for a dedicated lathe, an experimental setup was used. This setup consisted of various measurement and data acquisition equipment. The following table illustrates the major equipment used.

Equipment	Model #	Manufacturer
Function generator	PM 5133	Philips
Digital Oscilloscope	PM 3384	Fluke
Power Amplifier (actuators)	7602	Krohn-Hite
Power amplifier (shaker)	2706	Bruel & Kjaer
8-pole butterworth filter	901F	Frequency Devices Inc.
Mechanical Shaker	V411	Ling Dynamic Systems
PC computer	Pentium 90	Gateway Computers
Labview Software	Ver. 3.1	National Instruments
Digital Signal Processing Board	DSP2200	National Instruments
Data Acquisition Board	AT-MIO-16	National Instruments
Force Transducer	1051V5	Dytran Instruments
Displacement Sensor	Smμ-9100	Kaman

Table 4-4: Equipment used for experiments

The following figure illustrates the experimental setup.

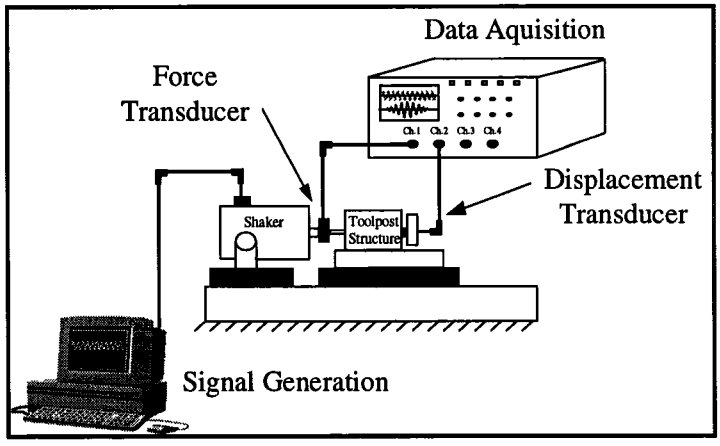


Figure 4-1: Experimental setup for system identification

A mechanical shaker was used to simulate the cutting force generated during machining. It allowed controlled inputs to excite the system. There were two ways in which the shaker was driven: a standard function generator or a software generated signal

using Matlab®. The computer generated signal utilizes the input / output capabilities of a data acquisition computer board. The function generator, although highly accurate, is limited because it is only able to generate three types of signals: sine, triangular, and square. On the other hand, the computer generated signal did not have this limitation. It is able to produce most any desired signal, (i.e. pseudo-random signals with predetermined means and variances, multi-frequency signals, along with the normal sine, triangle, and square wave). It can also send signals from a pre-measured machining process. The benefit of this setup is that it allows tests to be conducted on the toolpost without actually having a dedicated engine lathe.

4.3 Driving the Actuators[6]

Electrically, the PMN actuator disks act as capacitors with a capacitance of about 6μF. As the driving frequency of the power amplifier increases, the effective impedance of the actuator decreases. Because of this, the voltage drop across the actuator decreases, producing smaller displacements. This is especially evident as the effective impedance of the actuators approaches that of the power supply. Under these circumstances, the power amplifier is not capable of providing the actuators with the needed power. Because of this, a special driving circuit was created and a schematic is shown in the following figure:

---

[6] Information provided by Luu [37]

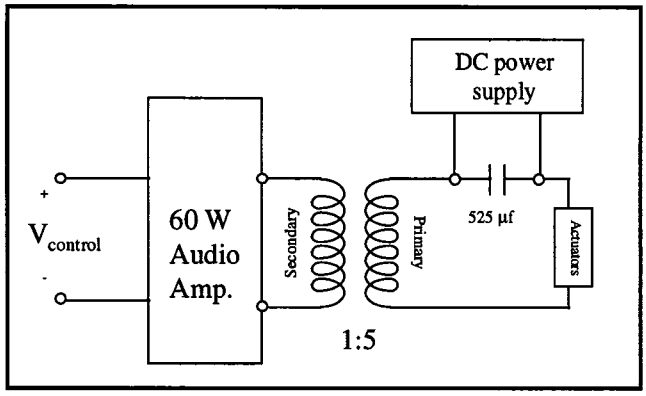


Figure 4-2: Schematic of actuator's power amplifier setup

A low impedance, 60 W audio amplifier was used to supply the driving energy to the actuators. The maximum output of this amplifier is 24 V rms. Because the desired voltage range is on the order of 100 V rmf, a 5:1 step-down audio transformer was used backwards to create a 1:5 step-up transformer. The transformer is a dual bobbin controller filament transformer, with a 110 V primary and a 22 V secondary. It is rated at 1 A at 22 V on the secondary at 60 Hz, producing an output impedance of approximately 22 Ω. The primary impedance is approximately 110 Ω. Because the transformer AC-couples the system, a 525 µF capacitor was used in conjunction with the DC supply to effectively bias the actuators. The DC power supply has its AC input shorted to the ground, but the DC offset is used across the capacitor.

4.3.1 Data Acquisition

When collecting data for system identification, three key questions need to be addressed which are:

1.) What type of signal should be used to excite the system?

2.) What is the correct sampling frequency?

3.) How many data points should be used?

1. What type of signal should be used to excite the system?

There are two conditions set on the excitation signal. These are that the signal should be persistently exciting and sufficiently rich.. This means that the excitation should be such that:

$$\alpha_1 I \geq \frac{1}{T_0}\int_t^{t+T_o} \varphi(\tau)\varphi^T(\tau)d\tau \geq \alpha_0 I, \qquad \vee t \geq 0 \tag{4-13}$$

This says that the signal vector $\varphi(t)$ should vary in such a way that the integral of the matrix $\varphi(t)^T\varphi(t)$ is uniformly positive definite over any time interval [t, t+T_o]. Sufficient richness means that the excitation signal should contain a sufficient number of frequencies to excite all of the modes of the system [38]. A signal is considered sufficiently rich of order n if it consists of at least $n/2$ distinct frequencies. This basically means that there should be at least half the number of distinct frequencies in the excitation signal as in parameters in the Θ vector equation (3-4). These two conditions will guarantee that the estimation of the parameters will converge to the actual values. The proof of this can be found in the works of Ioannou and Sun [38]. This estimation guarantee is especially important in adaptive control schemes in which constant on-line identification of the plant takes place. This will be further discussed in chapter 5.

2. What sampling frequency should be chosen?

In determining the sampling frequency, the rule of thumb is to choose one that is 5 to 20 times the bandwidth of the system [36]. This means that there should be at least 5-8 data points to represent the rise in the step response of the system. This is assisted by using an anti-aliasing filter with the cutoff frequency set at the Nyquist frequency. It should be noted however, that although it is better to sample too fast than to sample too slow, there is little information gained with the extra data points.

3. How many data points need to be collected?

Unfortunately, there is no simple rule to determine the number of data points. It depends on the signal-to-noise ratio, the complexity of the required model and the required accuracy. The following asymptotic expression can be used as a guideline [39].

$$Var\ G(s) \approx \left(\frac{n}{N}\right)\left(\frac{\Phi_v}{\Phi_u}\right) \tag{4-14}$$

Knowing the required accuracy (variance of the frequency function), the model order n, and the noise-to-signal ratio, the corresponding number of data points, N, could be calculated.

For this thesis research, system identification schemes were used to determine the dual input, single output, Smart Toolpost model. This was done by combining the toolpost's excitation force / displacement relationship with open actuators and the actuators' voltage/ compensating displacement relationship. The following sections will illustrate this task.

4.4 Identification of the Toolpost's Mechanical Structure

This section describes the development of the transfer function that mapped the cutting force to the tool tip displacement with three embedded, open, PMN actuators. The purpose of deriving the model with the actuators embedded is to take into account the extra structural dynamics contributed by the actuators.

The force signal was measured with a force transducer. The displacement was measured using a magnetic, variable impedance transducer (VIT) which is capable of measuring displacements on the order of microns. The setup is shown in the following figure.

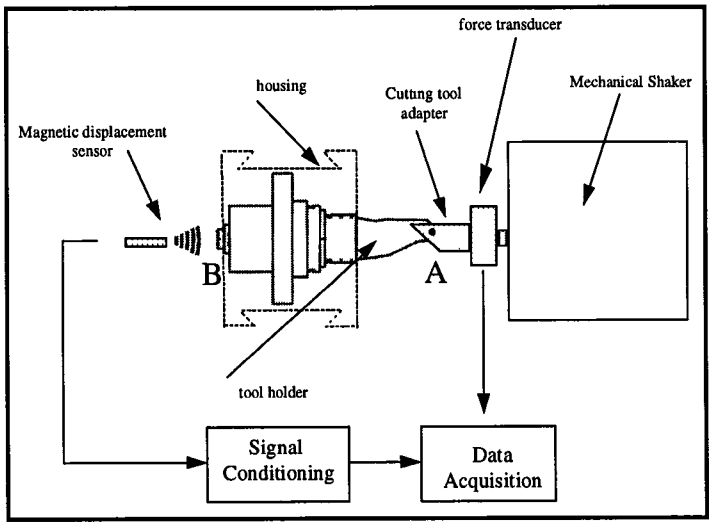
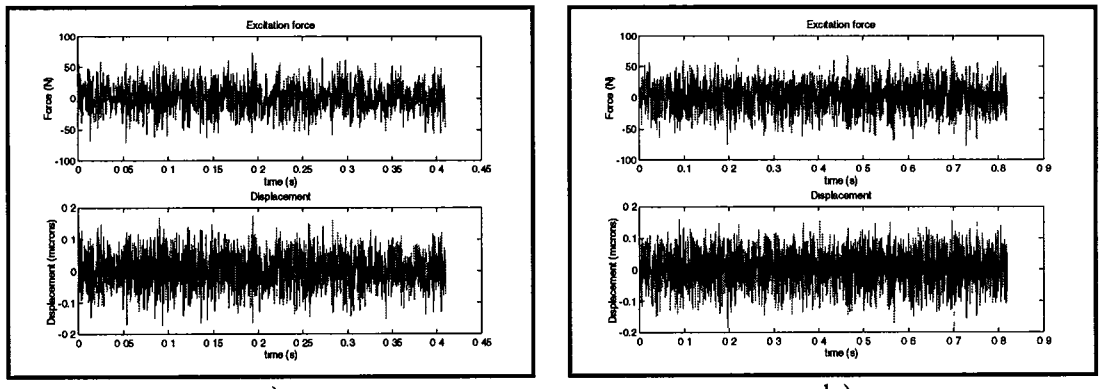


Figure 4-3: Experimental setup

Since from point A to point B in the structure is one continuous mass and assuming that there is no compression within the toolpost, the displacement at B is the same as the displacement at A. This allows the placement of the displacement sensor to be located away from the actual cutting tool.

Following are 2 plots showing the measured input / output data for the force / displacement relationship of the toolpost. The input signal was a pseudo-random excitation to the shaker and the output displacement signal was filtered through an 8-pole butterworth filter with a corner frequency of 10 kHz. The sampling frequencies were 5 and 10 kHz, and 4096 data points were collected.



a.) b.)

Figure 4-4: Input / Output data for the force/displacement relationship of the toolpost
a) sampled at 10kHz b.) sampled at 5 kHz

For a 5 volt maximum input excitation to the exciter, approximately 50 N was generated at the tool tip. Although during actual machining, the force fairly greater than that being used, we are assuming that during machining, we are still operating in the system's linear range in which case, the derived model is still valid.

The data from the first set was used to produce the following 4th order discrete time model .

$$G_{Tool}(z) = \frac{-.0001034z^3 + .002309z^2 - .00163z - .0005712}{z^4 - 1.359z^3 + 1.249z^2 - .8919z + .02563} \quad (4\text{-}15)$$

The error with respect to the Akaike criteria is 0.8657. This model represented in state variable format is:

$$x(k+1) = \begin{bmatrix} 1.359 & 1 & 0 & 0 \\ -1.249 & 0 & 1 & 0 \\ .892 & 0 & 0 & 1 \\ -.026 & 0 & 0 & 0 \end{bmatrix} x(k) + \begin{bmatrix} -.1034 \\ 2.31 \\ -1.63 \\ -.571 \end{bmatrix} 10^{-3} f(k) \qquad (4\text{-}16)$$

$$y(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} x(k)$$

Converting the model to continuous time and using a square-root balanced model reduction algorithm, the following 3rd order model was developed [40]:

$$\dot{x}(t) = \begin{bmatrix} 9258.34 & 7353.91 & -7568.69 \\ -15696.5 & -609.82 & 17637 \\ 6323.36 & -6678.64 & -10186.5 \end{bmatrix} x(t) + \begin{bmatrix} -17.4342 \\ 38.9251 \\ -21.7183 \end{bmatrix} f(t) \qquad (4\text{-}17)$$

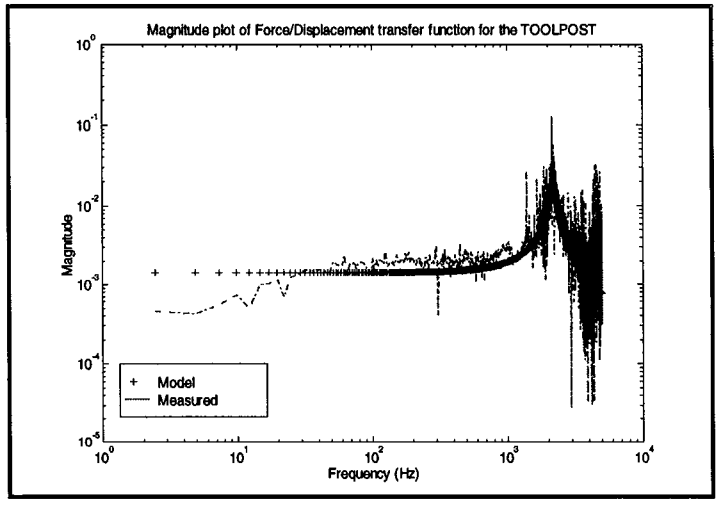
$$y(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(t)$$

For this model, the eigenvalues are located at: -688 ± 1388j (2178 Hz), and -162 (25.7 Hz). As will be shown, little information is lost by eliminating the mode associated with 25.7 Hz. This produced the following 2nd order model:

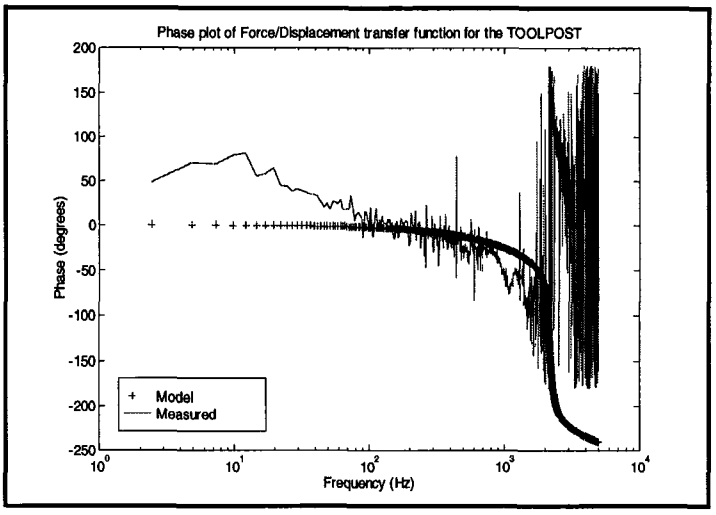
$$\dot{x}(t) = \begin{bmatrix} -215.963 & 13671 \\ -13671 & -1157.28 \end{bmatrix} x(t) + \begin{bmatrix} -2.0804 \\ -4.643 \end{bmatrix} f(t) \qquad (4\text{-}18)$$

$$y(t) = \begin{bmatrix} -2.0804 & 4.643 \end{bmatrix} x(t)$$

Comparing the frequency responses, we have:



a.)



b.)

Figure 4-5: Frequency response comparison: force / displacement relationship

As shown, the continuos model's magnitude and phase response matches quite nicely with the actual measured response. This is also apparent in a comparison of the measured time series with that of predicted output (Figure 4-6).

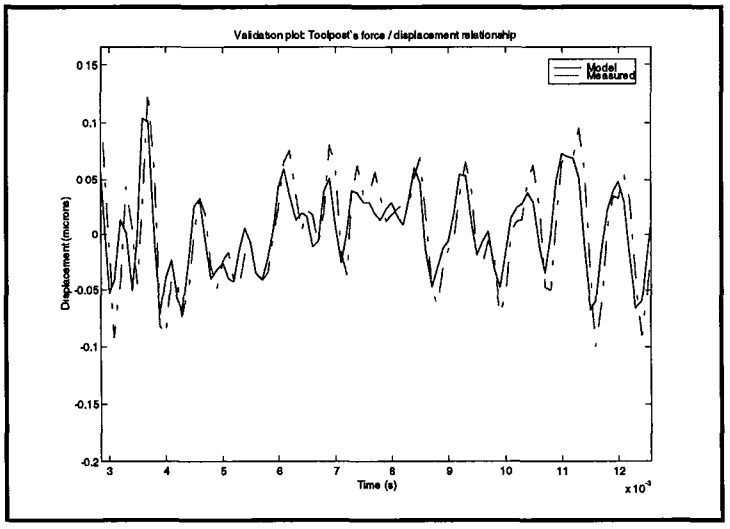


Figure 4-6: Validation plot: Smart Toolpost's force / displacement relationship

Again, this is a clear indication that the 2nd order model is a fairly accurate representation of the toolpost's dynamics. When compared to the model derived from 1st principles in

Chapter 3, for the measured mass of .963 kg, the effective stiffness is found to be 180 MN/m, and the damping coefficient, ζ, is found to be .05.

The bode plots of equation (4-18) are shown in the following figure:

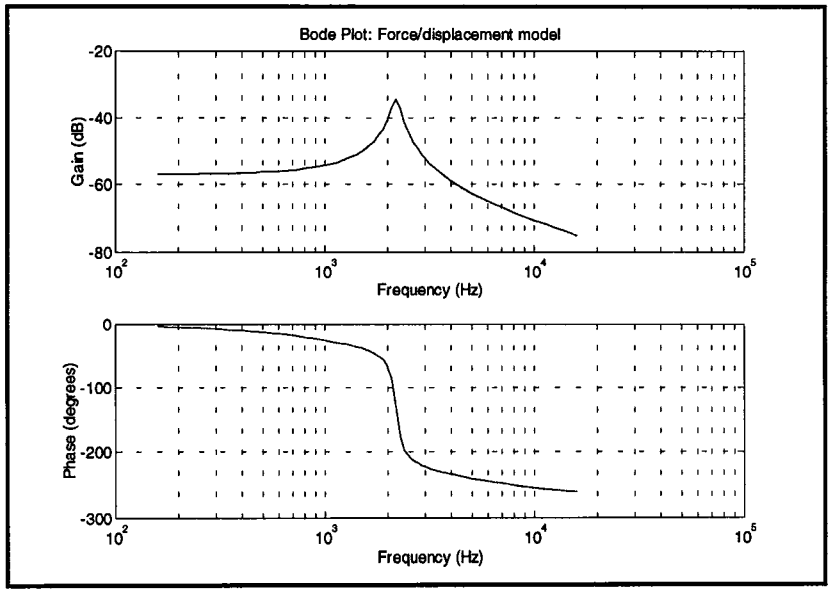
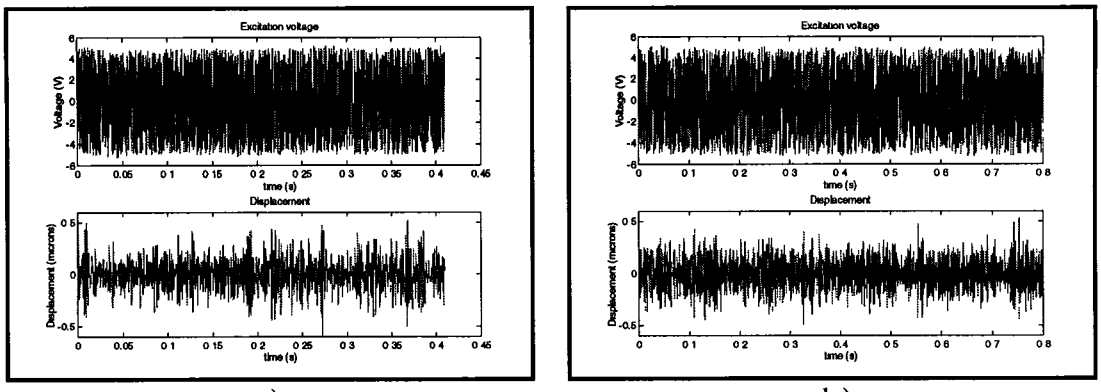


Figure 4-7: Bode Plot: Smart Toolpost's mechanical structure with 3 embedded actuators

This plot shows that the toolpost's mechanical structure has a natural frequency at 2178 Hz. which is consistent with findings from previous researchers [4,14,37,41].

4.5 Identification of the Embedded Actuators

This section describes the development of the transfer function that mapped the actuator's input voltage to the tool's tip displacement for three, serially linked, embedded PMN actuators. The excitation signals were created with the same 5V, pseudo-random signal generator as used in the previous section. The actual signal was first amplified 20dB in the power amplifier before being sent to the actuators. Following are 2 input /output plots:



a.) b.)

