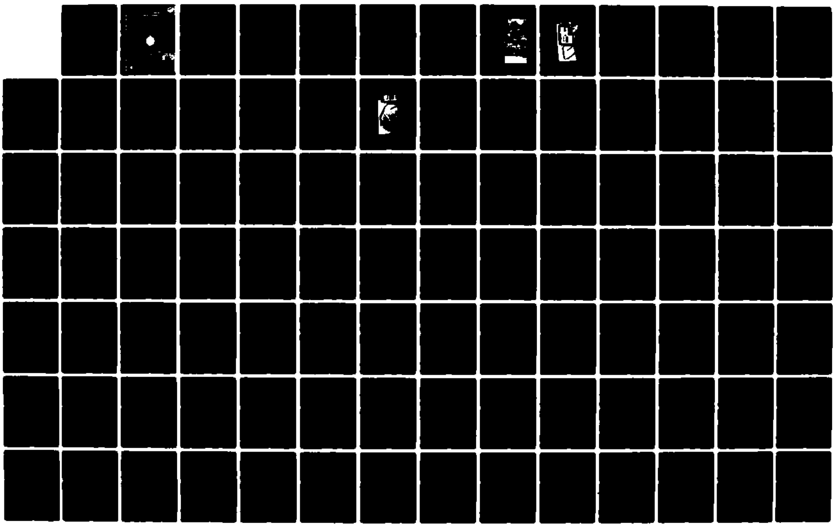
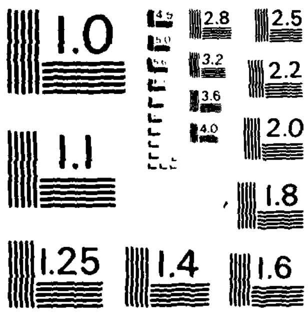
AD-A134 185 STABILITY ENHANCEMENT OF A FLEXIBLE ROBOT MANIPULATOR 1/2
(U) NAVAL ACADEMY ANNAPOLIS MD T D LOOKE 24 JUN 83
USNA-TSPR-126

UNCLASSIFIED F/G 12/1 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A134 185

U.S.N.A. - Trident Scholar project report; no. 126 (1983)

Stability Enhancement of a Flexible Robot Manipulator

A Trident Scholar Project Report
by

Midshipman First Class Thomas D. Looke, USN
Class of 1983
U. S. Naval Academy
Annapolis, Maryland

KA Knowles
Associate Prof. Kenneth A. Knowles, Advisor
Weapons and Systems Engineering Department

Accepted for Trident Scholar Committee

Charles F Powell
Chairman

June 24, 1983
Date

ABSTRACT

A computer software programming technique was developed to compensate a highly oscillatory robot system controlled by a bang-bang input. The assumptions that the system was linear and had lumped parameter characteristics allowed a fifth order, simplified dynamic model to be derived. Analysis using frequency response methods led to further simplification of the model to a third order system. Based on the third order model, a technique was developed which would compensate the system with a form of deadbeat control. Simulation of the model driven by the compensated bang-bang input verified the deadbeat response. The technique was implemented on an 8080-based microcomputer system which controlled the input. Actual system response to the compensated input was observed to be essentially free of the undesirable oscillatory motions, thus yielding an apparently rigid system.

PREFACE

This investigation was sponsored by the Trident Research Committee, Naval Academy Research Council, and conducted at the United States Naval Academy, Annapolis, Maryland. Computer simulation was conducted using the Naval Academy Time Sharing system, while the remainder of the work was carried out in the Fluid Control and Robotics Laboratory of the Weapons and Systems Engineering Department. The Technical Support Department of the Naval Academy was used extensively throughout the course of research for hardware assistance and shop facilities.

I would like to thank Associate Professor K. A. Knowles, my faculty advisor, for the guidance and wisdom which he has imparted throughout the year. Larry Heisig, John Hill, and Carvel Holton, and others of the Technical Support Department were invaluable in helping me solve some of my hardware difficulties. Bill Lowe, Rick Boyer, and Clyde Atwell were instrumental in helping me get this project on its feet and keeping it there. I would finally like to thank Eric Dyson for his drawing assistance and the use of his equipment.

This report uses many common control theory methods, terms, and symbols. It is assumed that the reader is familiar with the basic language of control systems.