Figure 4-8: Input/Output data for the force/displacement relationship of the toolpost
a) sampled at 10kHz b.) sampled at 5 kHz

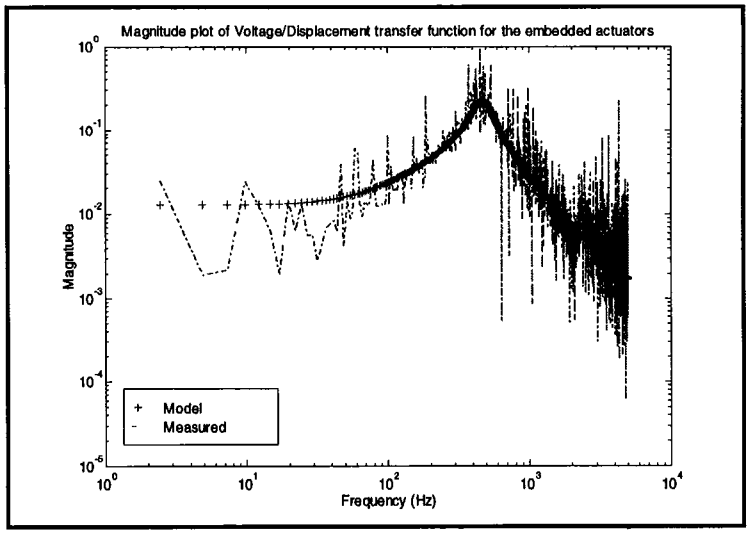
The output displacement data was filtered through a butterworth filter with a corner frequency of 10 kHz. Again, sampling frequencies of 10 and 5 kHz were used and 4096 data point were collected. Using sample set (a) and a method similar to the one used to derive the force/displacement model, the following voltage/displacement model was derived:

$$\dot{x}(t) = \begin{bmatrix} -634.9 & 2902.8 & 1196.38 \\ -2902.77 & -11.002 & -129.81 \\ -1196.38 & -129.81 & -4163.98 \end{bmatrix} x(t) + \begin{bmatrix} 12.5 \\ 1.6 \\ 10.5 \end{bmatrix} v(t) \qquad (4\text{-}19)$$

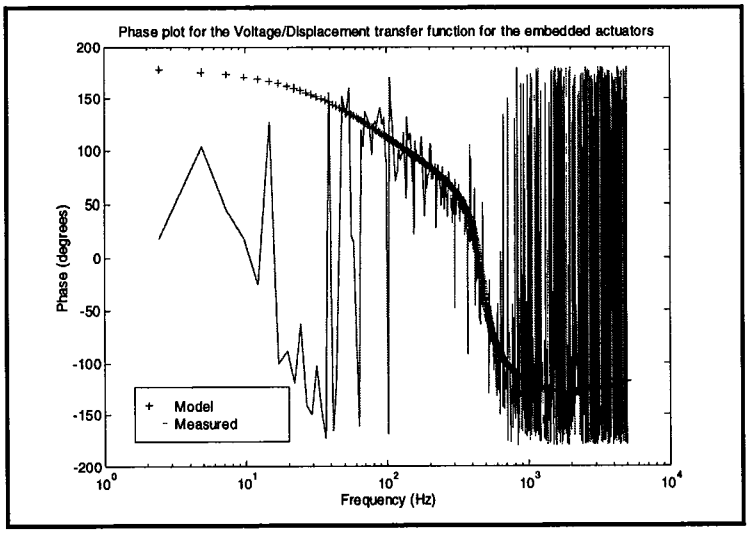
$$y(t) = \begin{bmatrix} 12.5 & -1.6 & -10.5 \end{bmatrix} x(t)$$

This model has eigenvalues at: $-476 \pm 2940j$ (474 Hz.) , and -3859 (614 Hz.)

Comparing frequency responses:



a.)



b.)

Figure 4-9: Frequency response comparison: voltage / displacement relationship

Again, the frequency response matches nicely with the measured response. Comparing the measured time series data with predictions, we find that the 3rd order model does compare well.

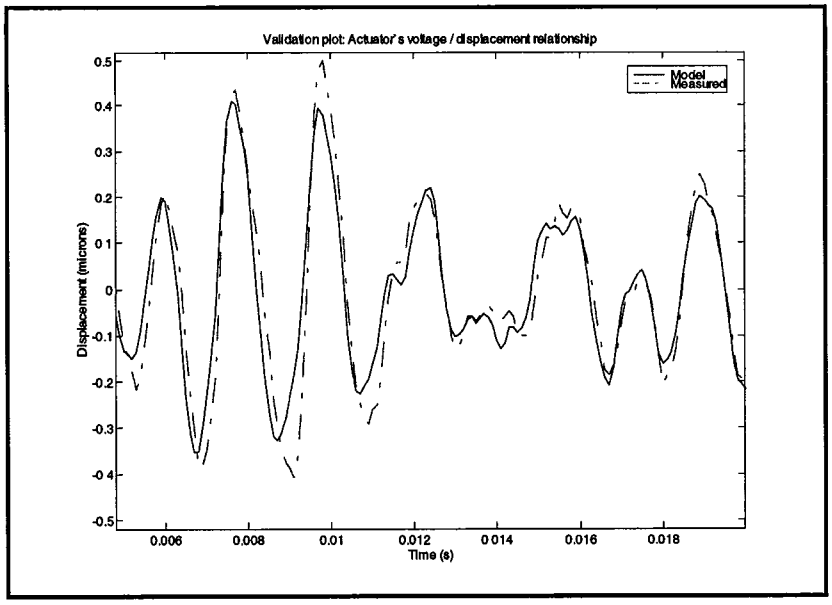


Figure 4-10: Validation plot : Smart Toolpost's voltage / displacement relationship

The bode plot is shown in the following figure:

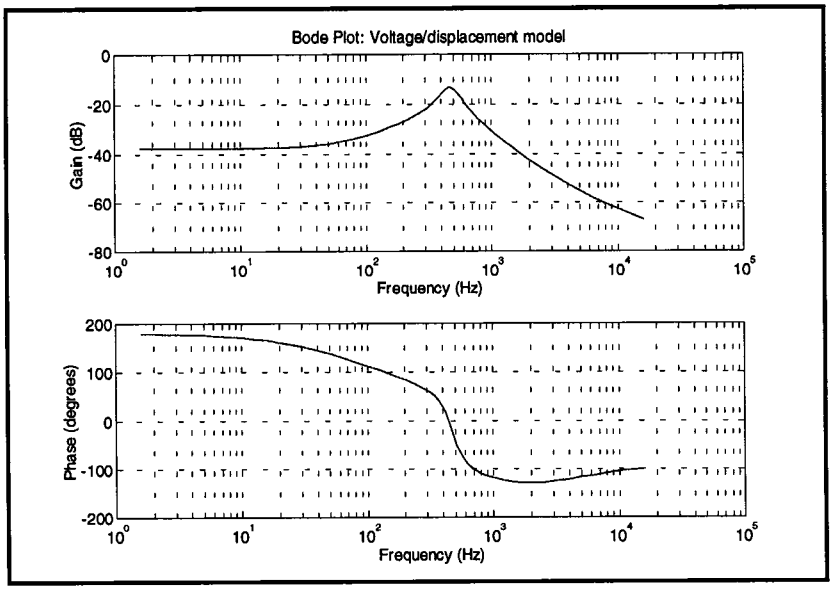


Figure 4-11: Bode plot: Toolpost's voltage/displacement relationship

4.6 Model Assembly

Now that both transfer functions have been developed, they can be combined to form the state variable realization of the smart toolpost system. The net displacement that the toolpost experiences is represented by the following formula:

$$d(s) = G_T(s)F_c(s) + G_A(s)V(s) \tag{4-20}$$

Where F_c is the cutting force and V is the control voltage.

Equation (4-20) can be represented in the following matrix form

$$d(s) = \begin{bmatrix} G_T(s) & G_A(s) \end{bmatrix} \begin{pmatrix} F_c(s) \\ V(s) \end{pmatrix} \tag{4-21}$$

Which can also be written as:

$$d(s) = \begin{bmatrix} \frac{N_T(s)}{D_T(s)} & \frac{N_A(s)}{D_A(s)} \end{bmatrix} \begin{pmatrix} F_c(s) \\ V(s) \end{pmatrix} = \hat{G}(s) \begin{pmatrix} F_c(s) \\ V(s) \end{pmatrix} \tag{4-22}$$

where N and D represent the numerator and denominator polynomials respectively. This can then be reduced to

$$\hat{G}(s) = \frac{1}{D_T(s)D_A(s)} \begin{bmatrix} N_T(s)D_A(s) & N_A(s)D_T(s) \end{bmatrix} \tag{4-23}$$

In general, this model has the following form.

(4-24)

$$\hat{G}(s) = \begin{bmatrix} e_1 & e_2 & \cdots & e_p \end{bmatrix} + \frac{1}{s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n} \begin{bmatrix} \beta_{11}s^{n-1} + \beta_{12}s^{n-2} + \cdots + \beta_{1n} & \vdots & \beta_{21}s^{n-1} + \beta_{22}s^{n-2} + \cdots + \beta_{2n} & \vdots & \cdots & \vdots & \beta_{p1}s^{n-1} + \beta_{p2}s^{n-2} + \cdots + \beta_{pn} \end{bmatrix}$$

where e is the direct transmission part of the realization (i.e. $e_i = G_i(\infty)$).

With this representation, the dynamical equation becomes: [42]

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -\alpha_n \\ 1 & 0 & \cdots & 0 & -\alpha_{n-1} \\ 0 & 1 & \ddots & 0 & -\alpha_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -\alpha_1 \end{bmatrix} x + \begin{bmatrix} \beta_{1n} & \beta_{2n} & \cdots & \beta_{pn} \\ \beta_{1(n-1)} & \beta_{2(n-1)} & \cdots & \beta_{p(n-1)} \\ \beta_{1(n-2)} & \beta_{2(n-2)} & \cdots & \beta_{p(n-2)} \\ \vdots & \vdots & & \vdots \\ \beta_{11} & \beta_{21} & \cdots & \beta_{p1} \end{bmatrix} u \qquad (4\text{-}25)$$

$$y = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix} x + \begin{bmatrix} e_1 & e_2 \cdots & e_p \end{bmatrix} u$$

For the smart toolpost, the direct transmission part does not exist because both transfer functions are strictly proper.

Converting the models (4-18) and (4-19) to their transfer functions $G_T(s)$ and $G_A(s)$, respectively, and using equation (4-25), the following state variable representation was produced:

$$\dot{x}(t) = \begin{bmatrix} -475.5 & 2939.6 & 0 & 0 & 0 \\ -2939.6 & -475.5 & 0 & 0 & 0 \\ 0 & 0 & -3858.9 & 0 & 0 \\ 0 & 0 & 0 & -686.6 & 13662.9 \\ 0 & 0 & 0 & -13662.9 & -686.619 \end{bmatrix} x(t) + \begin{bmatrix} 0 & 16.5 \\ 0 & -15.2 \\ 0 & -14.2 \\ 6.0 & 0 \\ 4.0 & 0 \end{bmatrix} \begin{Bmatrix} f_c(t) \\ v(t) \end{Bmatrix} \qquad (4\text{-}26)$$

$$y(t) = \begin{bmatrix} 5.4 & -9.0 & 12.8 & -0.42 & -3.6 \end{bmatrix} x(t)$$

This model has 5 states which correspond to 2 from the force/displacement model and 3 from the voltage/displacement model. It is presented in its modal canonical form which puts the real parts of the eigenvalues on the main diagonal and their corresponding imaginary values on the super diagonals. Remembering that although the system has 2 inputs, only one of them corresponds to the control voltage. Therefore, in order to develop a controller, the system must be controllable with respect to the 2nd column in the control matrix. Due to the nature of this model's construction, there are pole-zero cancellations with respect to the 2nd column, as illustrated in the following pole/zero plot:

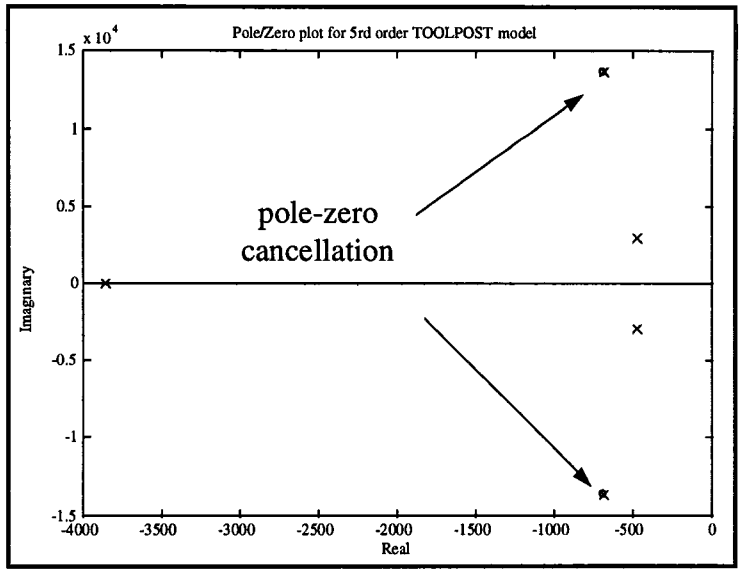


Figure 4-12: Pole/Zero plot of the complete Smart Toolpost model

Because of this, actions must be taken to make the system controllable as well as observable, but not affect the model's dynamics dramatically. Noticing that the force/displacement model has a dominant mode at 2171 Hz and the voltage/displacement model has one at 474 Hz, the 5th order model was reduced to a 4th order one. This was done by eliminating the 3rd state from equation (4-26) because of its relatively fast dynamics. The result is shown below:

$$\dot{x}(t) = \begin{bmatrix} -475.5 & 2939.6 & 0 & 0 \\ -2939.6 & -475.5 & 0 & 0 \\ 0 & 0 & -686.6 & 13662.9 \\ 0 & 0 & -13662.9 & -686.6 \end{bmatrix} x(t) + \begin{bmatrix} 0 & 16.5 \\ 0 & -15.2 \\ 6.0 & 0 \\ 4.0 & 0 \end{bmatrix} \begin{Bmatrix} f_c(t) \\ v(t) \end{Bmatrix} \qquad (4\text{-}27)$$

$$y(t) = \begin{bmatrix} 5.4 & -9.0 & -0.42 & -3.6 \end{bmatrix} x(t)$$

Taking the rank of the controllability and observability matrices, it can be verified that this model is not controllable with respect to the control input, but is observable. Because of this, the model was modified until it became both controllable and observable. This was done by moving the poles that corresponded to the force/displacement transfer function

further into the left half s-plane in a manner that preserved the natural frequency. This is illustrated in the following plot:

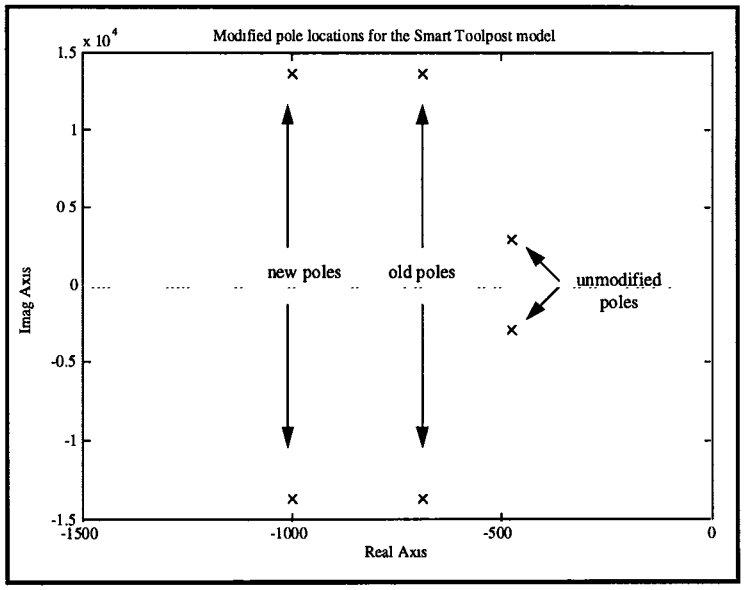


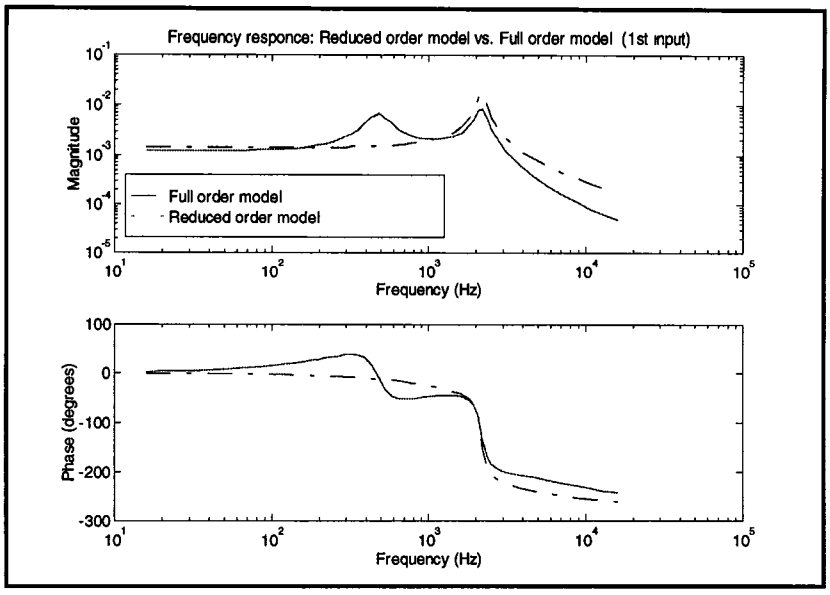
Figure 4-13: New pole locations for the modified Smart Toolpost model

The controllable model is shown below in its balanced realization:

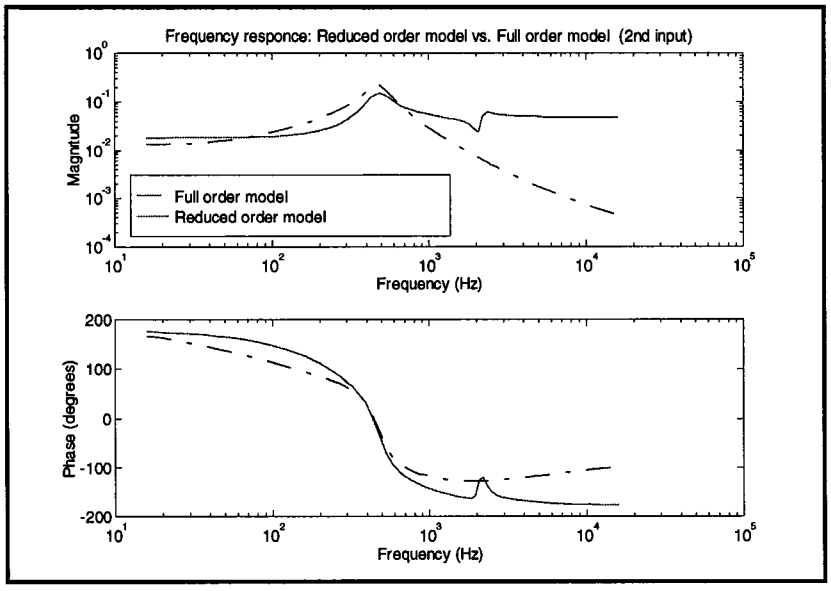
$$\dot{x}(t) = \begin{bmatrix} -993.68 & -2945.9 & 901.42 & 741.8 \\ 2945.55 & -40.93 & 312.83 & 112.92 \\ 896.26 & -312.76 & -1416.55 & -13721.4 \\ -692.6 & 92.17 & 13607.4 & -498.84 \end{bmatrix} x(t) + \begin{bmatrix} .5 & 13.82 \\ 0.06 & -2.62 \\ 0.38 & -7.68 \\ 1.96 & 3.78 \end{bmatrix} \begin{Bmatrix} f_c(t) \\ v(t) \end{Bmatrix} \quad (4\text{-}28)$$

$$y(t) = \begin{bmatrix} 13.83 & 2.62 & -7.59 & -4.25 \end{bmatrix} x(t)$$

The deviation in the frequency response for both inputs is shown in Figure 4-14. As can be seen, there are slight changes. In the frequency response that corresponds to the 1st input, (force/displacement), there is an additional peak located at around 500 Hz. For the 2nd input, (voltage/displacement), there is a moderate drop past 500 Hz, unlike the sharp drop off in the full order model.



a.)

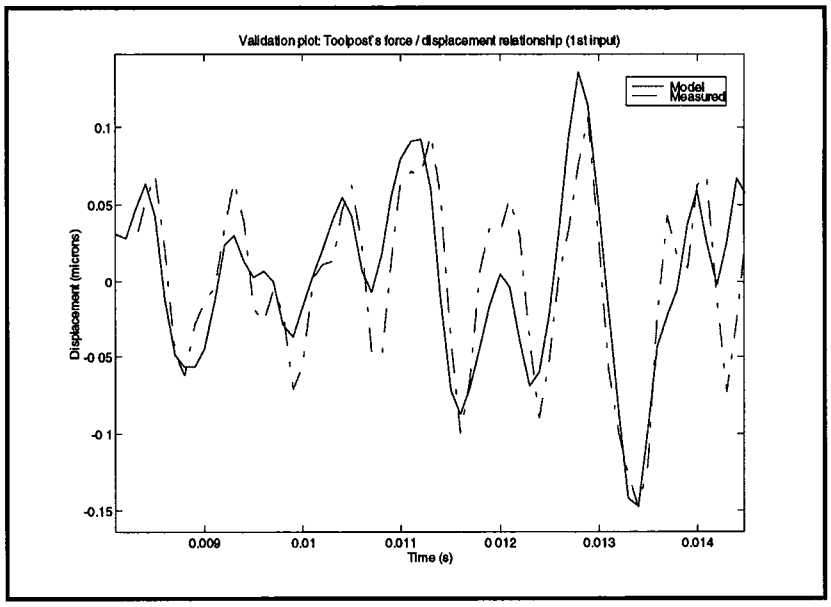


b.)

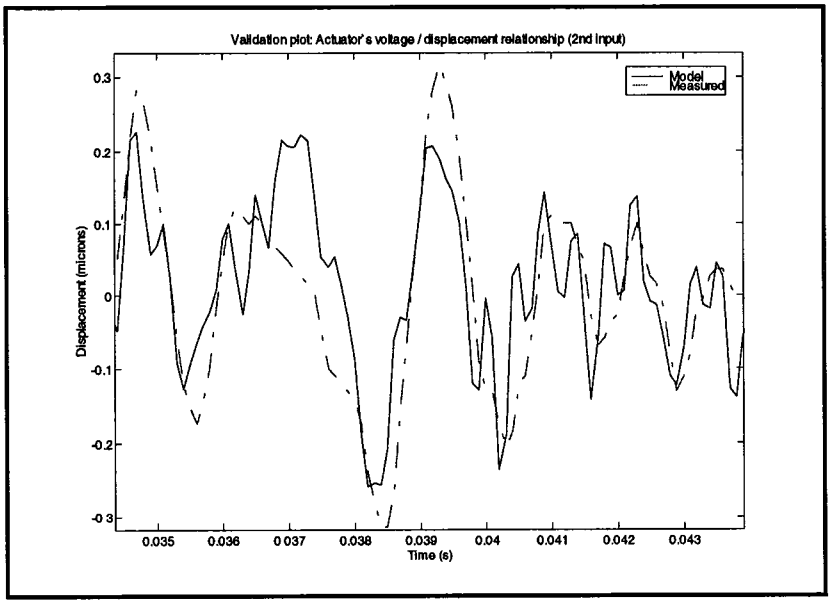
Figure 4-14: Model reduction effects on the frequency response

a.) 1st input b.) 2nd input

Although the frequency responses have changed slightly, the temporal response is still relatively consistent with measured data as shown in the following figures:



a.)



b.)

Figure 4-15: Model reduction effects on the predicted time response

a.) 1st input b.) 2nd input

5. CONTROL SYSTEM DESIGN

5.1 Introduction

Classical control methods were widely used in the nineteen sixties and early seventies because most of the controllers were built with analog circuits and classical design methods conformed nicely to that design approach. But with the advent of inexpensive microcomputers, state space (modern) techniques, more suitable for microcomputer implementation, came to dominate the literature. For the readers convenience, a brief discussion on the theory on optimal and adaptive control techniques is given. Each short tutorial is followed by a section detailing how the toolpost's controller is implemented using the aforementioned technique.

5.1.1 Linear Quadratic Control

The objective of optimal control is to determine the feedback control law

$$u = -Kx \tag{5-1}$$

that minimizes the following quadratic cost function,

$$J = \int_0^T (x^* Qx + u^* Ru)dt \tag{5-2}$$

which is subjected to the constraints of

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ y &= C_1 x + D_{11} w + D_{12} u \\ z &= C_2 x + D_{21} w + D_{22} u \end{aligned} \tag{5-3}$$

where $x \in \Re^n, u \in \Re^p$, $w \in \Re^m$, $y \in \Re^q$, $z \in \Re^q$, and Q and R are weighting matrices.

The following assumptions are made:

1. The pair (A,B_2) is stabilizable and (C_2,A) is detectable. This guarantees that the controller can reach all unstable states, and these states show up in the measurements.

2. rank D_{12} =m ; rank D_{21} = q . This insures that the controllers are proper.

3. Rank $\begin{bmatrix} A - j\omega I & B_2 \\ C_1 & D_{12} \end{bmatrix}$ = n + m for all frequencies

4. Rank $\begin{bmatrix} A - j\omega I & B_1 \\ C_2 & D_{21} \end{bmatrix}$ = n + q for all frequencies

This assumption simplifies the equations

5. $D_{11} = D_{22} = D_{12} = D_{21} = 0$
6. $C_2 = C_1 = C$

Figure 5-1: Assumptions for LQR design

Given assumption 5 and 6, equation (5-3), reduces to:

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ y &= C_1 x \end{aligned} \tag{5-4}$$

Using the Lagrange multiplier, $\lambda(t) \in \Re^n$, such that

$$J = \int_0^T \left(x^* Q x + u^* R u - \lambda^* (\dot{x} - Ax - B_2 u) \right) dt \tag{5-5}$$

and the Euler-Lagrange equation,

$$\frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{z}}\right) = \frac{\partial L}{\partial z} \tag{5-6}$$

for which:

$$L(x,\dot{x}) = x^*Qx + u^*Ru - \lambda^*(\dot{x} - Ax - B_2u) \tag{5-7}$$

and

$$z(t) = \begin{bmatrix} x(t) \\ \lambda(t) \\ u(t) \end{bmatrix} \tag{5-8}$$

the following 3 equations are produced:

$$\begin{aligned} &\frac{\partial}{\partial t}(\frac{\partial L}{\partial \dot{u}}) = 0 = \frac{\partial L}{\partial u} = u^*R + \lambda * B \\ &\frac{\partial}{\partial t}(\frac{\partial L}{\partial \dot{\lambda}}) = 0 = \frac{\partial L}{\partial \lambda} = -\dot{x} + Ax + Bu \\ &\frac{\partial}{\partial t}(\frac{\partial L}{\partial \dot{x}}) = -\dot{\lambda}^* = \frac{\partial L}{\partial x} = \lambda^*A + x^*Q \end{aligned} \tag{5-9}$$

From these equations, it is found that the optimal control signal u(t) is:

$$u(t) = -R^{-1}B^*\lambda(t) \tag{5-10}$$

and

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} A & -B_2R^{-1}B_2^* \\ -Q & -A^* \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = H \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{5-11}$$

with $x(0) = x_0$ and $\lambda(T) = 0$

Equation (5-11) is called the two-point boundary value problem (TPBVP) and H is called the Hamiltonian matrix.

If the following substitution is made:

$$\lambda = Px \tag{5-12}$$

then,

$$\dot{\lambda} = \dot{P}x + P\dot{x} = \dot{P}x + PAx - PB_2R^{-1}B_2{}^*Px = -Qx - A^*Px \tag{5-13}$$

Eliminating x, equation (5-13) becomes the celebrated Riccati equation:

$$-\dot{P} = PA + A^*P - PB_2R^{-1}B_2{}^*P + Q, \quad P(T) = 0 \tag{5-14}$$

This means that finding the solution to the TPBVP is simply a matter of solving for P in equation (5-14). As $\dot{P} \to 0$, $P(t) \to \overline{P}$, therefore, if $\{A, B_2\}$ is controllable, and $Q=C^*C$ where $\{A,C\}$ is observable, then we have the algebraic Riccati equation (ARE):

$$PA + A^*P - PB_2R^{-1}B_2{}^*P + C^*C = 0 \tag{5-15}$$

Substituting the solution P of equation (5-15) into equations (5-12) and (5-10), it is found that the optimal solution is:

$$u(t) = -R^{-1}B_2{}^*Px = -Kx \tag{5-16}$$

This feedback law produces the following closed loop system:

$$\begin{aligned} \dot{x} &= A_{cl}x + B_1w \\ y &= Cx \end{aligned} \tag{5-17}$$

Where $A_{cl} = (A - B_2R^{-1}B_2{}^*P)$.

The block diagram is illustrated in the following figure:

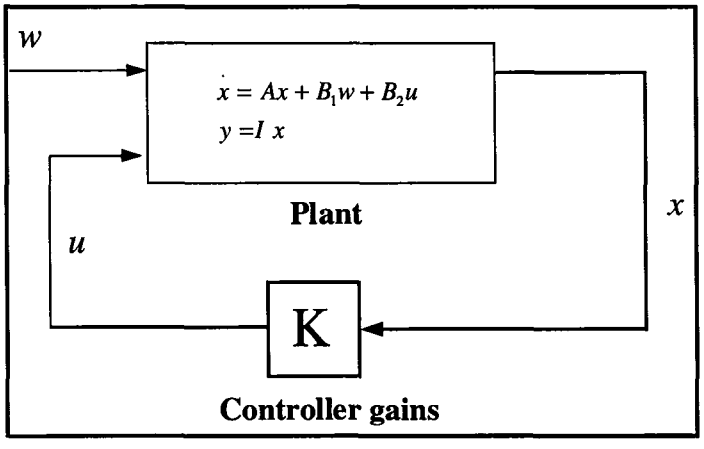


Figure 5-2: Full information controller

5.1.2 Observer / Estimator

In the previous section, an optimal, stabilizing control law was developed. The downside to implementing this type of control law is the fact that all of the states have to be measured. This is not practical and, in many cases, unnecessary. Fortunately, estimators are able to use measured information to determine the values of the remaining states. In particular, the Kalman filter provided an optimal solution to the estimator problem.

Consider the plant from equation (5-4)

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ y &= Cx \end{aligned} \tag{5-18}$$

in which C_1=C,

Also consider a duplicate system based on estimated states

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + B_1 w + B_2 u \\ \hat{y} &= C\hat{x} \end{aligned} \tag{5-19}$$

where the carets above the variables represent estimates of the actual values.

The idea of an estimator is to use the information between the actual output and the estimated output as a correction mechanism to drive the estimation process. Therefore, the following closed loop observer is proposed:

$$\dot{\hat{x}} = A\hat{x} + B_1 w + B_2 u + L(y - C\hat{x}) \tag{5-20}$$

where L is the observer gain that operates on the estimation error. From equations (5-18) and (5-19) ,the error, ($e = x - \hat{x}$) , produces the following state equation:

$$\dot{e} = (A - LC)e \tag{5-21}$$

This means that error dynamics can be arbitrarily tuned to converge to zero for any initial condition. It is desired to choose an observer gain that will make the error dynamics converge at least 2 or 3 times faster than the plant dynamics. Of course, there is a limit to how large L can be. This is due to the fact that large gains can cause actuator saturation, make the system more susceptible to noise, or require too fast a sampling rate. Simplifying equation (5-20) and using $u = -k\hat{x}$, the block diagram of the closed loop system is shown in the following diagram:

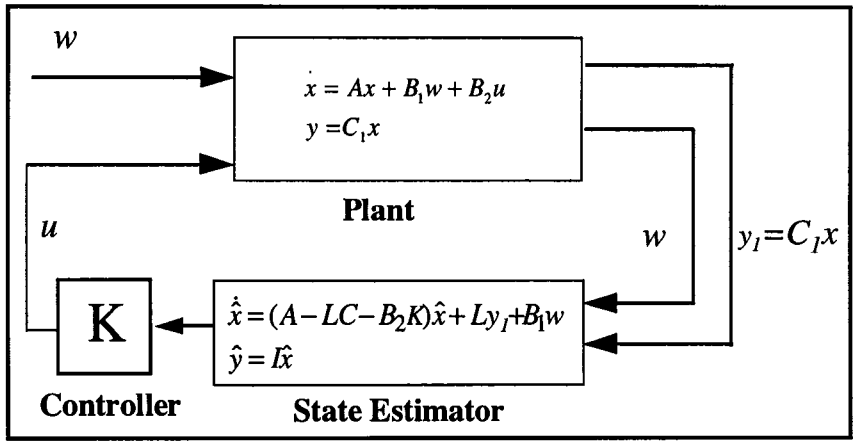


Figure 5-3: System with both controller and estimator

5.1.2.1 Optimal Observer / Kalman Filter

In a manner similar to that for finding an optimal controller, and optimal observer can also be developed. This is referred to as the stationary Kalman filter. The objective is to get an estimate of the states based on noise-corrupted measurements such that the variance of the error is minimized.

It is assumed that, for the plant,

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ y &= Cx + v \end{aligned} \qquad (5\text{-}22)$$

the following conditions hold:

$$E\{w(t)\}=0;\quad E\{v(t)\}=0;\quad E\{w(t)\,w(t+\tau)'\}=Q;\quad E\{v(t)\,v(t+\tau)'\}=R;\quad E\{w(t)\,v(t+\tau)'\}=0 \tag{5-23}$$

for all t and τ, where *w(t) and v(t)* are the disturbance and measurement error signals, respectively and $E(\cdot)$ is the expectation operator.

The objective is to minimize the cost function:

$$J=\frac{1}{2}\int_0^T(\tilde{x}'\Gamma Q\Gamma'\tilde{x}+y'Ry)dt \tag{5-24}$$

where Γ is weighting matrix on Q. In a manner similar to the controller design, the optimal observer gain is found to be:

$$L=\Sigma C^{*}R^{-1} \tag{5-25}$$

where Σ is the solution to the following algebraic Ricatti equation:

$$A\Sigma+\Sigma A^{*}+\Gamma Q\Gamma'-\Sigma C^{*}R^{-1}C\Sigma=0 \tag{5-26}$$

Σ is also the estimation error covariance, therefore, the trace of Σ represents the error variance and is a good indication on how well the filter is performing.

One of the downfalls to developing the Kalman filter in the manner discussed above is that it is not necessarily possible to determine the variances as illustrated in equation (5-23). Usually, R is taken to be the identity matrix with an order consistent with the number of inputs, Q is taken to equal $C^{*}C$, and trial and error is done by varying Γ.

After the controller and observer gains are calculated, the state variable representation of the controller becomes:

$$\dot{\hat{x}} = A_{cont.}\hat{x} + B_1 w + Ly$$
$$u = -k\hat{x} \tag{5-27}$$

where $\hat{x}$ is the controller's state vector and $A_{cont.} = A - LC - B_2 k$.

The closed loop system is the union of the controller poles and the observer poles:

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} A & -B_2 k \\ LC & A - B_2 k - LC \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + \begin{bmatrix} B_1 \\ B_1 \end{bmatrix} w$$
$$u = \begin{bmatrix} -k & 0 \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix} \tag{5-28}$$

This controller model represents the continuos time controller. In order to implement it using a digital computer, it must first be discretized. This is discussed in the next section.

5.1.3 Discrete Time Model

The equation that governs the response of a discrete time, linear, time-invariant system is the following:

$$x((k+1)T) = e^{AT} x(kT) + \int_0^T e^{A\lambda} B u(kT)\, d\lambda \tag{5-29}$$

where T is the constant sampling interval, t is the time, and $\lambda = T - t$.

Define:

$$G(T) = e^{AT}$$
$$H(T) = \left(\int_0^T e^{A\lambda} d\lambda \right) B \tag{5-30}$$

Therefore, equation (5-29) can now be written as:

$$x((k+1)T) = G(T)x(kT) + H(T)u(kT) \tag{5-31}$$

where G(T) and H(T) depends on the sampling frequency. If the matrix A is nonsingular, then:

$$H(T) = \left(\int_0^T e^{A\lambda} d\lambda \right) B = A^{-1}(e^{AT} - I)B = (e^{AT} - I)A^{-1}B \tag{5-32}$$

and the output equation still has the form:

$$y(kT) = Cx(kT) + Du(kT) \tag{5-33}$$

As mentioned is Section 4.3, the desired sampling for a reasonably smooth time response is:

$$5 \leq \frac{w_s}{w_b} \leq 20 \tag{5-34}$$

where w_s and w_b are the sampling frequency and closed-loop bandwidth respectively. Remembering from equation (5-28), the closed-loop bandwidth is now the union of the controller and observer poles. Therefore, since the system must be sampled at least 5 times the observer bandwidth, and the observer should be 2-3 times faster than the controller bandwidth, the sampling frequency must be at least 10 times that of the controller bandwidth. Clearly, this shows the limitations the digital control system has on its ability to control. The objective is to place the controller poles as far into the left half s-plane as feasibly possible. The following table illustrates the farthest controller and observer poles for control loop frequencies ranging from 10 to 100 kHz.

Control Loop frequency (kHz)	Farthest observer poles (rad/s) (10^4)	Farthest controller poles (rad/s) (10^4)	q (observer) (10^4)	q (controller) (10^4)	attenuation (random excitation) (dB)
100	-12.5	-6.28	20	7.5	90.2
90	-11.3	-5.65	16	6.5	86.9
80	-10.0	-5.02	13	5.2	86.9
70	-8.79	-4.39	10	4	86.9
60	-7.54	-3.77	7.7	3	75.5
50	-6.26	-3.14	5.3	2.1	65.5
40	-5.02	-2.51	3.5	1.4	58.8
30	-3.77	-1.88	2	.9	47.74
20	-2.51	-1.25	1	.48	31.9
15	1.88	.942	.6	.32	23
10	-1.26	-.628	.3	.23	13.87

Table 5-1: Control loop and corresponding farthest controller pole

In searching for the feedback controller gain k, observer poles L, and the corresponding attenuation, a program was written that incremented Q in equation (5-2) and kept R constant at $R=1$. This was done by changing q_{cont} in the equation:

$$Q = q_{cont} \cdot C \cdot C \qquad (5\text{-}35)$$

until the maximum controller frequency was achieved. Also, for equation (5-24) , R was kept constant at $R=1$, and q_{obsv} in the equation:

$$\Gamma Q \Gamma = q_{obsv.} \cdot C^{'} C \qquad (5\text{-}36)$$

was varied until the maximum observer frequency was achieved. Simulations were conducted to determine the attenuation on a bandlimited random signal. The following chart illustrates the effect the control loop speed has on the attenuation ability of the controller.

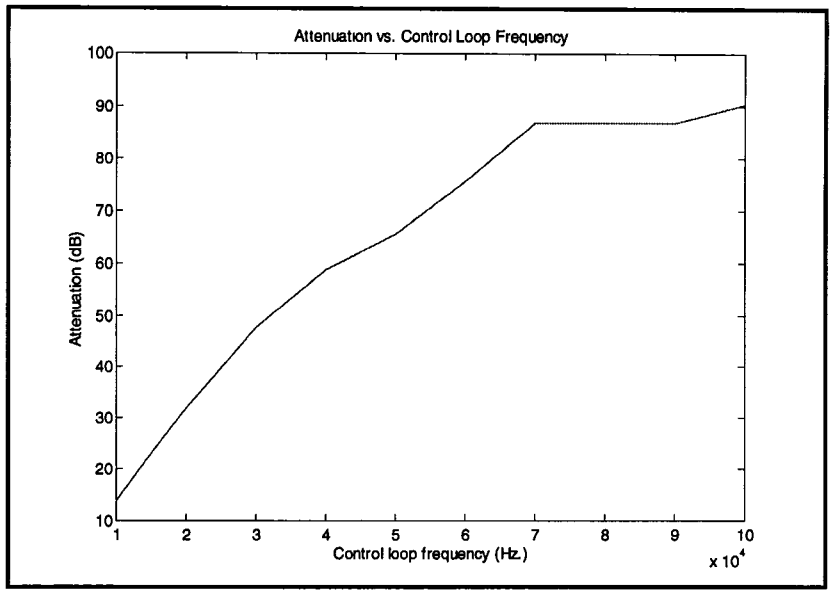


Figure 5-4: Effects of control loop frequency on vibration attenuation

We see that up until around 70 kHz, the attenuation is almost linear with respect to the increased sampling frequency. After 70 kHz, the attenuation drops off to around 87 dB.

The simulations were done using Matlab's Simulink software in which the output of the controlled and uncontrolled systems were compared. The control diagram is shown in the following figure:

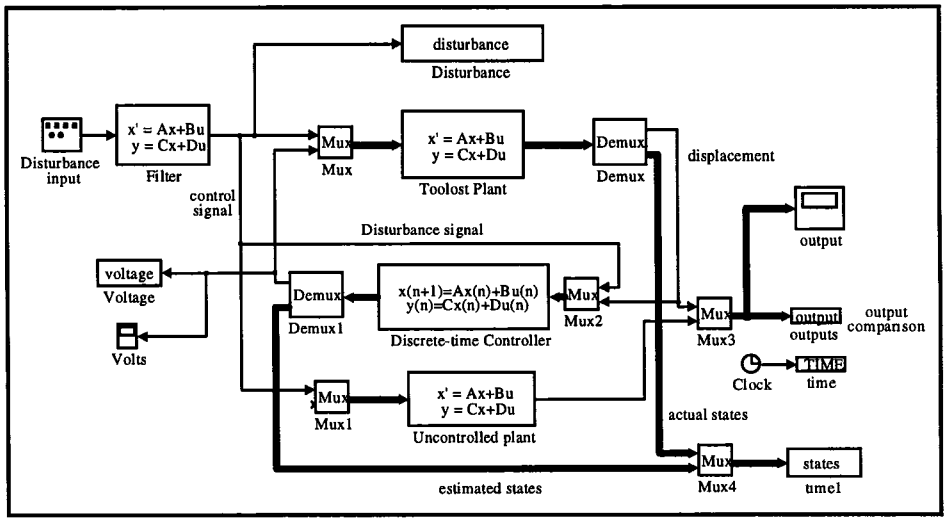


Figure 5-5: Block diagram of simulated system in Simulink

where the *demux* block was used to break the output into its states and displacements for comparison.

5.2 Toolpost's Controller Design

The major goal of the controller is to minimize the vibration along the radial direction of the workpiece during machining. This corresponds to minimizing the displacement of the tool tip. For this reason, a disturbance attenuation control system design approach is used. This method attempts to minimize the transfer function that maps the disturbance input (force), to the output (tool tip displacement). This means that an attempt will be made to move the closed loop poles as far into the left half plane as feasibly possible.