TABLE OF CONTENTS

CHAPTER ONE

INTRODUCTION

> "So far the predictions about what robots will do in the home, in the factory, and in space, cannot be contradicted, although assessment as to how soon they will be able to do this has been consistently over-optimistic. Whatever the length of time required for these predictions to come true it's already obvious that man has little to fear from robots, but that he could well fear a future without them" [1]

"Robot" has become a household term. The Random House Dictionary (Revised Edition, 1975) defines a robot as "a machine which resembles a man and does mechanically routine tasks on command." The general public recalls visions of R2-D2 and C3PO, as shown in Figure 1.1, or the unnamed robot from "Lost in Space," when confronted with this subject. Seldom does anyone picture a robot like that shown in Figure 1.2.

For the purposes of this paper, it is important to define what a robot is and what it is not. A robot is a computer controlled, reprogrammable, mechanical device which can perform multiple manipulative tasks without external guidance. An automated machine tool is not a robot because, even though it may be computer controlled, it can only perform the single task for which it was designed. There are mechanical manipulators which can perform multiple tasks, but require a human in the control loop. These systems are

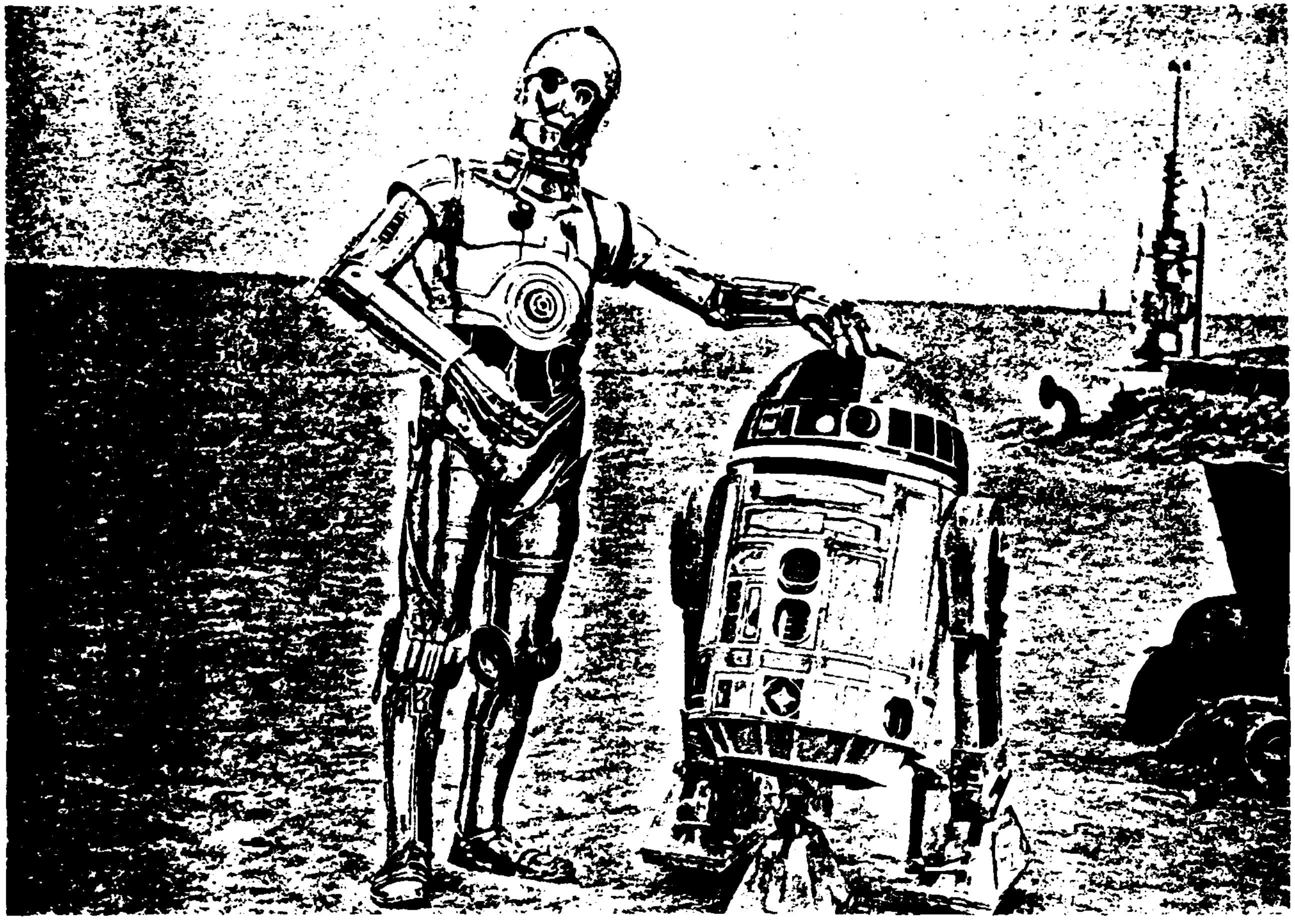


FIGURE 1.1 ROBOTS OF THE FANTASY WORLD

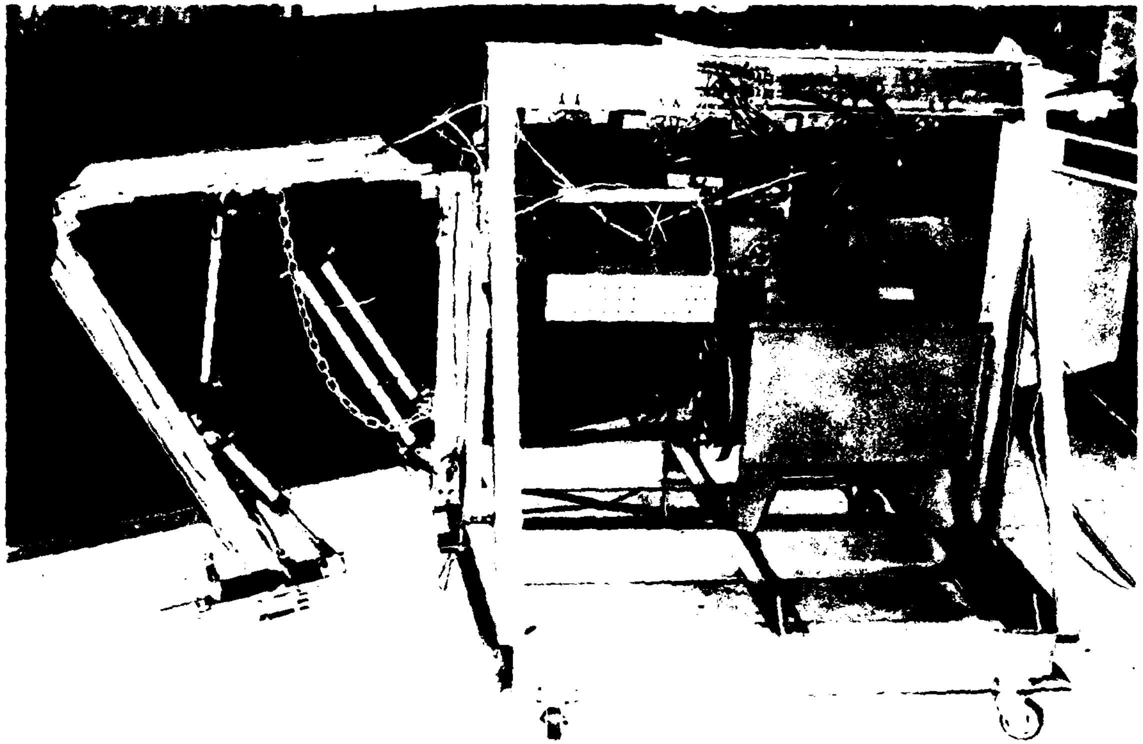


FIGURE 1.2 THE DINO ROBOT SYSTEM

known as teleoperators. A robot differs from a teleoperator in that it must be able to perform its tasks independent of all human control.

Robots are suited to perform many tasks. They are currently used in industry to perform jobs ranging from stacking heavy boxes in warehouses to assembling watches. In the home, robots could be used for every kind of household chore, from ironing to vacuuming the rug.

This project was concerned with the type of robot commonly found in industry. The logic capabilities of a modern computer combined with the manipulative skills of a mechanical arm, make robots appealing as industrial workhorses. As technology has advanced, so have the tasks which have been assigned to robots. Designers have become increasingly aware of the need to build robots which have very precise movements and are very accurate. Only these robots can perform the assembly procedures which are going to be routine tasks of the future.

At present, designers tend to rely on a single strategy to achieve accuracy: design the robot to be as mechanically rigid as possible. This approach accomplishes the desired goal, but at the expense of increased weight, increased size, increased cost, enormous power requirements, and decreased adaptability of tasking. In the future, design specifications will demand that these costs be decreased. To accomplish this, the designer will have to develop a robot system which does not have parasitic (excessive) mass and power requirements. In giving up the excess mass, however, the designer will be losing some of the mechanical rigidity which was required to make the robot precise.