The data acquisition board is able to convert at a rate of 100 kHz (50 kHz for two channels). As will be shown in chapter 6, taking in consideration computation time and data servicing time, the current fastest control loop is about 15 kHz. Therefore, interpolating Table 5-1, the fastest controller frequency can be 9240 rad/s. Using a value of q=3200 in equation (5-35), the following controller gains were determined:

$$K = [\,381.71 \quad -424.85 \quad 239.68 \quad -50.46] \tag{5-37}$$

Using $\Gamma Q \Gamma' = 6000 \cdot C'C$ and $R=1$, the following observer gains were determined:

$$L = [315.89 \quad -587.76 \quad -172.36 \quad -176.98]^T \tag{5-38}$$

The following bode plots show the effect of moving the poles had on reducing the transfer function that mapped the tool tip displacement to the excitation force:

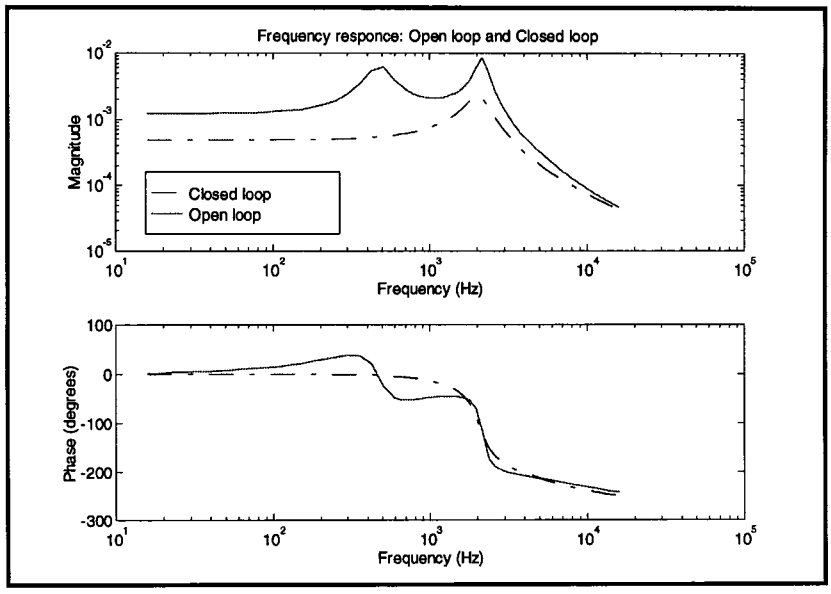
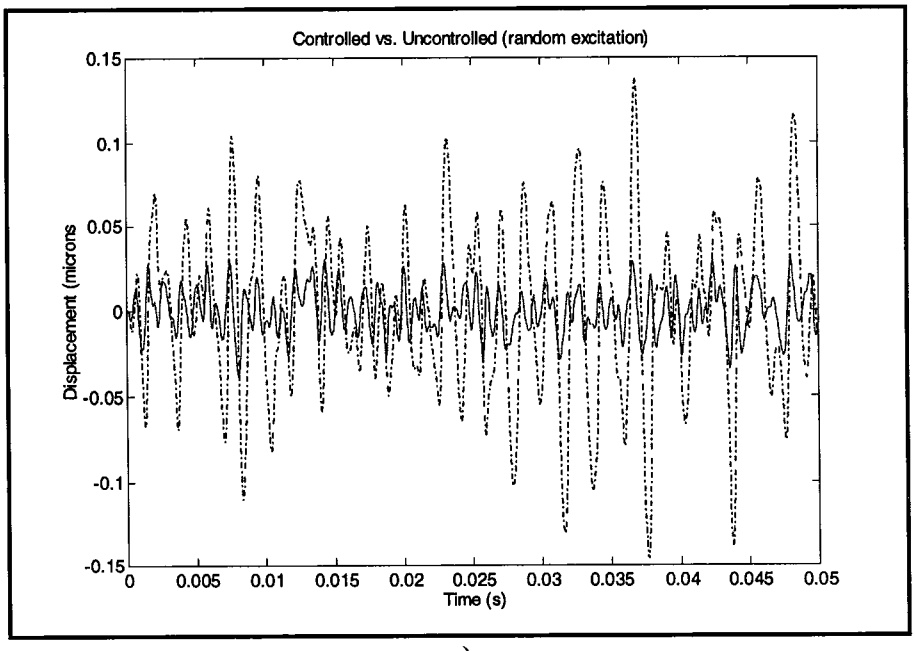
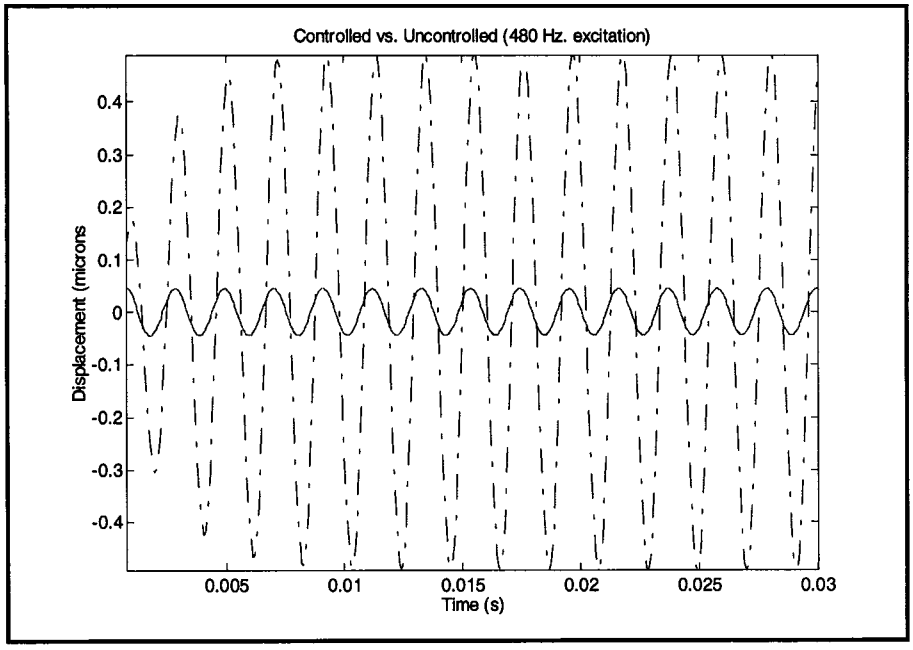


Figure 5-6: Bode Plot: Comparison between open-loop and closed-loop frequency responses

The magnitude response curve actually shifted downward reflecting at least a 20 dB attenuation in the displacement across much of the bandwidth of the system. The following plots will illustrate this attenuation for various inputs.



a.)



b.)

Figure 5-7: Tool tip displacement for controlled and uncontrolled system:

a.) Random @ 75 N b.) 450 Hz @ 75 N

From equation (5-28), the closed loop system now looks like:

$$\dot{\chi}(t) = \left[\begin{array}{cccc:cccc} -993.68 & -2945.9 & 901.42 & 741.8 & -9955.66 & -424.90 & 4963.89 & -516.05 \\ 2945.55 & -40.93 & 312.83 & 112.92 & 1887.76 & 80.57 & -941.24 & 97.85 \\ 896.26 & -312.76 & -1416.55 & -13721.4 & 5461.2 & 233.08 & -2722.95 & 283.08 \\ -692.6 & 92.17 & 13607.4 & -498.84 & -2722.05 & -116.18 & 1357.21 & -141.10 \\ \hdashline 1317.7 & 2600.1 & -7529.49 & -4219.31 & -24667 & -5970.9 & 13394.8 & 4445.05 \\ 2935.03 & 556.32 & -1611.01 & -902.77 & 1898.27 & -516.68 & 982.6 & 1113.53 \\ -5332.35 & -1010.71 & 2926.87 & 1640.13 & 11689.8 & 931.04 & -7066.37 & -15078.4 \\ -5136.04 & -973.50 & 2819.11 & 1579.75 & 1721.39 & 949.5 & 12145.5 & -2219.69 \end{array}\right] \chi(t) + \left[\begin{array}{c} .5 \\ .06 \\ .38 \\ 1.95 \\ \hdashline .5 \\ .06 \\ .38 \\ 1.95 \end{array}\right] f(t) \qquad (5\text{-}39)$$

where $\chi = \left[x \quad \hat{x}\right]^T$.

Using equation (5-31), and a sampling frequency of 15 kHz, the state variable representation of the discrete time controller is:

$$x(k+1)=\begin{bmatrix} 0.688 & 0.460 & 0.024 & 0.090 \\ 0.188 & 0.523 & -0.003 & -0.148 \\ -0.124 & -0.033 & 0.65 & 0.531 \\ -0.002 & -0.026 & -0.726 & 0.554 \end{bmatrix} x(k)+\begin{bmatrix} 6.54x10^{-6} & 5.22x10^{-3} \\ -7.75x10^{-6} & -24.4x10^{-3} \\ 102.5x10^{-6} & -14.5x10^{-3} \\ 13.7x10^{-6} & -4.66x10^{-3} \end{bmatrix} \begin{Bmatrix} f(k) \\ d(k) \end{Bmatrix} \quad (5\text{-}40)$$

$$v(k)=\begin{bmatrix} -381.71 & 424.85 & -239.68 & 50.46 \end{bmatrix} x(k)$$

A subroutine was written in *C* that computed the controller output using equation (5-40). In chapter 6, there will be a discussion on how the above algorithm and various instrumentation are used for real time microprocessor control implementation.

5.3 Adaptive Control

5.3.1 Introduction

Development of adaptive control systems has increased substantially over recent years. This can be attributed to the increase in availability, and decrease in cost, of dedicated computers and software. The term "adaptive control" in the control literature refers to systems that attempt to eliminate the effects of changes in the controlled system itself, in addition to the effects of external disturbances acting on the system. An adaptive control system modifies the parameters of the controller, or generates an auxiliary control signal, in order to maintain some index of performance at a desired level despite changes in the controlled system [39].

The adaptive control scheme can be thought of as conducting an on-line, real-time, system identification as a means to determining the controller gains. There are two ways in which the controller gains are selected: explicitly and implicitly. Explicit adaptive control deals with designing a controller whose parameters are functions of the plant's

parameters. Therefore, if the plant's parameters change, the controller gains change accordingly. Whereas implicit adaptive control deals with a plant model that is parameterized in terms of the controller which allows the controller gains to be computed without involving the plant parameter estimates. A diagram of a typical adaptive control loop is shown below.

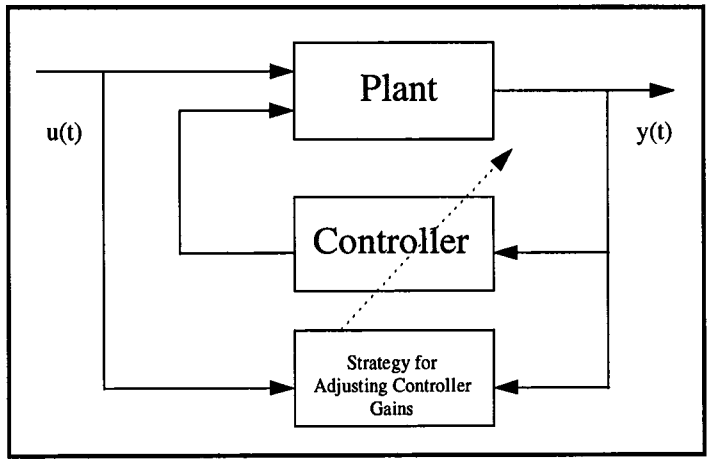


Figure 5-8: Adaptive control loop

5.3.2 Adaptive Control Schemes

There are various control schemes that can be termed adaptive. These are gain scheduling, model reference adaptive control (MRAC), and adaptive pole placement.

Gain scheduling involves measuring the appropriate controller gains for various plant conditions off line. Then the control scheme is to continuously identify the plant's condition and control accordingly. If the process parameters vary in a predictive manner, then this design approach would be appropriate.

Model reference adaptive control is a method in which the controller parameters change in a manner such that the cascaded model of the feed-forward controller and the plant matches a pre-described reference model.

Adaptive pole placement, often found in literature as self-tuning controller, has a fixed structure with variable parameters that sets its own setting to control and unknown process. This is particularly useful when the plant is unknown.

Out of the 3 adaptive control schemes mentioned, adaptive pole placement is the most relevant in this thesis research. As the name mentions, this is similar to the optimal approach to pole placement except the controller gains are not constant but are updated continuously during operation. This method has an advantage that it can be used to control a wide class of linear time invariant plants with unknown parameters. As mentioned earlier, there is indirect APPC and direct APPC. Unlike indirect APPC, direct APPC can only be applied to scalar plants and to plants where the desired parameters of the pole placement controller can be expressed in the form of the linear of bilinear parametric models. Indirect APPC is a more general class of adaptive control schemes and will be the one discussed further.

The parameter estimation scheme is very much like that used for system identification discussed in chapter 4. The differences are based on the way the two operations are used. Parameter estimation for adaptive control has the time constraint factor, it has to be done in real time. The reason why off-line parameter estimation was conducted in the first place is because, for real-time parameter estimation, the order of the

model must be known a priori. Therefore, as shown in the previous chapter, many model orders were tested and the current model was selected based on its size and accuracy.

5.4 Indirect Adaptive Pole Placement

The indirect adaptive approach involves first estimating the plant's parameters and then computing the controller whose parameters are based on these estimates. Since the closed loop poles have already been determined via LQR methods in the previous section, the controller gain can, adaptively, be determined by estimating the parameters and setting equal the two characteristic equations. This method is designed for a SISO system and is illustrated for the 4th order toolpost model. Only the force / displacement transfer function will be used in this approach Remember, this means that only the 1st column of the B matrix is used and the A matrix stays the same. Therefore, the poles of the system are not different, only the zeros. The equation for the estimates of the voltage/displacement transfer function has the following form:

$$\hat{y}(s) = G(s)u(s) \tag{5-41}$$

where:

$$G(s) = \frac{\hat{b}_1 s^3 + \hat{b}_2 s^2 + \hat{b}_3 s + \hat{b}_4}{s^4 + \hat{a}_1 s^3 + \hat{a}_2 s^2 + \hat{a}_3 s + \hat{a}_4} \tag{5-42}$$

Note that the order of the transfer function needs to be known a priori. This was the reason for the system identification. The benefit in adaptive control is that the system will still be able to perform in the same manner, even if the parameters of the system change. For example, change can occur during machining due to tool wear which can change the dynamics of the system. Also, the parameters can drift with aging. Changes can also be

disguised as parameter uncertainties in modeling. Now equation (5-42) can be written in the following form:

$$\hat{y}^{(4)} = -\hat{a}_1\ddot{y} - \hat{a}_2\ddot{y} - \hat{a}_3\dot{y} - \hat{a}_4 y + \hat{b}_1 u^{(3)} + \hat{b}_2 u^{(2)} + \hat{b}_3\dot{u} + \hat{b}_4 u \tag{5-43}$$

Define:

$$\theta^T = \begin{bmatrix} \hat{a}_1 & \hat{a}_2 & \hat{a}_3 & \hat{a}_4 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4 \end{bmatrix} \tag{5-44}$$

and

$$Y = \begin{bmatrix} -s^3 y & -s^2 y & -sy & -y & s^3 u & s^2 u & su & u \end{bmatrix}^T \tag{5-45}$$

then:

$$\hat{y}^4 = \theta^T Y \tag{5-46}$$

Now define $\Lambda(s)$ to be any arbitrary, stable, 4th order polynomial to be used to filter the signal vector to get rid of any differentiators, and obtain

$$\phi = \frac{Y}{\Lambda(s)} \tag{5-47}$$

and the following relation is produced:

$$\hat{y} = \frac{y^4}{\Lambda(s)} = \frac{s^4}{\Lambda(s)} y = \frac{\theta^T Y}{\Lambda(s)} = \theta^T \phi \tag{5-48}$$

As will be discussed in the next section, the objective is to get the parameter vector to converge to the actual parameters in a reasonable amount of time, i.e. $\theta^T \rightarrow \theta^{*T}$. where θ^T are the estimates of the parameters and θ^{*T} are the actual values. Once the parameters of equation (5-44) are determined, the adaptive controller and observer gains can be determined. For illustration purposes, the controllable canonical realization of the estimated plant is shown below:

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\hat{a}_4 & -\hat{a}_3 & -\hat{a}_2 & -\hat{a}_1 \end{bmatrix} \cdot x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \cdot u \qquad (5\text{-}49)$$

$$y = \begin{bmatrix} \hat{b}_4 & \hat{b}_3 & \hat{b}_2 & \hat{b}_1 \end{bmatrix} \cdot x$$

This is a convenient realization to determine the necessary controller gain estimate, $\hat{K}$, to be used in the feedback law, $u = -\hat{K}x$, by solving the following equation:

$$\det(sI - \hat{A} + B\hat{K}) = \tilde{A}(s) \qquad (5\text{-}50)$$

where $\hat{A}$ is an estimate of A and $\tilde{A}(s)$ is the characteristic polynomial of the desired closed loop system defined by:

$$\tilde{A}(s) = s^4 + \tilde{a}_1 s^3 + \tilde{a}_2 s^2 + \tilde{a}_3 s + \tilde{a}_4 \qquad (5\text{-}51)$$

Solving this equation involves setting corresponding coefficients equal to one another. This is illustrated below:

$$\det\left\{ \begin{bmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\hat{a}_4 & -\hat{a}_3 & -\hat{a}_2 & -\hat{a}_1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ k_1 & k_2 & k_3 & k_4 \end{bmatrix} \right\} = \tilde{A}(s) \qquad (5\text{-}52)$$

reduces to:

$$s^4 + (\hat{a}_1 + k_4)s^3 + (\hat{a}_2 + k_3)s^2 + (\hat{a}_3 + k_2)s + (\hat{a}_4 + k_1) = s^4 + \tilde{a}_1 s^3 + \tilde{a}_2 s^2 + \tilde{a}_3 s + \tilde{a}_4 \qquad (5\text{-}53)$$

Now set this equal to the desired characteristic equation where the closed loop poles are derived from optimal control problem in the previous section and we get:

$$\hat{K} = -\begin{bmatrix} \hat{a}_4 - \tilde{a}_4 & \hat{a}_3 - \tilde{a}_3 & \hat{a}_2 - \tilde{a}_2 & \hat{a}_1 - \tilde{a}_1 \end{bmatrix} \qquad (5\text{-}54)$$

which clearly shows how the controller gains are dependent on the estimates. As $\hat{A} \rightarrow A$, $\hat{K} \rightarrow K$. Notice how $\hat{K}$ is only dependent on the coefficients of the system's characteristic equation.

Therefore, the equation for the closed loop system becomes:

$$\dot{x} = (\hat{A} - B_2\hat{K}) \cdot x + B_1 f \qquad (5\text{-}55)$$
$$u = -\hat{K}x$$

In the next section, two parameter estimation update laws are discussed.

5.4.1 Parameter Estimation Methods

Because the estimation process has to be done in real time, the parameter estimation algorithm discussed in Section 4.1 had to have some adjusting. There are two popular methods in doing this: Gradient method and the Least squares method.

In order to effectively implement adaptive control, there is a need for reliable process identification methods. This method involves estimating the type of model used and using the input and output data to estimate its parameters. Unlike the off-line estimation schemes, the on-line ones are designed to be used with either stable or unstable plants.

5.4.1.1 Gradient Method

The gradient method, also known as the method of steepest descent, is one of the most widely used methods for solving optimization problems.

The object is to minimize the following cost function:

$$J(\theta) = \frac{\varepsilon^2 m^2}{2} = \frac{(y - \theta^T \phi)^2}{2m^2} \tag{5-56}$$

where $y = \theta^{*T}\phi$, and $m^2 = 1 + \varphi^T \varphi$ is a normalizing signal which ensures that $\frac{\phi}{m}$ is bounded.

This cost function is subjected to the constraint:

$$\varepsilon = \frac{y - \hat{y}}{m^2} \tag{5-57}$$

therefore,

$$\nabla J(\theta) = -\frac{(y - \theta^T \phi)\phi}{m^2} = -\varepsilon\phi \tag{5-58}$$

Since the minimizing trajectory is:

$$\dot{\theta} = -\Gamma \nabla J(\theta) \tag{5-59}$$

where Γ is a positive definite symmetric scaling matrix called the adaptive gain.

Therefore, the gradient adaptive update law becomes:

$$\dot{\theta} = \Gamma \varepsilon \phi \tag{5-60}$$

5.4.1.2 Least Squares Method with Data Weighting

The method of least squares is a popular method for parameter estimation. This method is based on minimizing the following discrete time cost function:

$$J_N(\theta) = \frac{1}{2}\left[Y_N - \Phi_{N-1}\theta\right]^T \left[Y_N - \Phi_{N-1}\theta\right] + \frac{1}{2}\left[\theta - \hat{\theta}(0)\right]^T P_0^{-1}\left[\theta - \hat{\theta}(0)\right] \tag{5-61}$$

where $Y_N^T = [y(1) \quad y(2) \quad \cdots \quad y(N)]$, $\Phi_{N-1} = [\phi(0) \quad \phi(1) \quad \cdots \quad \phi(N-1)]^T$ and the second term on the right side of equation (5-61) takes into account the initial conditions where P_0 is a measure of the confidence in the initial estimate of $\hat{\theta}(0)$.

Differentiating with respect to θ , and setting the results equal to zero yields:

$$\left[\Phi_{N-1}^T \Phi_{N-1} + P_0^{-1}\right]\theta = P_0^{-1}\hat{\theta}(0) + \Phi_{N-1}^T Y_N \tag{5-62}$$

Notice that this has the same form as equation (4-10) with the exception of the initial condition term. Equation (5-63) results in the parameter estimate being:

$$\theta_N = \left[\Phi_{N-1}^T \Phi_{N-1} + P_0^{-1}\right]^{-1}\left[P_0^{-1}\hat{\theta}(0) + \Phi_{N-1}^T Y_N\right] \tag{5-63}$$

Let

$$P(N-1)^{-1} = \Phi_{N-1}^T \Phi_{N-1} + P_0^{-1} \tag{5-64}$$

then

$$\theta_N = P(N-1)^{-1}\left[P_0^{-1}\hat{\theta}(0) + \Phi_{N-1}^T Y_N\right] \tag{5-65}$$

Rewriting equation (5-64) in terms of $\phi(\cdot)$ yields:

$$P(N-1)^{-1} = P(N-2)^{-1} + \phi(N-1)\phi(N-1)^T \tag{5-66}$$

which can be rewritten as:

$$P(N-1) = P(N-2) - \frac{P(N-2)\phi(N-1)\phi(N-1)^T P(N-2)}{1+\phi(N-1)^T P(N-2)\phi(N-1)a(N-1)} \tag{5-67}$$

Also, substituting equation (5-66) into (5-65) yields:

$$\theta(k) = \theta(k-1) + \frac{P(k-2)\phi(k-1)}{1+\phi(k-1)^T P(k-2)\varphi(k-1)}[y(k) - \phi(k-1)^T\theta(k-1)] \tag{5-68}$$

Equations (5-67) and (5-68) are referred to as the least squares update law.

Making the assumption that recent data is more informative than past data, an exponential data weighting modification can be added to the above law to improve its convergence properties. This involves minimizing the exponentially weighted cost function:

$$S_N(\theta) = \lambda(N-1)S_{N-1}(\theta) + \left[y(N) - \phi(N-1)^T\theta\right] \tag{5-69}$$

to produce:

$$\theta(k) = \theta(k-1) + \frac{P(k-2)\phi(k-1)}{\lambda(k-1) + \phi(k-1)^T P(k-2)\varphi(k-1)}[y(k) - \phi(k-1)^T\theta(k-1)] \tag{5-70}$$

and

$$P(k-1) = \frac{1}{\lambda(k-1)}\left[P(k-2) - \frac{P(k-2)\phi(k-1)\phi(k-1)^T P(k-2)}{\lambda(k-1) + \phi(k-1)^T P(k-2)\phi(k-1)}\right] \tag{5-71}$$

As mentioned in chapter 5, there needs to be a persistently exciting signal for the estimated parameter vector to converge to the actual value. This is somewhat contradictory of the purpose of the controller which is to attenuate the excitation signal (disturbance).

5.4.2 Applying Parameter Estimation to the MISO Smart Toolpost Model

For the Smart Toolpost model:

$$\dot{x}(t) = \begin{bmatrix} -993.68 & -2945.9 & 901.42 & 741.8 \\ 2945.55 & -40.93 & 312.83 & 112.92 \\ 896.26 & -312.76 & -1416.55 & -13721.4 \\ -692.6 & 92.17 & 13607.4 & -498.84 \end{bmatrix} x(t) + \begin{bmatrix} .5 & 13.82 \\ 0.06 & -2.62 \\ 0.38 & -7.68 \\ 1.96 & 3.78 \end{bmatrix} \begin{Bmatrix} f_c(t) \\ v(t) \end{Bmatrix} \tag{5-72}$$

$$y(t) = \begin{bmatrix} 13.83 & 2.62 & -7.59 & -4.25 \end{bmatrix} x(t)$$

there are two inputs and one output. Section 5.4 discussed adaptive pole placement for a SISO system. This section will discuss how that method can be extended to the present multiple input / single output, (MISO), system. The complete toolpost model is actually a combination of the mapping of the force to the displacement and also the mapping of the control voltage to the tool's tip displacement as was shown in chapter 4. If transfer functions were to be drawn from the above model to produce two SISO systems, the analysis can be applied to the plants individually. This is illustrated in the following diagram:

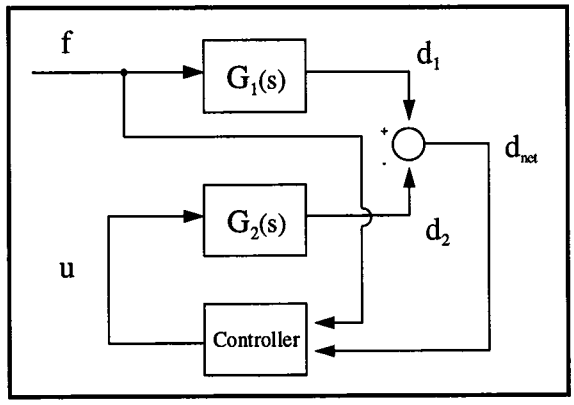


Figure 5-9: Decoupled Smart Toolpost System

where $G_1(s)$ and $G_2(s)$ represent the two transfer functions and d_1 and d_2 represent their individual contributions to the net displacement. Remembering that the poles of both transfer functions are the same as those in the A matrix in equation (5-72). Also from equation (5-53), the controller gains are only dependent on the coefficients of the characteristic equation. For this reason, $G_1(s)$ can be used for the parameter estimation. The only problem is that in operation, d_1 is not measurable directly, only the net

displacement. It turns out that the use of the correct form of the observer will allow the system to be decoupled and d_1 can be determined.

Decoupling could be achieved in equation (5-72) by changing the system's realization. Because the system has imaginary poles, the Jordon canonical realization, in which the eigenvalues are placed along the diagonal of the A matrix, was not attempted. A better approach was to use the modal canonical realization. Although this doesn't fully decouple the system, it turns out that the necessary information is still obtainable. For the toolpost model given in equation (5-72), the modal canonical realization is:

$$\dot{x}(t)=\begin{bmatrix} -475 & 2940 & 0 & 0 \\ -2940 & -475 & 0 & 0 \\ 0 & 0 & -1000 & 13643.4 \\ 0 & 0 & -13643.4 & -1000 \end{bmatrix} x(t)+\begin{bmatrix} -0.56 & -17.91 \\ 0.02 & -5.56 \\ 1.28 & 11.47 \\ 2.49 & -4.31 \end{bmatrix} \begin{Bmatrix} f_c(t) \\ v(t) \end{Bmatrix} \qquad (5\text{-}73)$$

$$y(t)=\begin{bmatrix} -9.23 & -0.11 & 3.17 & -5.39 \end{bmatrix} x(t)$$

By inspection, it can be seen that the 1st two states correspond to $G_2(s)$ and the second two states correspond to $G_1(s)$. Using the output equation:

$$y=\begin{Bmatrix} \hat{x} \\ d_1 \end{Bmatrix}=\begin{Bmatrix} I \\ \tilde{c} \end{Bmatrix}\hat{x}=\begin{bmatrix} I & & & \\ 0 & 0 & 3.17 & -5.39 \end{bmatrix}\hat{x} \qquad (5\text{-}74)$$

will produce the value d_1 to be used by the parameter estimator. This particular state order need also be reflected in the observer so that the estimated states are in the correct order. The adaptive control scheme using the Smart Toolpost model is shown in the following diagram:

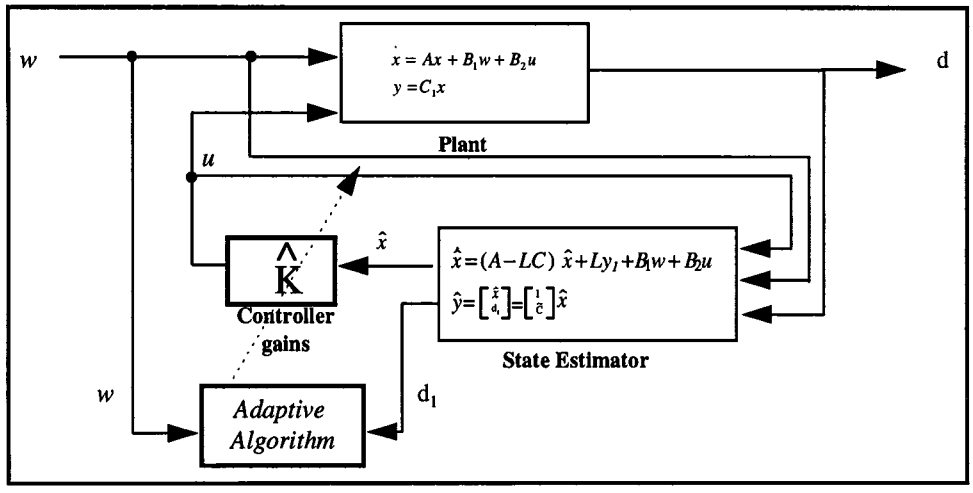
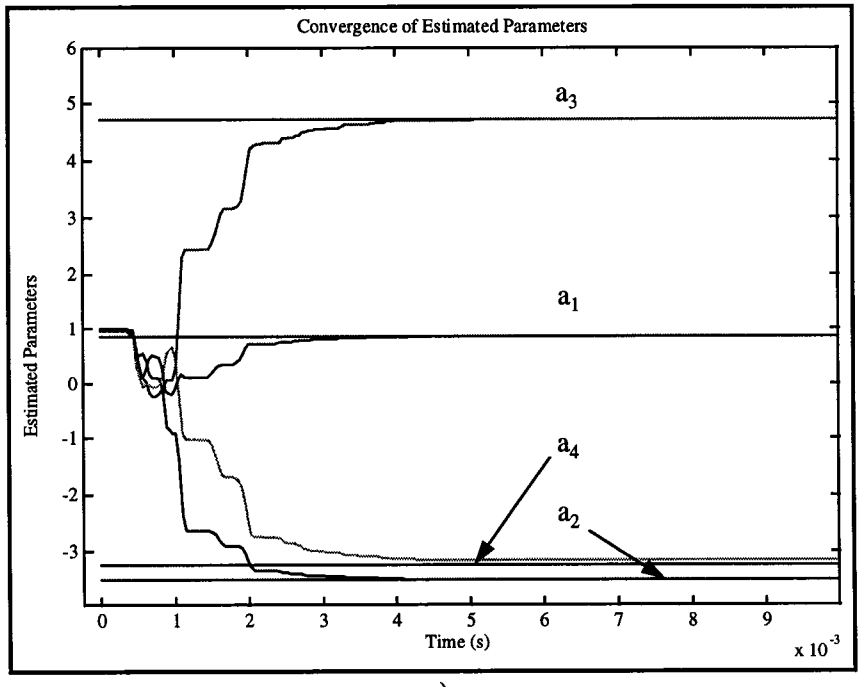
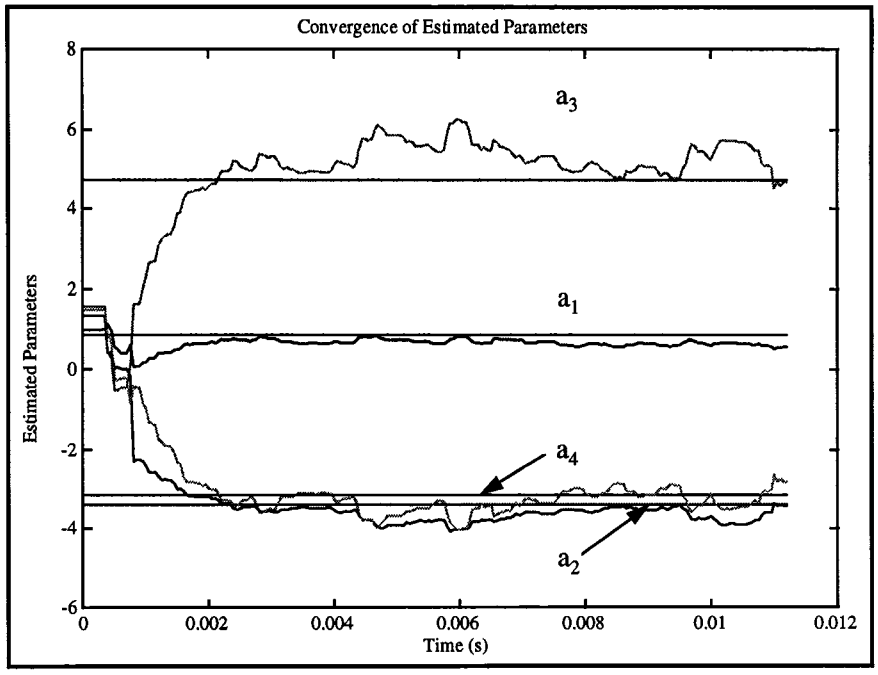


Figure 5-10: Block diagram of the Smart Toolpost's Adaptive Control System

Following are two plots which illustrates the parameter convergence of the smart toolpost system for the open loop and closed loop force / displacement mappings respectively.



a.)



b.)

Figure 5-11: Convergence of Smart Toolpost's parameter estimator

a.) open loop b.) closed loop

Notice that for the open loop, the parameters converged to the exact values, but for the closed loop, the estimates fluctuated about the actual values. This is attributed to the fact that an observer, which is used to estimate the states, is used to estimate the displacement due to the disturbance. Since the observer cannot exactly estimate the states, the parameter identifier cannot exactly estimate the parameters. Nevertheless, all is not lost in this approach. Looking again at Figure 5-11, we see that the parameters converge to a region of the actual parameter. Therefore, although this identifier might not be suitable for adaptive control, any large deviations from the actual parameters would be an indication that something is wrong with the toolpost system. Therefore this technique can be used as a form of health monitoring.

6. IMPLEMENTATION

6.1 Digital vs. Analog Controllers

With the availability of inexpensive microprocessors, the use of digital control systems has rapidly increased in recent years [44]. There are also some advantages of working with discrete time control systems verses analog. For example, discrete time controllers offer greater flexibility when modifications are needed. To modify the controller, simply modifying the controller code is usually all that is needed, unlike the analog case in which components may have to be changed.

6.2 System Setup

The control system consists of a number of components including a power amplifier, a digital to analog converter / analog to digital converter (DAC/ADC) , and a Pentium based computer equipped with Microsoft C ver. 6 and Watcom C/C++ ver. 10.6. The force signal is sent directly from the force sensor to channel 0 of the ADC without any signal conditioning because of its high signal to noise ratio. The signal from the displacement sensor is filtered through an 8-pole, butterworth filter with its corner frequency at 10 kHz and sent to channel 1 of the ADC. It is then sent to the computer's CPU in which the states and outputs are computed. This value is then sent to the DAC

whose output is then relayed to a power amplifier and finally to the serially connected actuators. This scheme is shown in the following figure.

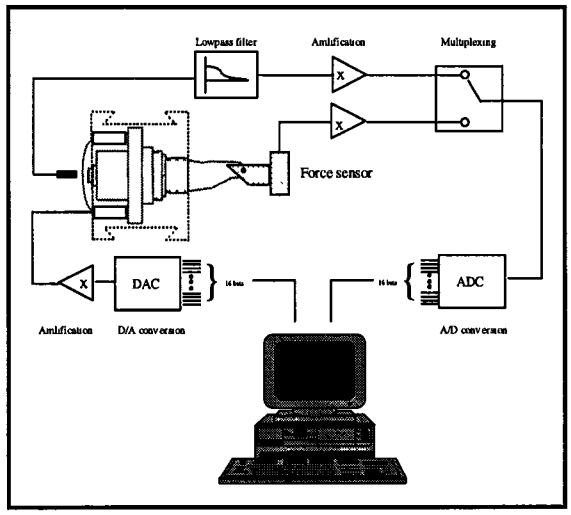


Figure 6-1: Controller setup

The AD/DA conversions are done by the National Instrument's AT-MIO 16X computer board which is capable of scanning 16 channels (8 differential or 16 single ended) analog inputs and also producing 2 analog output signals. This board fits into the EISA expansion slots of the Pentium based computer and it utilizes a 16 bit, 10 microsecond, analog to digital converter (ADC) and a 16 bit digital to analog converter (DAC). The ADC uses successive approximation to convert the voltage signal to a digital value. It can be programmed for uni-polar (0 - 65,535) or bi-polar mode using two's compliment (0 - 32,768 (positive)) and (32,769 - 65535 (negative)) with maximum inputs of 10V. It is guaranteed to convert at a rate at least 100,000 samples per second. The DAC has an output voltage range of 0 to V_{ref}-1 LSB V (bipolar: -V_{ref} to + V_{ref} - 1 LSB V). where 1 LSB = V_{ref} / 2^n-1, for n=16 bits. Using the internal reference source of 10 V, the voltage range becomes 0 to 9.999847 in steps of 152.6 micro-volts. (bipolar: -10 to

9.999695 is steps of 305.2 micro-volts). The bipolar setup corresponds to (-32,768 - 32,767). An external reference can also be utilized.

6.3 Real Time Control

Real time control involves a control approach that is time dependent. This means that the controller reads, writes, and computes information at set instances in time. This is different from simulation in which the success of a computation is not related to when it appears. This is because one time step in a simulation usually is not the same time step in real time. This puts a constraint on the equipment used and also the control algorithm. The system must be able to read data, use the data in the control algorithm, and then produce the output in between sampling periods. Debugging a real-time program is more difficult than the simulation because for simulation, it is easy to reproduce the actions exactly. This is not always possible with real time systems.

6.3.1 Real Time Programming

The control software was written using the C programming language. Two different approaches were attempted which involved 1.) register level programming incorporating an interrupt service routine (ISR) and 2.) using Matlab's Real-Time Workshop. It was found that Matlab's Real Time Workshop offered the greatest benefits. This is because the actual control code and ISR are handled by the software package.

6.3.1.1 Interrupt Service Routines

In a computer, a crystal oscillator generates a sequence of pulses at a constant frequency. Time is kept by counting these "ticks" of the clock. The IBM PC system uses an Intel 8254 chip which is a 16 bit counter/timer that operates at a frequency of 1,193,180 Hz. An unsigned 16 bit integer can keep track of 65535 counts and because of this, the duration is limited to about .055 sec (PC's default interrupt rate). In designing a real time system, it is useful to think of the clock operation and the control program to be independent processes and both running in parallel. Because the computer is a strictly sequential device, this is not actually the case. But if the computer is fast enough, it could seem as though two operations are indeed in parallel. The only requirement is that the control program is completed in between sampling intervals. This pseudo-parallelism is achieved by using interrupts. Interrupts are achieved by utilizing an ISR, counter, and an interrupt controller.

The IBM PC system uses two, 8-bit, Intel 8259 Programmable Interrupt Controllers. Each one has 8 interrupt lines numbered 0 to 7 on the 1st chip and 8 to 15 on the second. The lower interrupts have the highest priority with the timer being the highest. The counter issues an interrupt request by sending a signal to one of the interrupt controller's request input lines. The controller then compares the request's priority with the current process's priority and selects the one with the highest. If the request has a higher priority, it will be sent to the CPU at which time the CPU can honor it or ignore it depending whether or not it is set up to respond to interrupt requests. If the CPU is 'enabled' then the CPU will enter its 'Interrupt Acknowledge Sequence" at which time it

suspends its current operation and handles the new request. The old information, called a vector, is saved so that when the new request is finished, the old operation can resume.

A software package called XIGNAL, developed by Auslander and Tham and modeled after the SIGNAL package on the UNIX system, was used to create the control code [45]. It allows the use of the PC's interrupts without having to deal with the low level coding that is usually necessary. A function called "rsettimer", found in the XIGNAL C-library, allows the interrupt frequency to be altered. This is done by specifying the number of ticks of the counter that are required for the desired sampling frequency. This follows the formula :

$$N_{ticks} = T_{samp.} \cdot 1{,}193{,}180 \qquad (6\text{-}1)$$

where N_{ticks} is the number of ticks and $T_{samp.}$ is the sampling interval in seconds. Therefore, the slowest sampling frequency is about 18 Hz, and the fastest is around .4 MHz. The command "xignal(timer_IRQ,control_program)" installs the function "control_program" which contains the control code as an interrupt routine. This means that every time a pulse is sensed at the "timer_IRQ" interrupt line from the counter, the control program will run. The controller source code is found in appendix B. A diagram of the control algorithm is shown in the following figure:

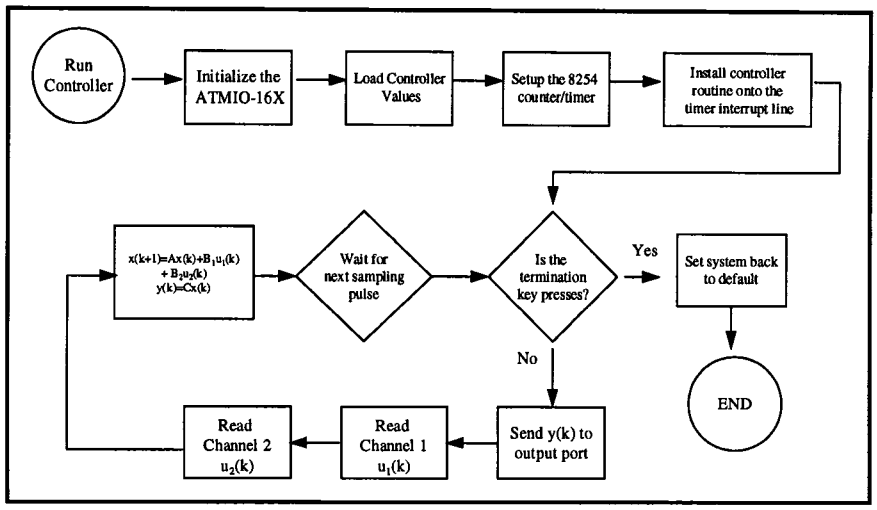


Figure 6-1: Diagram of controller algorithm

The controller is initiated by running a program file called "control.exe". This program first initializes the I/O board and sets the input and output channels to their proper configurations. It then loads the controller values from a file called "control.txt". This allows flexibility during the development phase so that a new controller program does not have to be compiled every time a change has been made to the controller values. A Matlab program called "makelist.m" was written to create the "control.txt" text file that contains the controller values. Next, the counter is set to the correct interrupt rate and the control algorithm is installed onto the timer interrupt line as an interrupt routine. The system then checks to see if any key on the keyboard has been pressed to indicate a desired abort. If not, the system then proceeds with its control. At the frequency specified by the counter, the system will output the present y(k) value to the DAC, read two analog input channels and service the data by multiply by the appropriate controller values.

There is a number of factors that determine the fastest sampling rate possible including the sampling rate of the DAQ board, the control program, and the processor speed. For single conversions, the DAQ board receives a strobe on its SingleConversion register[1]. It then reads one sample value and set the "data ready" bit in the Status 1 register to indicate that the data is ready to be serviced. The register map is included in appendix C. As will be shown, there is a significant lead time associated with collecting data, point by point.

In determining the speed of the control loop, the time duration for the controller code with and without data sampling was conducted. In testing the speed of the controller code without data sampling, a command to send 5V to the DAC was placed immediately before the computation began and also a command to send -5V to the DAC directly after the computation was complete. An oscilloscope was used to analyze the square wave that was produced. This is shown in Figure 6-2 where Δt is the control loop duration.

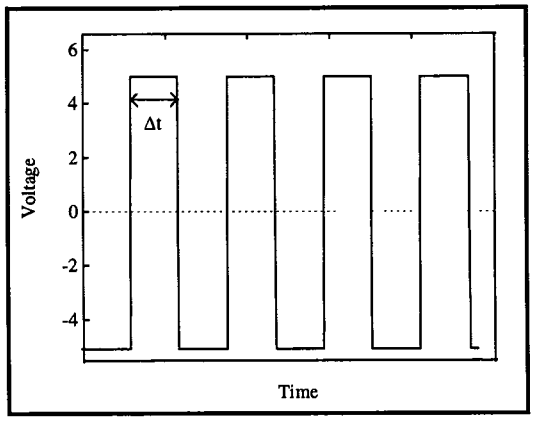


Figure 6-2: Square wave to determine duration of controller algorithm

[1] A strobe refers to accessing a register address either by reading or writing to it.