To meet the reduced power consumption design specifications, robots of the future will possess a greater degree of mechanical flexibility. Flexibility, as it will be used in this discussion, is the tendency of a system to show imprecise motions due to the elastic deformation of its components. A very flexible system is usually greatly underdamped and thus displays highly oscillatory motion. If a designer is not restricted by excessive rigidity requirements, he can more easily develop a light, small, inexpensive robot system which will be power efficient. The task of the robot designer of the future will be to develop a robot system with all of the properties discussed above, but which appears rigid in its motion in order to achieve the desired degree of positional accuracy.

The goal of the research outlined herein was to take an existing robot manipulator which was very flexible in one axis of motion and enhance its performance by making this motion appear to be more rigid. This was accomplished by developing a computer compensation program which modified the robot control signals in such a manner as to permit rapid manipulator motions, but without the usual, undesirable mechanical oscillations.

CHAPTER TWO

THE DINO SYSTEM

The subject of the research was a robot manipulator, shown in Figure 1.2. It is located in the Weapons and Systems Engineering Department of the United States Naval Academy, in the Fluid Control and Robotics Laboratory of the Weapons and Systems Engineering Department, and it has been nicknamed "DINO," short for "dinosaur," which it resembles.

A discussion of the hardware of the system is very important to understand some of the limitations of the investigation. The robot manipulator system shown in Figure 1.2 has a well defined range of motion, limited by the scope of the pistons which control motion. The system has four degrees of freedom. In other words, the end effector (hand) may be moved anywhere within its three dimensional range of motion, and then oriented along one axis.

The DINO system was a flexible robot system, but only one degree of freedom was greatly affected by this flexibility. The axis of this predominant flexibility was the vertical, or z-axis. In other words, motion in the horizontal plane, centered on the vertical hinges, showed the characteristic underdamped qualities of a very flexible system. It was this motion, and only this motion, which the investigator tried to compensate. There was no great need to compensate the other axes, as they already appeared to be fairly rigid.

The mechanical components of the system included the following:

an aluminum frame

aluminum I-beam forearm and upper arm

solenoid control valves

hydraulic pistons

a five horsepower hydraulic pump

specially designed precision hinges for shoulder and elbow joints

household hinges for the horizontal axis and wrist joint

50K potentiometers for measuring steady state position

an end effector (hand) designed by Associate Professor K. A. Knowles

a Schaevitz (1-G) Accelerometer for measuring instantaneous acceleration

Solenoid actuators were utilized because they are relatively inexpensive, very reliable, and tolerant of fluid contamination. They were also the most restrictive elements of the system, because they were bang-bang devices. Bang-bang means that they could only produce three outputs: a fixed flow rate to the pistons in one direction, zero flow to the pistons, and a fixed flow rate to the pistons in the opposite direction. Such inputs are normally referred to as being positive steps, zero, and negative steps, respectively.

The usual control compensation techniques did not apply to this system because there was no means for the magnitude of the input to be altered. An input could only be altered by controlling its on and off times, not its magnitude.

The actuators were controlled by an electrical signal coming from a microcomputer. The computer hardware included an 8080-Based microcomputer with twin floppy disk drives. The software language used was Compiled Microsoft BASIC operating under the CP/M Operating System. A high level language was chosen to facilitate program writing and implementation. Compiled microsoft BASIC is almost as fast as Microsoft FORTRAN, and it provided adequate control of the system.

CHAPTER THREE

SYSTEM IDENTIFICATION AND MODELING

The purpose of this chapter is to outline the procedure used to create a model of the flexible robot manipulator system to be compensated. A model is an approximate mathematical description of a physical system. If an engineer wishes to compensate a system using analytical methods, he must first create an appropriate mathematical model of the system which will allow him to do this.

The procedure outlined in this chapter for finding a model to describe the system to be compensated is not unique to this project. There are fundamental mathematical and experimental tools which are available and were used to obtain the mathematical description of the system. In some ways, this system is similar to most mechanical systems, allowing for some general analytical tools to be of assistance in identifying system characteristics. In other ways, the peculiarities of the system and the goals of the project called for unique analyses and assumptions.

3.1 The Basis for the Model

A knowledge of the physical laws which govern this system was the basis for constructing the model. In every case, a model is a simplification of the actual system. Therefore, it is the task of the engineer to decide which simplifying assumptions he wishes to apply to his model. It should be mentioned that all of the simplifying assumptions were chosen to facilitate the analysis of the compensation technique as well as to accurately describe the system. There is an inherent trade-off between these two factors. To develop a simple and realistic compensation technique, it is desirable to have a simple model of the physical system. It is likewise desirable for the model to accurately describe the system, but as a model more closely resembles the actual system, it necessarily becomes more complicated.