It was determined that controller calculations took approximately 37μs. A similar method was utilized to determine the duration of the control loop for a single input and dual input scenario. It was found that for a single input, the control loop took 51.6μs. This means that approximately 14.5μs. was needed to collect the read the data. For the dual input scenario, the duration was 63.2μs. Therefore, the maximum control loop that can be achieved with this current program for a dual input/single output controller is 15.8 kHz. This is the reason the controller was designed for a 15 kHz control loop.

6.4 Matlab's Real Time Workshop

Matlab's Real Time Workshop is a toolbox that incorporates graphical programming in producing a real time control program in an executable form. This works by first producing driver blocks that incorporate the working dynamics of the input/output board that is being used. The driver blocks are created by writing the driver information in the C programming language following a template that Matlab software needs to incorporate the two. Once the file is created, a binary "cmex" file is produced. A "cmex" file is a C subroutine that is dynamically linked with Matlab. This file is then used to create a "driver block". Two driver blocks needed to be created: one that incorporated the ADC information and the other that incorporated the DAC information. The source codes for the drivers are found in appendix D.

In order to produce the controller executable, the following steps are needed:

Use the device driver blocks: "mio16xad" and "mio16xda" located in the "atmio_drivers.m" file. Once the discrete time control program has been prepared in

Matlab, attach the AT-MIO-16X ADC to the input and the AT-MIO-16X DAC to the output. This is illustrated in the following figure:

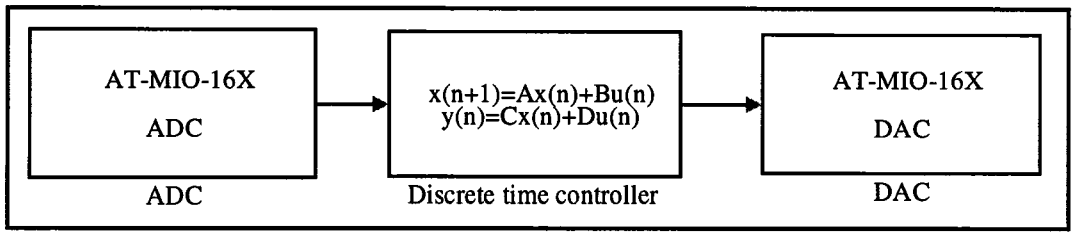


Figure 6-3: Controller Code generation using Matlab's Real Time Workshop

The driver blocks are setup so that various parameters such as number of channels and sampling frequency can be set directly inside the blocks. The sampling frequency that is set in the driver block should be consistent with the sampling frequency used to produce the discrete time model from the continuous time using equation (5-29).

Modifications to the software were necessary to create a successful controller code. This is because the RTW software program was designed for the UNIX platform on a Silicon Graphics workstation (SGI) and the C compiler was written for the Windows 3.1 platform. The modified makefile template is called "mio_watc.tm2" and has reference to another modified file called "mio_main.c". After the building process is finished on the SGI, three text files are produced: "CONTROL.h", "CONTROL.c", and "CONTROL.mk". These three files are downloaded from the SGI to a separate directory on a PC that has the 'code building' Real Time Workshop files and Watcom C/C++ ver. 10.6 compiler. The command "wmake -f CONTROL.mk " is then issued at the c:\ prompt in the DOS shell. After this is done, a control executable, CONTROL.exe appears in the directory. This file is then copied onto the computer with the I/O board. It should be noted that this executable initializes the board each time it is executed. The ADC

calibration register should be strobed before using the board. Since this operation takes approximately 1.25 sec to complete and only has to be done once upon startup of the computer, it is not part of the initialization routine. Instead a separate program called "init_mio.exe" was created. This program should be executed once before implementation begins.

7. CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

The overall objective of this research is to design a strategy for control and implementation of the Smart Toolpost for vibration compensation during a turning process using electrostrictive actuators. This thesis comprises a little over two years worth of effort. Several different approaches with regard to the system modeling, controller designs, and implementation were attempted. Only the methods that gave the best results are presented in this work. Most of the software code is included in the appendix, but for brevity, all of the supporting header files and assembly files are not included. For this reason, all of the files that were mentioned are on a single disk which can be found in the Advanced Design and Manufacturing Lab.

In summarizing this thesis, it can be broken down into three distinct areas: systems modeling, control systems design, and microprocessor implementation.

1. System identification was used to accurately model the toolpost's mechanical structure with three embedded, electrically open actuators. Also, the voltage / tool tip displacement mapping was developed. These two models were then combined and conditioned to produce a 4th order, linear, controllable and observable dual input / single

output, state variable representation of the Smart Toolpost system. This method was considered more appropriate than modeling from 1st principles because it produced models in the form that was conducive to modern linear control algorithms. Results from simulations and testbed measurements show that the actuators have adequate control authority and are excellent candidates to be used for vibration control.

2. The controller design approach was one in which the cutting force was considered to be a random disturbance to the Smart Toolpost system and the overall goal was to minimize the mapping between this disturbance and the tool tip displacement. The poles of the optimal controller and the observer were determined by taking into account the limitations of the present equipment. Since the goal is to attenuate a random disturbance, it would not have been feasible to try and model the disturbance. For this reason, the controller has to receive the disturbance as one of its inputs as well as the output displacement. It was found that a 20 dB attenuation was attainable with the current limitation on the control loop speed.

An adaptive pole placement method that utilized the poles derived from optimal control method was also investigated. It was found that although, a least squares on-line parameter identifier could not be used for control, it was suffice to be used for health monitoring.

3. A vehicle for time efficient controller implementation was also developed. Two methods were mentioned in the thesis. Self coding was one of the options. A program was created that incorporated an interrupt service routine (ISR) that allowed the control routine to be run at a pre-specified sampling frequency using the PC's system clock and

programmable interrupt controller. The other method utilizes Matlab's Real Time Workshop to develop the control code and ISR. This method has the advantage in that it combined the ease of working in an already familiar graphical environment to design, evaluate, and construct the controller. This allows more energy to be used in the actual controller design as opposed to the implementation. The system setup involved developing a driver for the DAQ cards that was able to interact with Matlab. Although for a 4th order, linear, discrete time system, the control code is not very advanced, benefits would be readily observed when trying to implement an adaptive, non-linear, or neural-network type controller.

7.2 Suggestions for future work

1. Testing the controller during actual machining process. Since the controller needs the disturbance force information, a method needs to be determined to measure the cutting forces generated during turning. One idea is to use fiber optic strain gauges, embedded between the cutting tool and the toolpost structure. Another method may be to use PVDF strain gauges surface mounted on the toolpost shaft. Of course, there should be appropriate measures to ensure protection of the sensor.

2. Some method of actuator displacement amplification is needed. Displacement amplification would greatly enhance the capability of the smart toolpost, because it may allow the toolpost to reduce vibration at heavy-duty machining conditions.

3. Experiments with similar analysis using other membranes with different material properties is needed. Results from these experiments may provide further evidence to determine if more control authority can be gained without sacrificing structural integrity.

Appendix A

Derivation of a 3-D Constitutive Model for PMN Material

The following is the derivation of a 3-D constitutive model governing the electromechanical behavior of a relaxor ferroelectric as functions of stress, polarization, and temperature. This derivation comes from the works of Hom et al [24,29].

From the 1st law of thermodynamics:

$$dU = \sigma_{ij} d\varepsilon_{ij} + E_k dD_k + TdS \qquad \text{(A-1)}$$

Where:

U: Material's internal energy per unit volume

σ: Stress tensor

ε: Strain tensor

E: Electric field vector

D: Electric displacement vector

T: Temperature

S: Entropy per unit volume

The strain tensor is defined as:

$$\varepsilon_{ij} = \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{33} \\ \varepsilon_{12} \\ \varepsilon_{23} \\ \varepsilon_{13} \end{bmatrix} = \varepsilon_p = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \end{bmatrix} \quad \text{(A-2)}$$

The electric field **E** and the electronic displacement **D**, are related to the material's polarization **P**, by the equation;

$$D_i = \varepsilon_0 E_i + P_i \quad \text{(A-3)}$$

Where ε_0 is the permitivity of free space.

The Gibbs free energy is defined as

$$G = U - \sigma_{ij}\varepsilon_{ij} - \frac{1}{2}\varepsilon_0 E_k E_k - TS \quad \text{(A-4)}$$

differentiating equation (A-4) yields

$$dG = dU - d\sigma_{ij}\varepsilon_{ij} - \sigma_{ij}d\varepsilon_{ij} - \varepsilon_0 E_k dE_k - SdT - TdS \quad \text{(A-5)}$$

Substituting equation (A-1), into the above equation yields

$$dG = \sigma_{ij}d\varepsilon_{ij} + E_k dD_k + TdS - d\sigma_{ij}\varepsilon_{ij} - \sigma_{ij}d\varepsilon_{ij} - \varepsilon_0 E_k dE_k - SdT - TdS \quad \text{(A-6)}$$

The electric field and the electric displacement are related to the polarization by the following equation.

$$dD_k = \varepsilon_0 dE_k + dP_k \quad \text{(A-7)}$$

Therefore,

$$dG = -\varepsilon_{ij}d\sigma_{ij} - SdT + E_k dP_k \quad \text{(A-8)}$$

The Gibbs free energy equationn also has the following form:

$$dG = \frac{\partial G}{\partial \sigma_{ij}} d\sigma_{ij} + \frac{\partial G}{\partial P_k} dP_k + \frac{\partial G}{\partial T} dT \qquad \text{(A-9)}$$

which shows that

$$\varepsilon_{ij} = -\frac{\partial G}{\partial \sigma_{ij}}; \quad E_k = \frac{\partial G}{\partial P_k}; \quad S = -\frac{\partial G}{\partial T} \qquad \text{(A-10)}$$

a.) b.) c.)

Assuming that the electrically induced strain ε^E depends only on second order polarization terms:

$$\varepsilon_{xy}{}^{E} = Q_{xyij} P_i P_j \qquad \text{(A-11)}$$

Where **Q** are the electrostrictive coefficients.

The total strain of the ceramic is the sum of the elastic strain, the electrically induced strain and the thermally induced strain.

$$\hat{\varepsilon}_{uv} = \varepsilon_{uv} - \varepsilon_{uv}{}^{E} - \alpha_{uv}(T - T_0) \qquad \text{(A-12)}$$

Therefore, the stress in the relaxor ferroelectric is

$$\sigma_{xy} = C_{xyuv} \hat{\varepsilon}_{uv} \qquad \text{(A-13)}$$

where:

T_0: Stress free temperature (generally the burnout temperature)

C: Elastic given by:

$$C = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \upsilon & \upsilon & 0 & 0 & 0 \\ \upsilon & 1-\nu & \upsilon & 0 & 0 & 0 \\ \upsilon & \upsilon & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}_{uv} \qquad \text{(A-14)}$$

where

E: Young's modulus

ν: Poisson's ratio

α: Thermal expansion coefficient vector given by:

$$[\alpha \quad \alpha \quad \alpha \quad 0 \quad 0 \quad 0] \tag{A-15}$$

Integrating the 1st term in equation (A-10), in the following form:

$$\sigma_{xy} = -\frac{\partial G}{\partial \varepsilon_{xy}} \tag{A-16}$$

yeilds:

$$G = -\int C_{xyuv}(\varepsilon_{uv} - \varepsilon_{uv}{}^{E} - \alpha_{uv}(T - T_0))d\varepsilon_{xy} \tag{A-17}$$

using:

$$\varepsilon_{uv}{}^{E} = Q_{uvij} P_i P_j \tag{A-18}$$

$$G = \frac{C_{xyuv}\varepsilon_{xy}\varepsilon_{uv}}{2} - C_{xyuv}Q_{xyij}\varepsilon_{xy}P_i P_j - C_{xyuv}\varepsilon_{xy}\alpha_{uv}(T - T_0) + f(\mathbf{P}) \tag{A-19}$$

Where **f** is a function of polarization only.

The stress free relationship between the polarization and the applied electric field assumes the form of the hyperbolic tangent function:

$$|P| = P_s \tanh(k|E|) \tag{A-20}$$

where $\mathbf{P_s}$ is the spontaneous polarization and $\boldsymbol{k}$ is the new material constant. Both $\mathbf{P_s}$ and $\boldsymbol{k}$ are temperature dependent and their values are determined experimentally.

Assuming that the material is isotropic, equation (A-20), can be rewritten as

$$E_i = \frac{1}{k}\arctan h(\frac{|\mathbf{P}|}{P_s})\frac{P_i}{|\mathbf{P}|} \quad \text{(A-21)}$$

Remembering that:

$$E_i = \frac{\partial G}{\partial P_i} \quad \text{(A-22)}$$

then differentiaitng equation (A-19) yields:

$$E_i = -2C_{xyuv}Q_{xyij}P_j\varepsilon_{xy} + \frac{\partial f}{\partial P_i} \quad \text{(A-23)}$$

Therefore, using equations (A-12) and (A-23) yields:

$$\frac{\partial f}{\partial P_i} = 2C_{xyuv}Q_{uvij}P_j(Q_{xykl}P_kP_l + \alpha_{xy}(T-T_0)) + \frac{1}{k}\arctan h(\frac{|\mathbf{P}|}{P_s})\frac{P_i}{|\mathbf{P}|} \quad \text{(A-24)}$$

where, because σ_{xy}=0:

$$\varepsilon_{xy} = Q_{xykl}P_kP_l + \alpha_{xy}(T-T_0) \quad \text{(A-25)}$$

Integrating equation (A-24) and substituting $f(p)$ into equation (A-19), yields:

$$G = \frac{C_{xyuv}}{2}\left[\varepsilon_{uv} - \varepsilon_{uv}^E - \alpha_q(T-T_0)\right]\left[\varepsilon_{xy} - \varepsilon_{xy}^E - \alpha_{xy}(T-T_0)\right] + \frac{1}{2k}\left[|P|\ln\left(\frac{P_s+|P|}{P_s-|P|}\right) + P_s\ln\left(1-\left(\frac{|P|}{P_s}\right)^2\right)\right] \quad \text{(A-26)}$$

Therefore, using equation (A-10) yields:

$$E_i = -2C_{xyuv}Q_{xyij}(\varepsilon_{uv} - Q_{uvkl}P_kP_l - \alpha_{uv}(T-T_0))P_j + \frac{1}{k}\arctan h(\frac{|\mathbf{P}|}{P_s})\frac{P_i}{|\mathbf{P}|} \quad \text{(A-27)}$$

Assuming isothermal conditions, equation (A-27) can be reduced to:

$$E_i = -2Q_{xyij}\sigma_{xy}P_j + \frac{1}{k}\arctan h(\frac{|\mathbf{P}|}{P_s})\frac{P_i}{|\mathbf{P}|} \quad \text{(A-28)}$$

Using equation (A-10) yields:

$$\varepsilon_{ij} = s_{ijkl}{}^{P}\sigma_{xy} + Q_{ijmn}P_mP_n \qquad \text{(A-29)}$$

Where s = 1/E.

Equations (A-28) and (A-29) together form the 3 dimensional constitutive model that define the electromechanical behavior of a relaxor ferroelectric. The first term on the right side of equation (A-28) is referred to as the converse electrostrictive effect and the second term is referred to as the direct electrostrictive effect.

For isotropic behavior, the non-zero elements of **Q** are related by:

$$Q_{111}=Q_{222}=Q_{333}$$

$$Q_{122}=Q_{133}=Q_{211}=Q_{233}=Q_{311}=Q_{322} \qquad \text{(A-30)}$$

$$Q_{412}=Q_{523}=Q_{613}=2(Q_{111}-Q_{122})$$

Therefore, there are only two independent components, Q_{111} and Q_{122}.

Appendix B

Controller C Code

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

FILE
        control.c  -  real time controller

SYNOPSIS
        uses the x
  #include "xignal.h"

compile with:   cl /Gs control.c /link xignal  using Microsoft C version 6.


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

/*************************************************************************
                                                                IMPORTS
*************************************************************************/

#include <stdio.h>
#include "envir.h"
#include "xignal.h"
#include "alarm.c"
#include <conio.h> /* inp(), outp() etc.. */
#include <time.h>  /* clock */
#include "atmio16x.h"  /* ATMIO-16X Register map */
#include <io.h>
#include <stdlib.h>
#include <math.h>
#include <memory.h>

#define XIGTMR        1
#define BITS_TO_VOLTS   0.0003051804379339
#define num_inputs                  1
#define order                       4

/*************************************************************************
                                                        PRIVATE  DATA
*************************************************************************/
        float   A[order][order];        /* dynamics matrix */
```

```
        float  B[order][num_inputs];    /* input matrix */
        float  C[order];              /* output matrix */
        float  xold[order]={0};       /* state variables */
        int    arraysize=order;       /* model order */
        float  y = 0; /* output of C_signal */
        int long tick_count=119;      /* interrupt freq. of about 10,000 Hz. */
        float  force;
        float  displacement;
        int                  tick;

/*************************************************************************
        Function prototypes
*************************************************************************/

        short initialize (void);
        short initialize_Am9513A (void);
        short clear_ai (void);
        short disable_rtsi (void);
        void  load_variables(float [][order],float [][num_inputs],float []);
        void  C_Signal(void);
        short setup_Input (void);
        short setup_Output (void);

main()
{
                initialize ();
                clear_ai ();
                setup_Input();
                setup_Output();
                load_variables(A,B,C);   /* Variable are in a file called test2.txt*/
                rsetalarm(tick_count);

                xignal(XIGTMR, C_Signal);   /* interrupt CPU and run controller

        at the set sampling frequency  */


                                  while (!kbhit());      /* wait until keyboard is hit to abort */
        xignal(XIGTMR, XIG_DFL);      /* restore timer interrupts */
        outpw(DAC0,0);                                          /* Turn off control */
}

short initialize (void)

        {
                outpw (CMD1, 0x0000);
                outpw (CMD2, 0x0000);
                outpw (CMD3, 0x0000);
                outpw (CMD4, 0x0000);

                inp (DAQClear);
                outp (DMATCAClear, 0x00);
                inp (DMATCBClear);
                inp (DMAChannelClear);
                inp (DACClear);
                inp (TMRREQClear);
                /* outp (ADCCalibration, 0x00);  Initialize after bootup instead

               using init_mio.exe */
                initialize_Am9513A ();
```

```
                disable_rtsi ();

                return 0;
        }

short initialize_Am9513A (void)

        {
                short ctr;

                outpw (Am9513ACommand, 0xFFFF);
                outpw (Am9513ACommand, 0xFFEF);
                outpw (Am9513ACommand, 0xFF17);
                outpw (Am9513AData, 0xF000);
                for (ctr=1; ctr<6; ctr++) {
                                        outpw (Am9513ACommand, 0xFF00+ctr);
                                        outpw (Am9513AData, 0x0004);
                                        outpw (Am9513ACommand, 0xFF08+ctr);
                                        outpw (Am9513AData, 0x0003);
                }
                outpw (Am9513ACommand, 0xFF5F);


                                return 0;
              }

short disable_rtsi (void)

        {
                short i;

                for (i=0; i<56; i++)
                                        outp (RTSISwitchShift, 0x00);
                outp (RTSISwitchStrobe, 0x00); /* load the 56-bit pattern into the
        RTSI switch */

                return 0;
        }

short clear_ai (void)

        {
                inp (DAQClear);

                return 0;
        }


void C_Signal()
{
        int temp_force,temp_disp;

        /*******************************************************
         This function computes the feedback control signal.  The
         inputs are force and displacement.
            /
         Don't forget the function prototype
         void C-Signal(float [][],float [] ,float [], float [],float, int)

         Also: the output y is external
```

```
*/


int row;
int column;
float xnew[order]={0};

outpw (DAC0, (int) y/BITS_TO_VOLTS);                    /* send y(k) */

y=0;

outp (SingleConversion,0);   /* initiate single A/D  */

while ( (4096 & inpw (STATUS1) ) != 4096); /* wait for data to be ready */

temp_force = inpw(ADCFIFO);
        if (temp_force > 32768l)
         {
                temp_force=(temp_force-65536l);
         }
 force= (float) temp_force*BITS_TO_VOLTS; /* Convert u1(k) to volts */

if(num_inputs>1)

{
        outp (SingleConversion,0);   /* initiate single A/D  */
        while ( (4096 & inpw (STATUS1) ) != 4096 ); /* wait for data to be ready */

        temp_disp = inpw(ADCFIFO);
        if (temp_disp > 32768l)
                {
                        temp_disp=(temp_disp-65536l);
                }
        displacement=temp_disp*BITS_TO_VOLTS; /* Convert u2(k) to volts */
}



        /* COMPUTATION OF CONTROLLED OUTPUT*/

        for (row=0; row<arraysize; row++ )
        {
                for(column=0; column<arraysize; column++)
                {
                        xnew[row]=xnew[row]+A[row][column]*xold[column];
                }
                xnew[row]=xnew[row]+B[row][0]*force+B[row][1]*displacement; /*x(k+1)*/
        }

/* COMPUTE OUTPUT */


    for(column=0; column<arraysize; column++)

{
        y=y+C[column]*xold[column];      /* y(k) */

    }
        memcpy(&xold[0],xnew,sizeof(xnew));
```

```
}

void load_variables(float A[][order],float B[][num_inputs],float C[])

        {

        /* filename: load_var.c

                This program load the discrete time parameters of the
                state variable representation of the controller algorithm.
                and computes:


                 The input file is organized as follows.
                 1st nxn values => elements of A  (arraysize^2)
                 next n values => elements of B(1)
                 next n values => elements of B(2)
                 last n values => elements of C          */


#define FNAME "\\control.txt"



        FILE *fp;
        int    row;
        int    column;

        float word;

        /****** LOAD THE ELEMENTS INTO THEIR PROPER PLACES*/

        fp = fopen(FNAME, "r");

        if (fp == NULL)
        {
                printf("Error opening file %s\n", FNAME);
                exit(1);
        }