The first assumption made was that the system was deterministic. This meant that the system output (response) was solely a function of the system input (excitation) and the system parameters. In other words, the system was not random.

For the purposes of the model, it was assumed that the parameters of the system were fixed and constant. This assumption was chosen because of the difficulty of measuring most of the system parameters and the impossibility of controlling them with the computer.

One of the most important assumptions was that the system was linear. This meant that the system obeyed the laws of superposition.

These laws are as follows:[2]

1) Multiplying the inputs by any constant must multiply the output by the same constant.

$$f(ax) = ay$$

where $f(x) = y$ and a = constant

2) The response to several inputs applied simultaneously must be the sum of the individual responses to each input applied separately.

$$f(x + u) = y + z$$

where $f(x) = y$

and $f(u) = z$

It was difficult to test the conditions of superposition to determine the validity of the assumption of linearity, because of the nature of the input. The input was limited to a positive step input, u, a negative step input, -u, and a zero step input, 0. The first condition could only be tested as follows: $f(u) = -f(-u)$ where $a = -1$. Any other test for the first condition would be trivial. The second condition could not be tested at all, because only one input could be fed into the system at a time. The second condition calls for two inputs, which is impossible for this system.

The final assumption was that the manipulator could be described by a lumped parameter system. Therefore, the system could be modeled with a finite number of state variables.[3] This assumption greatly facilitated the mathematical manipulation of the dynamics of the system.

The model was thus governed by the assumptions which were deemed important by the investigator. It was then the investigator's task to determine the transfer function characteristics of the system based on these assumptions.

3.2 Methods of Determining a Transfer Function

There are three general methods by which the characteristic parameters of physical systems may be determined:

(1) System analysis leads to a transfer function in terms of system parameters which are evaluated by experimental tests. In the analysis of a translational hydraulic pilot valve and power piston, for example, the behavior of the various components is described by a set of equations based on physical laws. On the basis of the probable excitation and load, suitable linearizing assumptions are made. The transfer function is then derived from these linear equations. Finally, the parameters (the oil compressibility, the flow/displacement ratio of the pilot valve, the load mass, spring constant, damping, etc.) are evaluated either experimentally or theoretically.

(2) The sinusoidal characteristics of the system are determined. With the components excited by a variable-frequency sine-wave generator, the input and output amplitude and phase are measured to yield the frequency response and phase chracteristic of the controlled system.

(3) Transient tests are used, with the response system determined when the input is a step function; an impulse function, or a more general transient input. The system is characterized in the time domain.[4]

The first method of determining physical characteristics is the dynamic approach to modeling physical systems. The second method describes the Bode response analysis, and the final method is the transient response analysis. Each of these techniques was used to verify some aspect of the transfer function characteristics of the flexible robot manipulator (DINO).

The dynamic approach to modeling physical systems was used in conjunction with the assumptions that the system could be considered a lumped parameter system to find the expected form of the transfer function. Bode response analysis was used to experimentally verify the results of the dynamic approach, and (hopefully) to simplify them. The transient response method was used to find the constants associated with the system parameters evaluated in the dynamic approach.

3.3 The Dynamic Approach

The horizontal rotational motion of the robot manipulator shown in Figure 3.3.1 was modeled as a rotational mechanical system. Because the system was assumed to be a lumped parameter system, the manipulator was divided into a finite number of inertial masses connected by joints which contributed a torsional spring constant and viscous friction. Figure 3.3.2 shows the diagram of the system. Figure 3.3.3 shows a network representation of the system.