        /* EXTRACT THE VALUES OF THE A MATRIX */

        for (column=0;column<arraysize;column++)
        {
                for (row=0;row<arraysize;row++)
                {
                 fscanf(fp,"%f",&word);
                 A[row][column]=word;

                }
        }

        /* EXTRACT THE VALUES OF B */

        for (column=0;column<num_inputs;column++)

        {
                for (row=0;row<arraysize;row++)
```

```
                {
                fscanf(fp,"%f",&word);
                B[row][column]=word;

                }
        }


        /* EXTRACT THE VALUES OF C */

        for (column=0;column<arraysize;column++)
        {
                fscanf(fp,"%f",&word);

                C[column]=word;
        }


        /* clean up */
        fclose(fp);
}

short setup_Input (void)
{
int i;
int VALUE;
int CHAN_FLAGG=0;
printf("\n\nSystem is setup to read %d input channels\n",num_inputs);
if (num_inputs==2)
{
printf("First channel==> Force   Second channel ==> displacement");
}

/* Configure scanning sequence memory */
        for (i=0;i<num_inputs;i++)
        {

        if (i==num_inputs-1) CHAN_FLAGG=1;

        /* All channels are set for differential, bipolar, save in  FIFO, no DSP */
        switch(i)
                {
                case 0:
                VALUE=4104;
                if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                outpw(CONFIGMEM, VALUE);/* Sets channel 0 */
                break;
                case 1:
                VALUE=4168;
                if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                outpw(CONFIGMEM, VALUE);/* Sets channel 1 */
                break;
                case 2:
                VALUE=4232;
                if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                outpw(CONFIGMEM, VALUE);/* Sets channel 2 */
                break;
                case 3:
                VALUE=4296;
                if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
```

```
                    outpw(CONFIGMEM, VALUE);/* Sets channel 3 */
                    break;
                    case 4:
                    VALUE=4360;
                    if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                    outpw(CONFIGMEM, VALUE);/* Sets channel 4 */
                    break;
                    case 5:
                    VALUE=4424;
                    if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                    outpw(CONFIGMEM, VALUE);/* Sets channel 5 */
                    break;
                    case 6:
                    VALUE=4488;
                    if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                    outpw(CONFIGMEM, VALUE);/* Sets channel 6 */
                    break;
                    case 7:
                    VALUE=4552;
                    if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                    outpw(CONFIGMEM, VALUE);/* Sets channel 7 */
                    break;
                    case 8:
                    VALUE=4616;
                    if (CHAN_FLAGG=1) VALUE = 0x04|VALUE;       /* sets last channel bit */
                    outpw(CONFIGMEM, VALUE);/* Sets channel 8 */
                    break;
                    }
          }
          outp(CONFIGMEMLD, 0x00); /* Loads channel configurations */
          printf("\n\nInput setup is complete!\n",0);
return(0);
}

short setup_Output (void)
{
          outpw (CMD2, 0xC01);                  /* sets DAC0 and DAC1 of bipolar operation
                                                Primary DMA channel 0,secondary DMA channel 1 */
          printf("\n\nOutput setup is complete!\n",0);
return(0);
}
```

Appendix C

Register Map for National Instruments' AT-MIO 16X Data Acquisition Board

Register Name	Offset (Hex)	Type	Size
ADC Event Strobe Register Group			
CONFIGMEMCLR Register	1B	Read-only	8-bit
CONFIGMEMLD Register	1B	Write-only	8-bit
DAQ Clear Register	19	Read-only	8-bit
DAQ Start Register	1D	Read-only	8-bit
Single Conversion Register	1D	Write-only	8-bit
ADC Calibration Register	1F	Write-only	8-bit
DAC Event Strobe Register Group			
TMREQ Clear Register	1F	Read-only	8-bit
DAQ Update Register	18	Write-only	16-bit
DAC Clear Register	1E	Read-only	8-bit
General Event Strobe Resgister Group			
DMA Channel Clear Tegister	0B	Read-only	8-bit
DMATCA Clear Register	19	Write-only	8-bit
DMACB Clear Register	09	Read-only	8-bit
External Stobe Register	1E	Write-only	8-bit
Calibration DAC 0 Load Register	0A	Write-only	8-bit
Calibration DAC 1 Load Register	1A	Write-only	8-bit
Am9513A Data Register	14	Write-only	16-bit
Am9513A Command Register	16	Read-only	16-bit
Am9513A Status Register	16	Read-only	16-bit
Digital I/O Register Group		Read-only	
Digital Input Register	1C	Write-only	16-bit
Digital Output Register	1C	Write-only	16-bit
RTSI Switch Register Group		Write-only	
RTSI Switch Shif Register	0C	Write-only	8-bit
RTSI Switch Strobe Register	0E	Write-only	8-bit
Configuration and Status Register Group			
Command Register 1	0	Write-only	16-bit
Command Register 2	2	Write-only	16-bit
Command Register 3	4	Write-only	16-bit
Command Register 4	6	Read-only	16-bit
Status Register 1	18	Read-only	16-bit
Status Register 2	1A	Read-only	16-bit
Analog Input Register Group		Read-only	
ADC FIFO Register	0	Write-only	16-bit
CONFIGMEM Register	8	Write-only	16-bit
Analog Output Registe Group		Write-only	
DAC 0 Register	10	Write-only	16-bit
DAC 1 Register	12	Write-only	16-bit

Table C-1: Register Map for National Instrument's AT-MIO 16X I/O computer board

Appendix D

C Code for the AT-MIO 16X Driver Blocks

Analog to Digital Converter driver block

```
/*
 * mioad.c
 *
 *  S-Function device driver for the National Instruments
 *  AT-MIO-16X analog input section of the board.
 *          Designed for use with Watcom C/C++ ver. 10.6
 *
 * Copyright (c) 1993-1995 The MathWorks, Inc.
 * All Rights Reserved
 * Modified by Adrian Hood on 8/18/96
 */

#define S_FUNCTION_NAME mioad

#include <stddef.h> /* For NULL */
#include <stdlib.h> /* For min */
#include <math.h>  /* For pow() */
#include <stdio.h>
#include <conio.h> /* for outp,outpw,inp,inpw */
#include "simstruc.h" /* Where simulation structure, S, is defined */
#include "atmio16x.h" /* Register Map without initialization functions */

#ifdef MATLAB_MEX_FILE
#include "mex.h"
#endif

#ifndef min
#define min(x, y)      ((x) < (y) ? (x) : (y))
#endif


/* Input Arguments */
#define BASE_ADDRESS_ARG            ssGetArg(S,0)
#define LOW_HIGH_GAIN_ARG           ssGetArg(S,1)
```

```
#define NUM_CHANNELS_BITS_ARG                    ssGetArg(S,2)
#define SAMPLE_TIME_ARG          ssGetArg(S,3)
#define ACCESS_HW_ARG            ssGetArg(S,4)
#define NUMBER_OF_ARGS           (5)

/* Indices into vectored input arguments */
#define LOW_INPUT            (0)
#define HIGH_INPUT           (1)
#define GAIN             (2)
#define LOW_HIGH_GAIN                    (3)
#define CHANNELS             (0)
#define NUM_CHANNELS_BITS                (1)
#define NSAMPLE_TIMES                (1)

/* Storage Allocation */
#define MAX_INPUTS           (16)
#define BASE_ADDRESS                     (16)
#define NUM_CHANNELS                     (17)
#define ACCESS_HW            (18)
#define ADC_ZERO             (19)

/*
 * Integer storage is used for the number of effective
 * bits for each channel plus the above necessary storage.
 * Each input channel's programmable resolution
 * is stored in a real work vector.
 */
#define NUMBER_OF_IWORKS     (MAX_INPUTS + 4)
#define NUMBER_OF_RWORKS     (MAX_INPUTS)

/* hardware registers */
#define START_CONV                   ((base_addr) + 0x1D)
#define ADC_WORD                         ((base_addr) + 0x00)
#define STATUS(base_addr)    ((base_addr) + 0x1A)

/* Status register bit definitions */
#define ADC_BUSY         (0x1000) /* mask for "data ready" bit 12 */
#define ADC_MAX_RESOLUTION   (65536)
#define ADC_MAX_BITS         (16)
#define BIT_DIFFERENCE       (0)   /* Check on this value */
#define ADC_BIPOLAR_ZERO     (0x0800)/* Check on this value */
#define ADC_UNIPOLAR_ZERO    (0)   /* Check on this value */


static void mdlInitializeSizes(S)
        SimStruct *S;
{
        int num_channels;

        if (ssGetNumArgs(S) != NUMBER_OF_ARGS)
        {
#ifdef MATLAB_MEX_FILE
                mexErrMsgTxt("Wrong number of input arguments passed.\nFive arguments are expected\n");
#endif
        }
        /* Check the size of the number of channels,
         * number of bits per channel input.
         */
        else if (mxGetN(NUM_CHANNELS_BITS_ARG) < NUM_CHANNELS_BITS)
        {
```

```
#ifdef MATLAB_MEX_FILE
                mexErrMsgTxt("The size of the Number of Channels, Bits Per Channel vector must be at least
1\n");
#endif
        }
        else
        {
                num_channels = mxGetPr(NUM_CHANNELS_BITS_ARG)[CHANNELS];
#ifdef MATLAB_MEX_FILE
                if ((num_channels < 1) || (num_channels > MAX_INPUTS)) {
                        mexErrMsgTxt("Number of input channels must be between 1 and 16\n");
                }
#endif
                /* Set-up size information */
                ssSetNumContStates(S, 0);
                ssSetNumDiscStates(S, 0);
                ssSetNumOutputs(S, num_channels);
                ssSetNumInputs(S, num_channels);
                ssSetDirectFeedThrough(S, 0); /* No direct feed-through */
                ssSetNumSampleTimes(S, NSAMPLE_TIMES);
                ssSetNumInputArgs(S, NUMBER_OF_ARGS);
                ssSetNumIWork(S, NUMBER_OF_IWORKS); /* A/D card hardware register pointers */
                ssSetNumRWork(S, NUMBER_OF_RWORKS); /* Programmable gain and A/D LSB resolution
*/
        }
}

/* Function to initialize sample times */
static void mdlInitializeSampleTimes(S)
        SimStruct *S;
{
        ssSetSampleTimeEvent(S, 0, mxGetPr(SAMPLE_TIME_ARG)[0]);
        ssSetOffsetTimeEvent(S, 0, 0);
}

static void mdlInitializeConditions(x0, S)
        double *x0;
        SimStruct *S;
{
        int arg_str_len = 128;
        char arg_str[128];
        unsigned int base_addr;
        int num_channels;
        int num_bits;
        double prog_gain;
        double volts_per_bit;
        unsigned int num_inputs;
        int access_hw;
        int adc_resolution[MAX_INPUTS];
        int num_adcs;
        double low_input;
        double high_input;
        int i;


        mxGetString(BASE_ADDRESS_ARG, arg_str, arg_str_len);
        base_addr = (unsigned long)strtol(arg_str, NULL, 0);
        ssSetIWorkValue(S, BASE_ADDRESS, (int)base_addr);

        num_channels = min(MAX_INPUTS, (int)mxGetPr(NUM_CHANNELS_BITS_ARG)[CHANNELS]);
```

```
        ssSetIWorkValue(S, NUM_CHANNELS, num_channels);

        /* Check the size of the low, high, gain input */
        if (mxGetN(LOW_HIGH_GAIN_ARG) != LOW_HIGH_GAIN){
#ifdef MATLAB_MEX_FILE
                mexErrMsgTxt("The size of the Low Input, High Input, Gain vector must be 3\n");
#endif
        }else{
                /* Get the input voltage range */
                low_input = (double)mxGetPr(LOW_HIGH_GAIN_ARG)[LOW_INPUT];
                high_input = (double)mxGetPr(LOW_HIGH_GAIN_ARG)[HIGH_INPUT];
        }

        /* Set up the ADC zero value for the different
         * input voltage ranges.
         */
        if (low_input == 0. &&
                0.5 <= high_input && high_input <= 10.){
                ssSetIWorkValue(S, ADC_ZERO, ADC_UNIPOLAR_ZERO);
        } else if (( -10. <=low_input && low_input <= -.25) &&
                        (.25 <= high_input && high_input <= 10.0)){
                ssSetIWorkValue(S, ADC_ZERO, ADC_BIPOLAR_ZERO);
        }
#ifdef MATLAB_MEX_FILE
         else{
                mexErrMsgTxt("Input voltage range must be between -10.0V and 10.0V\n");
        }
#endif

        /* Initialize the ADC resolution and number
         * of bits to their respective maximums.
         */
        for (i = 0; i < num_channels; i++){
                adc_resolution[i] = ADC_MAX_RESOLUTION;
                ssSetIWorkValue(S, i, BIT_DIFFERENCE);
        }

        num_adcs = min(num_channels, (int)mxGetN(NUM_CHANNELS_BITS_ARG) - 1);

        /* Now save the bit difference and the new resolution
         * to integer storage space and real storage space,
         * respectively.
         */
        if (num_adcs > 0){
                for (i = 0; i < num_adcs; i++){
                        num_bits = (int)mxGetPr(NUM_CHANNELS_BITS_ARG)[i + 1];
#ifdef MATLAB_MEX_FILE
                        if ((num_bits < 1) || (num_bits > ADC_MAX_BITS)) {
                                mexErrMsgTxt("The number of bits must be between 1 and 16\n");
                        }
#endif
                        adc_resolution[i] = (int)pow(2.0, (double)num_bits);
                        ssSetIWorkValue(S, i, ADC_MAX_BITS - num_bits);
                }
        }

        prog_gain = (double)mxGetPr(LOW_HIGH_GAIN_ARG)[GAIN];
#ifdef MATLAB_MEX_FILE
        if ((prog_gain < 0.5) || (prog_gain > 100.0)) {
                mexErrMsgTxt("The input gain must be between 0.5 and 100\n");
```

```
                /* Note: allowable gains are 1,2,5,10,20,50,and 100 */
        }
#endif

        for (i = 0; i < num_channels; i++){
                volts_per_bit = 10.0 / (prog_gain * (double)adc_resolution[i]);
                ssSetRWorkValue(S, i, volts_per_bit);
        }

        access_hw = (int)(mxGetPr(ACCESS_HW_ARG)[0]);
        ssSetIWorkValue(S, ACCESS_HW, access_hw);

        /* If this is a MEX-file, then accessing the hardware
         * depends on the value of the access hardware flag in the
         * S-function's dialog box.
         */
#ifdef MATLAB_MEX_FILE
        if (!access_hw)
                return;
#endif  /* MATLAB_MEX_FILE */

        /* Set up the board to read the A/D,
         * starting at channel 0 and ending at the
         * number of input channels minus one.

        num_inputs = num_channels - 1;
        hw_outportb(CHANNEL_MUX(base_addr), num_inputs << 4); Need documentation
        Note: the above two lines have been commented out because I don't beleive
        they are needed.  Instead I wrote to  the CONFIGMEM register to do the
        configuration*/
}

/* Function to compute outputs */
static void mdlOutputs(y, x, u, S, tid)
        double *y;
        double *x;
        double *u;
        SimStruct *S;
        int tid;
{
        unsigned int base_addr = ssGetIWorkValue(S, BASE_ADDRESS);
        int num_channels = ssGetIWorkValue(S, NUM_CHANNELS);
        int adc_zero;
        int i;

        /* If this is a MEX-file, then accessing the hardware
         * depends on the value of the access hardware flag in the
         * S-function's dialog box.
         */
#ifdef MATLAB_MEX_FILE
        int access_hw = ssGetIWorkValue(S, ACCESS_HW);
        if (!access_hw)
        {
                /* Allow the inputs to be directly passed to the
                 * outputs for a loopback situtation.  Note that
                 * to use this capability, an input port needs to
                 * be added to the block.
                 */
                for (i = 0; i < num_channels; i++) {
                        y[i] = u[i];
```

```
                }
                return;
        }
#endif /* MATLAB_MEX_FILE */

        adc_zero = ssGetIWorkValue(S, ADC_ZERO);

        for (i = 0; i < num_channels; i++)
        {
                double volts_per_bit;
                unsigned char ADLow, ADHigh;
                long int yout;
                float ADValue;


                outp(START_CONV,0x00);  /* Start single conversion */
/*              volts_per_bit = ssGetRWorkValue(S, i); */
                volts_per_bit = 0.0003051804379339; /* for bipolar mode(20V/65535 bits)*/
                while ((inpw(STATUS1) & ADC_BUSY)!=4096);/* Wait for the A/D to finish */
                yout = inpw(ADCFIFO);   /* FIFO buffer */

                /* 0-32768 ==> positive  65535-32769 ==> negative */
                if (yout > 32768l)
                {
                        yout=(yout-65536l);
                }


                /* Convert value to voltage */
                y[i] = (double) yout*volts_per_bit;

        }
}

/* Function to compute model update */
static void mdlUpdate(x, u, S, tid)
        double *x;
        double *u;
        SimStruct *S;
        int tid;
{
}

/* Function to compute derivatives */
static void mdlDerivatives(dx, x, u, S, tid)
        double *dx;
        double *x;
        double *u;
        SimStruct *S;
        int tid;
{
}

/* Function to perform housekeeping at execution termination */
static void mdlTerminate(S)
        SimStruct *S;
{
}
```

```
#ifdef MATLAB_MEX_FILE  /* Is this file being compiled as a MEX-file? */
#include "simulink.c"   /* Mex glue */
#else
#include "cg_sfun.h"    /* Code generation glue */
#endif
```

Digital to Analog Converter driver block

```
/*
 * mioda.c
 *
 *  S-Function device driver for the National Instruments
 *  AT-MIO-16X analog output section of the board.
 *  Designed for use with Watcom C/C++ ver. 10.6
 *
 * Copyright (c) 1993 by The MathWorks, Inc.
 * All Rights Reserved
 * Modified by Adrian Hood on 6-30-96
 */

#undef S_FUNCTION_NAME
#define S_FUNCTION_NAME         mioda

#include <stddef.h> /* For NULL  */
#include <stdlib.h> /* For min   */
#include <math.h>   /* For pow() */
#include <conio.h>
#include <stdio.h>
#include "simstruc.h" /* Where simulation structure, S, is defined */
#include "atmio16x.h" /*Register Map without intitialization functions*/


#ifdef MATLAB_MEX_FILE
#include "mex.h"
#endif

#ifndef min
#define min(x, y)   ((x) < (y) ? (x) : (y))
#endif

/* compiler dependant low level hardware calls */



/* Input Arguments */
#define BASE_ADDRESS_ARG                ssGetArg(S,0)
#define OUTPUT_REFERENCE_ARG            ssGetArg(S,1)
#define NUM_CHANNELS_BITS_ARG           ssGetArg(S,2)
#define SAMPLE_TIME_ARG                         ssGetArg(S,3)
#define ACCESS_HW_ARG                   ssGetArg(S,4)
#define NUMBER_OF_ARGS                          (5)

/* Indeces into vectored input arguments */
#define CHANNELS                        (0)
#define NUM_CHANNELS_BITS       (1)

#define NSAMPLE_TIMES           (1)

/* Storage Allocation */
/* Each output channel's resolution and number of bits
 * is stored in an integer work vector.  Then add
```

```
 * the other storage space needed.
 */
#define MAX_OUTPUTS                     (2)
#define BASE_ADDRESS                    (4)
#define NUM_CHANNELS                    (5)
#define ACCESS_HW                               (6)
#define NUMBER_OF_IWORKS        ((MAX_OUTPUTS * 2) + 3)

#define OUTPUT_REFERENCE        (0)
#define NUMBER_OF_RWORKS        (1)


#define DAC_ZERO                                (0)
#define DAC_MAX_VOLTAGE                 (32767)
#define DAC_MIN_VOLTAGE                 (-32768)
#define DAC_MAX_BITS                    (16)
#define BIT_DIFFERENCE          (0)  /* Need to determine what this is! */


static void mdlInitializeSizes(S)
        SimStruct *S;
{
        int num_channels;

        if (ssGetNumArgs(S) != NUMBER_OF_ARGS) {
#ifdef MATLAB_MEX_FILE
                mexErrMsgTxt("Wrong number of input arguments passed.\nFive arguments are expected\n");
#endif
        }
        /* Check the size of the number of channels,
         * number of bits per channel input.
         */
        else if (mxGetN(NUM_CHANNELS_BITS_ARG) < NUM_CHANNELS_BITS)
        {
#ifdef MATLAB_MEX_FILE
                mexErrMsgTxt("The size of the Number of Channels, Bits Per Channel vector must be at least
1\n");
#endif
        } else {
                num_channels = mxGetPr(NUM_CHANNELS_BITS_ARG)[CHANNELS];

#ifdef MATLAB_MEX_FILE
                if ((num_channels < 1) || (num_channels > MAX_OUTPUTS)) {
                        mexErrMsgTxt("Number of output channels must be between 1 and 2\n");
                }
#endif
                /* Set-up size information */
                ssSetNumContStates(S, 0);
                ssSetNumDiscStates(S, 0);
                ssSetNumOutputs(S, num_channels);
                ssSetNumInputs(S, num_channels);
                ssSetDirectFeedThrough(S, 1); /* Direct dependency on inputs */
                ssSetNumSampleTimes(S, NSAMPLE_TIMES);
                ssSetNumInputArgs(S, NUMBER_OF_ARGS);
                ssSetNumIWork(S, NUMBER_OF_IWORKS); /* D/A card hardware register pointers */
                ssSetNumRWork(S, NUMBER_OF_RWORKS); /* D/A card output reference voltage */
        }
}

/* Function to initialize sample times */
```

```
static void mdlInitializeSampleTimes(S)
        SimStruct *S;
{
        ssSetSampleTimeEvent(S, 0, mxGetPr(SAMPLE_TIME_ARG)[0]);
        ssSetOffsetTimeEvent(S, 0, 0);
}

static void mdlInitializeConditions(x0, S)
        double *x0;
        SimStruct *S;
{
        int arg_str_len = 128;
        char arg_str[128];
        unsigned int base_addr;
        int num_channels;
        double output_reference;
        int access_hw;
        int num_bits;
        int dac_resolution[MAX_OUTPUTS];
        int num_dacs;
        int i;

        mxGetString(BASE_ADDRESS_ARG, arg_str, arg_str_len);
        base_addr = (unsigned long)strtol(arg_str, NULL, 0);
        ssSetIWorkValue(S, BASE_ADDRESS, (int)base_addr);

        num_channels = min(MAX_OUTPUTS, (int)mxGetPr(NUM_CHANNELS_BITS_ARG)[CHANNELS]);

        printf("Number of output channels: %d\n",num_channels);

        ssSetIWorkValue(S, NUM_CHANNELS, num_channels);

        /* Initialize the ADC resolution and number
         * of bits to their respective maximums.
         */
        for (i=0; i < num_channels; i++)
        {
                ssSetIWorkValue(S, i, DAC_MAX_VOLTAGE);
                ssSetIWorkValue(S, i + MAX_OUTPUTS, BIT_DIFFERENCE);
        }

        num_dacs = min(num_channels, (int)mxGetN(NUM_CHANNELS_BITS_ARG) - 1);

        /* Now save the bit difference and the new resolution
         * to integer storage space.
         */
        if (num_dacs > 0)
        {
                for (i=0; i < num_dacs; i++)
                {
                        num_bits = (int)mxGetPr(NUM_CHANNELS_BITS_ARG)[i + 1];

#ifdef MATLAB_MEX_FILE
                        if ((num_bits < 1) || (num_bits > DAC_MAX_BITS)) {
                                mexErrMsgTxt("The number of bits must be between 1 and 16\n");
                        }
#endif
                        dac_resolution[i] = (int)pow(2.0, (double)num_bits) - 1;
                        ssSetIWorkValue(S, i, dac_resolution[i]);
                        ssSetIWorkValue(S, i + MAX_OUTPUTS, DAC_MAX_BITS - num_bits);
```

```
                }
        }

        output_reference = mxGetPr(OUTPUT_REFERENCE_ARG)[0];
#ifdef MATLAB_MEX_FILE
        if ((output_reference < -10.0) || (output_reference > 10.0)) {
                mexErrMsgTxt("The output reference voltage must be between -10.0 and 10.0 volts\n");
        }
#endif
        ssSetRWorkValue(S, OUTPUT_REFERENCE, output_reference);

        access_hw = (int)(mxGetPr(ACCESS_HW_ARG)[0]);
        ssSetIWorkValue(S, ACCESS_HW, access_hw);

        /* If this is a MEX-file, then accessing the hardware
         * depends on the value of the access hardware flag in the
         * S-function's dialog box.
         */
#ifdef MATLAB_MEX_FILE
        if (!access_hw)
                return;
#endif  /* MATLAB_MEX_FILE */

        /*
         * Initialize the board outputs to be zero volts (0).
         */
         outpw(DAC0,0);
         outpw(DAC1,0);
}

/* Function to compute outputs */
static void mdlOutputs(y, x, u, S, tid)
        double *y;
        double *x;
        double *u;
        SimStruct *S;
        int tid;

{
        /* This function writes a 16 bit value to the DAC on board.*/

        unsigned int base_addr = ssGetIWorkValue(S, BASE_ADDRESS);
        int num_channels = ssGetIWorkValue(S, NUM_CHANNELS);
        double output_reference = ssGetRWorkValue(S, OUTPUT_REFERENCE);
        int i;
        float value;
        int dac_resolution;
        /* If this is a MEX-file, then accessing the hardware
         * depends on the value of the access hardware flag in the
         * S-function's dialog box.
         */
#ifdef MATLAB_MEX_FILE
        int access_hw = ssGetIWorkValue(S, ACCESS_HW);
        if (!access_hw)
        {
                /* Allow the inputs to be directly passed to the
                 * outputs for a loopback situtation.  Note that
                 * to use this capability, an output port needs to
                 * be added to the block.
                 */
```

```
                for (i = 0; i < num_channels; i++) {
                        y[i] = u[i];
                }
                return;
        }
#endif  /* MATLAB_MEX_FILE */


        for (i = 0; i < num_channels; i++) {
                dac_resolution = ssGetIWorkValue(S, i);

                /* value = (int)(dac_resolution * (u[i] / -output_reference)); */
                value = 65535* u[i]/20;
        if (value > DAC_MAX_VOLTAGE) {
                        value = DAC_MAX_VOLTAGE;
                }
                if (value < DAC_MIN_VOLTAGE) {
                        value = DAC_MIN_VOLTAGE;
                }
                switch (i) {
                        case 0:
                                outpw(DAC0, (int)  value);
                        break;
                        case 1:
                                outpw(DAC1,  (int) value);
                        break;
                }
        }
}

/* Function to compute model update */
static void mdlUpdate(x, u, S, tid)
        double *x;
        double *u;
        SimStruct *S;
        int tid;
{
}

/* Function to compute derivatives */
static void mdlDerivatives(dx, x, u, S, tid)
        double *dx;
        double *x;
        double *u;
        SimStruct *S;
        int tid;
{
}

/* Function to perform housekeeping at execution termination */
static void mdlTerminate(S)
        SimStruct *S;
{
        /*
         * This function zeros the DAC outputs.
         */
        unsigned int base_addr = ssGetIWorkValue(S, BASE_ADDRESS);
        int num_channels = ssGetIWorkValue(S, NUM_CHANNELS);
        int i;
```

```
        /* If this is a MEX-file, then accessing the hardware
         * depends on the value of the access hardware flag in the
         * S-function's dialog box.
         */
#ifdef MATLAB_MEX_FILE
        int access_hw = ssGetIWorkValue(S, ACCESS_HW);
        if (!access_hw)
                return;
#endif  /* MATLAB_MEX_FILE */

                for (i = 0; i < num_channels; i++) {

                switch (i) {
                        case 0:
                                outpw(DAC0, DAC_ZERO);
                break;
                        case 1:
                                outpw(DAC1, DAC_ZERO);
                        break;
                }
        }
}

#ifdef MATLAB_MEX_FILE          /* Is this file being compiled as a MEX-file? */
#include "simulink.c"          /* Mex glue */
#else
#include "cg_sfun.h"           /* Code generation glue */
#endif
```

BIBLIOGRAPHY

[1] E. H. M. Weck, O. Schulze, F. Michels, R. Bonse, "Optimization of Machine Tool Performance and Accuracy", Manufacturing Science and Engineering, PED-Vol. 68-2, 1994.

[2] S. J. Rober, Y. C. Shin, "Control of Cutting Force for End Milling Processes Using an Extended Model Reference Adaptive Control Scheme", Manufacturing Science and Engineering, PED-Vol. 68(2), 1994.

[3] L. K. Daneshmend, H. A. Pak, "Model Reference Adaptive Control of Feed Force in Turning", Journal of Dynamic Systems, Measurement, and Control, Sept. 1986 , Vol. 108, pg. 215-222.

[4] W. F. Ko , A Systems Engineering Approach to Design a Smart Tool Post Structure, Master's Thesis, University of Maryland at College Park, 1995.

[5] " The SMA Partnership for Synthesis and Processing of Smart Materials: Final Program Review.", Lockheed-Martin, 1995.

[6] A. Safari, " Development of Piezoelectric Composites for Transducers", Journal of Physics III France 4, July 1994, pg. 1129-1145.

[7] T. Bailey, J. Hubbard, "Distributed Piezo-Polymer Active Vibration Control of a Cantilever Beam", AIAA Journal of Guidance and Control Vol. 6(5), pg. 605-611

[8] K. Uchino, "Electrostrictive Actuators: Materials and Applications", Ceramic Bulletin, Vol. 65(4), 1986.

[9] S. M. Yang, Y. J. Lee, "Modal Analysis of Stepped Beams with Piezoelectric Materials", Journal of Sound and Vibration, Vol 176(3), pg. 289-300, 1994.

[10] H. E. Merrit, "Theory of Self-Excited Machine-Tool Chatter", Journal of Engineering for Industry, Nov. 1965, pg. 447-454.

[11] S. V. Batrakov, B. K. Tikhomirov, " Insert Lathe Tools with Vibration -Damping Holder", Machining Tools, 61(12) pg. 28-30, 1990.

[12] N. Olgac, B. T. Holm-Hansen, "A Novel Active Vibration Absorption Technique: Delayed Resonator", Journal of Sound and Vibrations, Vol. 176(1), pg. 93-104, 1994.

[13] N. Olgac, B. T. Holm-Hansen, "Tunable Active Vibration Absorber:The Delayed Resonator", Journal Dynamic Systems,Measurement, and Control, Vol. 117, pg. 513-519, 1995.

[14] G. Dold, Design of a Microprocessor-Based Adaptive Control System for Active Vibration Compensation Using PMN Actuators, Master's Thesis, University of Maryland at College Park, 1996.

[15] M. Regelbrugge, A. Carrier, W. Dickson, " Canceling Vibrations with Smart Materials: A Case Study ", SPIE Vol 2447 1995, pg 80-90.

[16] T. Comstock, Chatter Suppression by Controlled Mechanical Impedance, Ph.D. Thesis, University of Cininnati, 1968.

[17] T. Comstock, F. Tse, J. Lemon, "Applications of Controlled Mechanical Impedance for Reducing Machine Tool Vibration", Transactions of the ASME, Journal of Engineering for Industry.

[18] A. G. Ulsoy, Y. Koren, F. Rasmussen, "Principal Developments in the Adaptive Control of Machine Tools", Journal of Dynamic Systems, Measurement, and Control, June 1983 Vol. 105, pg. 107-112.

[19] O. Masory, Y. Koren, "Stability Analysis of a Constant Force Adaptive Control System for Turning", Journal of Engineering for Industry, Nov. 1985, Vol. 107 pg 295-300.

[20] M. Tomizuka, S. Zhang, "Modeling and Conventional/Adaptive PI Control of a Lathe Cutting Process", Journal of Dynamic Systems, Measurement, and Control, Dec. 1988, Vol. 110 pg 350-354.

[21] J. Rasmussen, T. Tsao, R. Hanson, S. Kapoor, " Dynamic Variable Depth of Cut Machining Using Piezoelectric Actutors ", International Journal of Machine Tool Manufacturers, 1994, Vol. 34(3) pg. 379-392.

[22] M. Shiraishi, K. Yamanaka, H. Fujita, "Optimal Contol of Chatter in Turning ", International Journal of Machine Tool Manufacturers, 1991 Vol. 31(1), pg. 31-43.

[23] S. Tewani, K. Rouch, B. Walcott, " A Study of Cutting Process Stability of a Boring Bar with Active Dynamcie Absorber ", International Journal of Machine Tool Manufacturers 1995, Vol. 35(1), pg 91-108.

[24] C. Rogers, Smart Materials, Structures, and Mathematical Issues, Technomic Pub. Co., Lancaster, PA, 1989.

[25] N. Wereley, G. Kamath, M. Hurt, "Analysis and Testing of Bingham Plastic Behavior in Semi-Active Electrorheological Fluid Dampers", Smart Materials and Structures 1996, pg. 576-590.

[26] J. Carlson, D. Catanzarite, K. St. Clair, "Commercial Magneto-Rheological Fluid Devices", 5th Int. Conf. on ER, MR Suspension and Associated Technology, Sheffield, 1995.

[27] R. Stanway, J. Sproston, A. El-Wahed, "Application of Electro-rheological Fluids in Vibration Control: A Survey", Smart Materials and Structures 1996, pg. 1-19.

[28] J. Bi , M. Anjanappa, "Investigation of Active Vibration Damping Using Magnetostrictive Mini Actuator", SPIE, Vol. 2190 pg. 171-180.

[29] C. L. Hom, S.M. Pilgrim, N. Shankar, K. Bridger, M. Massuda, S. Winzer, "Calculation of Quasi-Static Electromechanical Coupling Coefficients for Electrostrictive Ceramic Materials", IEEE Transactions on Ultrasonic, Ferroelectrics, and Frequency Control, Vol. 41(4) , pg. 542-551, July 1994.

[30] K. M. Rittenmyer, R. Y. Ting, "New Materials for Large Strain Actuator Applications", SPIE, Vol. 2189, pg. 2-13.

[31] M. Fripp, N. Hagood, " Comparison of Electrostrictive and Piezoceramic Actuation for Vibration Suppression ", SPIE Vol. 2443 , pg 334-348.

[32] D. Askeland, The Science of Engineering of Materials, PWS-KENT publishing Co., Boston, MA. (1989) pg. 678-697.

[33] S. J. Kim, J. D. Jones, "Semi-Active Control of a Composite Beam Using Embedded Piezoelectric Actuators", Smart Structures and Material, AD-Vol. 24/AMD-Vol. 123, pg. 131-138, 1991.

[34] C. L. Hom, N. Shankar, "A Fully Coupled Constitutive Model for Electrostrictive Ceramic Material", Journal of Intelligent Materials Systems and Structures, 1991.

[35] C. Liang, F. Sun, C. A. Rogers, "Coupled Electro-Mechanical Analysis of Piezoelectric Ceramic Actuator-Driven Systems: Determination of the Actuator Power Consumption and System Energy Transfer", *SPIE Smart Structures and Intelligent Systems,* Vol. 1917, pg 286-298 1993.

[36] L. Ljung, T. Glad, Modeling of Dynamic Systems, Prentice Hall, Englewood Cliffs, NJ, 1994.

[37] H. Luu, Implementation of a Designed Tool Post for Tool Vibration Compensation Using PMN Actuators, Master Thesis, University of Maryland at College Park, 1996.

[38] P. A. Ioannou, J. Sun, Robust Adaptive Control, Prentice Hall, Upper Saddle River, NJ, 1996.

[39] Personal Contact with L. Ljung (co-author of [36]).

[40] R. Chiang, M. Safonov, Robust Control Toolbox for use with Matlab® , MathWorks, Inc., Natick, Mass. 1992.

[41] Z. Eshete, In Process Machine Tool Vibration Cancellation Using PMN Actuators, Ph.D. thesis, University of Maryland at College Park, 1996.

[42] C. T. Chen, Linear System Theory and Design, Saunders College Publishing, Fort Worth, TX., 1984.

[43] K. Ogata, Discrete Time Control Systems, Prentice Hall, Englewood Cliffs, NJ, 1987.

[44] D. Auslander, C. Tham, Real-Time Software for Control, Prentice Hall, Englewood Cliffs, NJ 1990.

Other Relevant Reading Material

[1] B. Balachandran, A. H. Nayfeh, Applied Nonlinear Dynamics, John Wiley and Sons, Inc., New York, NY, 1995.

[2] J. Sirkis, W. Spillman, P. Gardiner, "The field of smart structures as seed by those working in it: survey results", SPIE Smart Structures and Intelligent Systems, Vol. 2444, pg. 18-29.

[3] P. Lawrence, K. Mauch, Real-Time Microcomputer System Design: An Introduction, McGraw-Hill Book Co. 1987.

[4] J. Mos'cin'ski, Z. Ogonowski, Advanced Control with Matlab and Simulink, Ellis Horwood, London, England, 1995.

[5] M. Green, D.J. Limebeer, Linear Robust Control, Prentice Hall, Englewood Cliffs, NJ, 1995.

[6] B. Shahian, M. Hassul, Control System Design Using Matalb, Prentice Hall, Englewood Cliffs, NJ, 1993.

[7] S. J. Elliott, "Active Control of Structure-Borne Noise", Journal of Sound and Vibration, Vol 177(5), pg 651-673 (1994).

[8] S. J. Huang, R. J. Lian, "A Dynamic Absorber with Active Vibration Control", Journal of Sound and Vibration, Vol. 178(3), pg. 323-335, 1994.

[9] S. Pilgrim, M. Massuda, A. Sutherland, "Electromechanical Determination of the High-Field Phase Transition of $Pb(Mg_{1/3}Nb_{2/3})O_3$-$PbTiO_3$-$(Ba,Sr)TiO_3$ Relaxor Ferroelectrics", Journal of American Ceramic Societies, Vol. 75(7) , pg. 1970-74, 1992.

[10] S. Pilgrim, M. Massuda, J. D. Prodey, A. P. Ritter, "Electromechanical Properties of Some $Pb(Mg_{1/3}Nb_{2/3})O_3$-$PbTiO_3$-$(Ba,Sr)TiO_3$ Ceramics: I" , Journal of American Ceramic Societies, Vol. 75(7) 1964-1969, 1992.

[11] D. Reuster, M. Kaye, F. E. Eastep "A New Basis Set for Computing the Resonant Frequencies of Vibrating Membranes", Journal of Sound and Vibration, Vol. 178(5), pg. 635-641, 1994.

[12] Z. Li, P. M. Bainum, "Vibration Control of Flexible Spacecraft Integrating a Momentum Exchange Controller and a Distributed Piezoelectric Actuator", Journal of Sound and Vibration, Vol. 177(4) pg. 539-553, 994.

[13] K. W. Wang, Y. S. Kim, D. B. Shea, "Structural Vibration Control via Electrorheological-Fluid-Based Actuators with Adaptive Viscous and Frictional Damping", Journal of Sound and Vibration, Vol. 177(2), pg. 227-237, 1994.

[14] Rajesh Seshadri, "Active Vibration Control of Beams Using Piezoelectric Actuators", Scholarly paper: University of Maryland at College Park, 1994.

[15] S. M. Yang, Y. J. Lee, "Modal Analysis of Stepped Beams with Piezoelectric Materials", Journal of Sound and Vibration, Vol 176(3), pg. 289-300, 1994.

[16] X. Bao, Z. Ounaies, V. L.Varadan, V. V. Varadan, "Active Noise Control Using Piezoelectric Actuator for a Machine", SPIE, Vol. 2189, pg 211-223.

[17] R. Igbal, Dynamic Modeling of PZ-Actuator Systems for Control Design, Master Thesis,:University of Maryland at College Park, 1988. pg. 3-11.

[18] J. B. Horner, D. J. Rutterman, P. Meckl, "Effect of Actuator Coupling on Active Vibration Control of Flexible Structures", Journal of Guidance and Control, vol 17(1), pg. 214-217, 1992.

[19] K. K. Denoyer, M. K. Kwak, "Dynamic Modeling and Vibration Suppression of a Slewing Active Structure Utilizing Piezoelectric Sensors and Actuators", Smart Structures and Intelligent Systems, Vol .1917, 1993.

[20] K. Uchino, S. Nomura, L. Cross, S. Jang, R. Newnham, "Electrostrictive Effect in Lead Magnesium Niobate Single Crystals", Journal of Applied Physics, Feb. 1980, Vol 51(2) pg. 1142-1145.

[21] D. Viehland, S. J. Jang, L. E. Cross, M. Wuttig, "Deviation from Curie-Weiss Behavior in Relaxor Ferroelectrics", Physical Review B, Oct. 1992, Vol. 46(13), pg. 8003-8006.

[22] D. Hrovat, D. L. Margolis, M. Hubbard, "An Approach Toward the Optimal Semi-Active Suspension", Journal of Dynamic Systems, Measurement, and Control, Sept. 1988, Vol. 110, pg 289-296.

[23] T. Watanabe, "A Model-Based Approach to Adaptive Control Optimization in Milling", Journal of Dynamic Systems, Measurement, and Control, March 1986, Vol. 108, pg. 56-54.

[24] E. H. M. Weck, O. Schulze, F. Michels, R. Bonse, "Optimization of Machine Tool Performance and Accuracy", Manufacturing Science and Engineering, PED-Vol. 68-2 , 1994.

[25] S. J. Rober, Y. C. Shin, "Control of Cutting Force for End Milling Processes Using an Extended Model Reference Adaptive Control Scheme", Manufacturing Science and Engineering, PED-Vol. 68(2), 1994

[26] G. L. Childs, Dielectric Aging of PMN Relaxors, Master, University of Maryland at College Park, pg. 1-10.

[27] A. R. Shahin, P. H. Meckl, J. D. Jones, " Vibration Control using Shape Memory Alloy Wires", Adaptive Structures and Composite Materials: Analysis and Application, AD-Vol. 45 / MD-Vol. 54, 1994.

[28] B. A. Frankpitt, "A Model of the Dynamics of a Lathe Toolpost that Incorporates Active Vibration Suppression", Scholarly paper: Institute for Systems Research: University of Maryland at College Park, 1994.

[29] T. T. Soong, M.C. Costantinou, "Passive and Active Structural Vibration Control in Civil Engineering", Springer-Verlag, Wien New York, 1994.

[30] Aronson, B. Aronson, " Machine Tool 101", Manufacturing Engineering, 7 part series from Jan. to July 1, 1994.

[31] R. Herzog, "Active Versus Passive Vibration Absorbers", Journal of Dynamics Systems, Measurements, and Control, Sept. 1994 Vol 116, pg 367-371.

[32] M. Appleyard, P.E. Wellstead, " Active Suspensions : Some Background ", IEE Process-Control Theory Appl., Vol 142, No. 2, March 1995 pg. 123-128.

[33] A. Burton, A. Truscott, P. Wellstead, " Analysis, Modelling and Control of an Advanced Automotive Self-Leveling Suspension System ", IEE Process-Control Theory Appl., Vol 142, No. 2, March 1995 pg. 129-139.

[34] G. Prokop, F. Sharp, " Performance Enhancement of Limited-bandwidth Active Automotive Suspensions by Road Preveiw ", IEE Process-Control Theory Appl., Vol 142, No. 2, March 1995 pg. 140-148.

[35] J. Galvagni, "Electrostrictive Actuators and Their Use in Optical Applications ", Optical Engineering, Nov. 1990, Vol. 29(11), pg. 1389-1391.

[36] B. Chen, Y. Chang, "Robust PI Controller Design for a Constant Turning Force System", International Journal of Machine Tool Manufacturers, 1991, Vol. 31(3), pg. 257-272.

[37] B. Srinivas, "The Forces in Turning", SME Increasing Productivity with Advanced Machining Concepts Clinic, Aug. 1982.