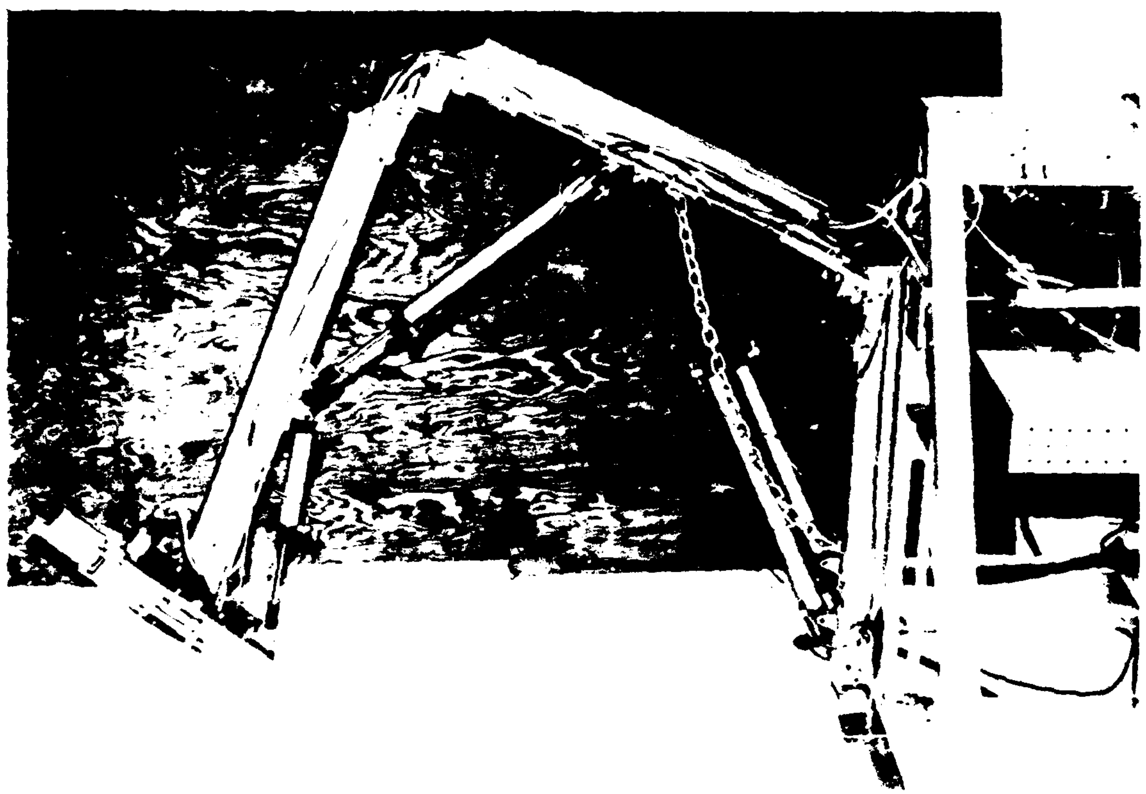


FIGURE 3.3.1 FLEXIBLE ROBOT MANIPULATOR

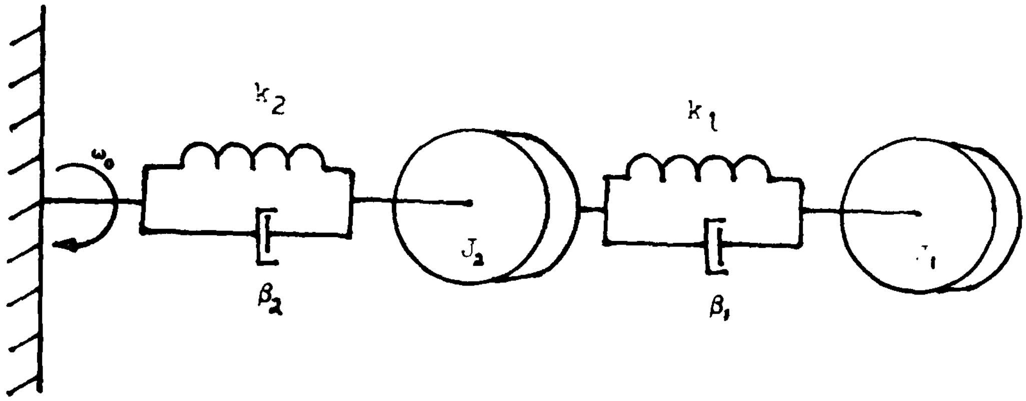


FIGURE 3.3.2 DIAGRAM OF ROBOT MANIPULATOR

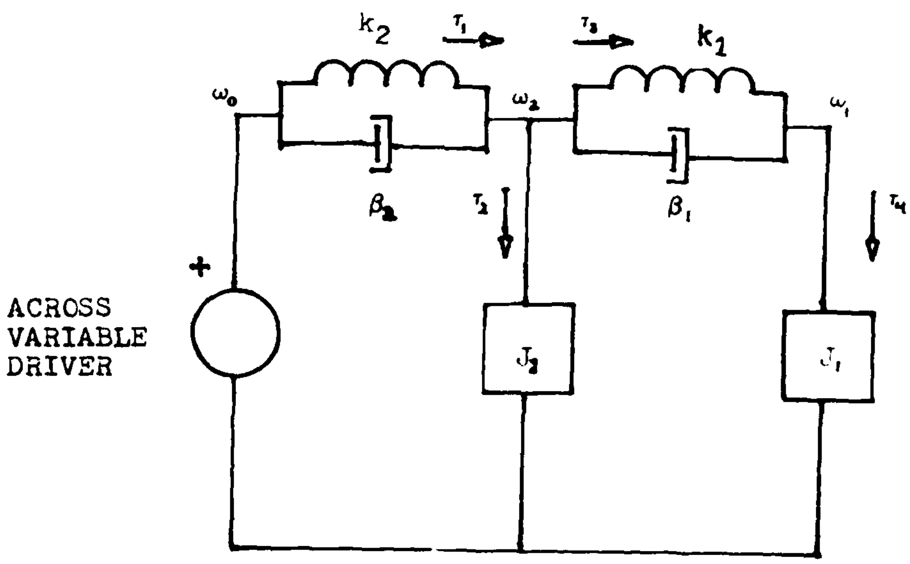


FIGURE 3.3.3 NETWORK DIAGRAM OF ROBOT MANIPULATOR

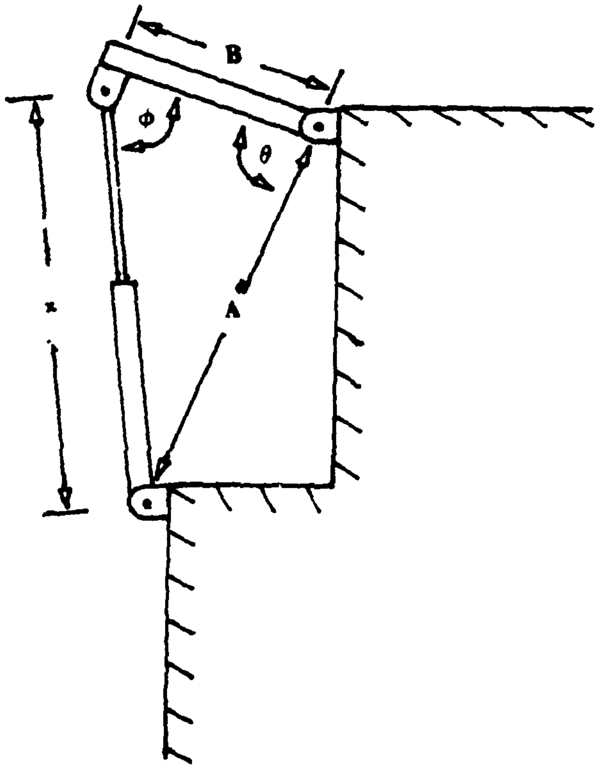


FIGURE 3.3.4 DIAGRAM OF INPUT SYSTEM

It could not automatically be assumed that the across variable driver was constant. Figure 3.3.4 shows a diagram of the mechanical input. The following analysis shows that the across variable driver was not a function of any of the chosen states. Therefore, Figure 3.3.3 could be used to analyze the transfer function of the system.

Figure 3.3.4 shows a diagram of the piston input which controls motion in the horizontal plane. The viewer is looking down on the input system from the positive z-axis. The piston, whose length is x, is considered to be a constant velocity input device. With this assumption, $\dot{\omega}_0$, or the time rate of change of θ may be found. $\frac{d\theta}{dt}$ for Figure 3.3.4 is equal to ω_0 in Figure 3.3.3

The Law of Cosines states that

$$x^2 = A^2 + B^2 - 2AB\cos(\theta) \qquad 3\text{-}1$$

where from Figure 3.3.4, A and B are constant.

The time derivative of 3-1 yields

$$2x\frac{dx}{dt} = 2AB\sin(\theta)\frac{d\theta}{dt}$$

Given $\frac{dx}{dt} = V_0$ and $\frac{d\theta}{dt} = \omega_0$

$$2xV_0 = \omega_0\, 2AB\sin(\theta)$$

$$\omega_0 = \frac{V_0}{AB}\,\frac{A}{\sin(\theta)}$$

The Law of Sines states that

$$\frac{x}{\sin\theta} = \frac{A}{\sin(\phi)}$$

Therefore

$$\omega_0 = \frac{V_0}{B}\csc(\phi) \qquad 3\text{-}2$$

Using mathematical modeling techniques and the network shown in Figure 3.3.3, the system transfer function was derived as follows:

Using nodal analysis

$$\tau_1 = \tau_2 + \tau_3 \qquad 3\text{-}3$$

$$\tau_3 = \tau_4 \qquad 3\text{-}4$$

where

$$\tau_1 = (\omega_0 - \omega_2)(\beta_2 + k_2/s)$$

$$\tau_2 = \omega_2(sJ_2)$$

$$\tau_3 = (\omega_2 - \omega_1)(\beta_1 + k_1/s)$$

$$\tau_4 = \omega_1(sJ_1)$$

Substituting into 3-3 and 3-4 yields

$$(\omega_0 - \omega_2)(\beta_2 + k_2/s) = \omega_2(sJ_2) + (\omega_2 - \omega_1)(\beta_1 + k_1/s)$$

$$(\omega_2 - \omega_1)(\beta_1 + k_1/s) = \omega_1(sJ_1)$$

Isolating variables

$$\omega_0(\beta_2 + k_2/s) = \omega_2(\beta_1 + \beta_2 + \frac{k_1k_2}{s} + sJ_2) - \omega_1(\beta_1 + \frac{k_1}{s})$$

$$0 = -\omega_2(\beta_1 + \frac{k_1}{s}) + \omega_1(\beta_1 + \frac{k_1}{s} + sJ_1)$$

Solving for $\frac{\omega_1}{\omega_0}$, the transfer function is found to be

$$\frac{\omega_1}{\omega_0} = \frac{\left[\frac{\beta_2\beta_1}{J_1J_2}\right]\left[s^2 + s\left(\frac{\beta_1k_2 + \beta_2k_1}{\beta_2\beta_1}\right) + \frac{k_1k_2}{\beta_1\beta_2}\right]}{s^4 + s^3\left(\frac{\beta_1J_1 + \beta_2J_1 + \beta_1J_2}{J_1J_2}\right) + s^2\left(\frac{\beta_1\beta_2 + k_1J_1 + k_2J_1 + k_1J_2}{J_1J_2}\right) + s\left(\frac{\beta_1k_2 + \beta_2k_1}{J_1J_2}\right) + \frac{k_1k_2}{J_1J_2}} \tag{3-5}$$

Equation 3-5 shows that the expected response of the system, based on the assumptions discussed previously, was fourth order. If the assumptions were valid, then the experimental analysis which follows should verify equation 3-5 with data taken from the actual system.

3.4 Bode Response Analysis

Using dynamic methods, the transfer function of the model was found automatically. It was desired to verify the analytical findings by experimental methods. Using Bode techniques, it was hoped that the model could be further simplified based on actual data obtained from the system.

An input to a linear system will yield an output which may be calculated using the transfer function. Given a linear system, an analysis of the response of the system given a particular input may be helpful in discovering the transfer function of the system. If the input is a variable-frequency sine-wave signal, then the plot of the ratio of the output magnitude to the input magnitude (in decibels) versus log frequency will yield a Bode plot. This technique is especially useful for determining the nature of the poles and zeros of a transfer function.

For transfer functions of linear systems, there are four characteristics which affect the Bode plot. [5]

Case (1) A constant, K

Case (2) Real zeros or poles of the form

$$(s + z_1) \text{ or } \frac{1}{(s + p_1)}$$

Case (3) A zero or pole at the origin: s or $\frac{1}{s}$

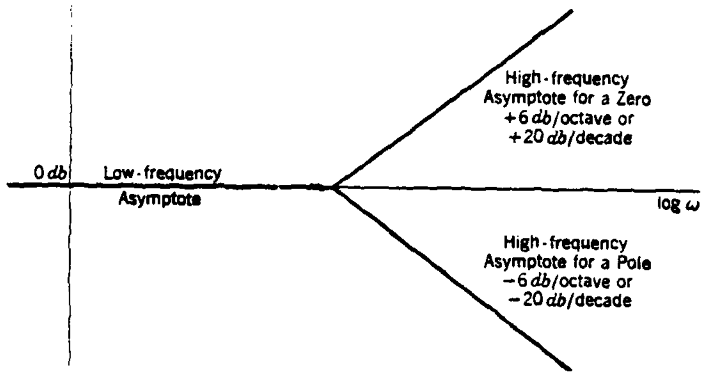
Case (4) Complex conjugate zeros or poles:

$$(s^2 + 2\zeta\omega_n s + \omega_n^2) \text{ or } \frac{1}{(s^2 + 2\zeta\omega_n s + \omega_n^2)}$$

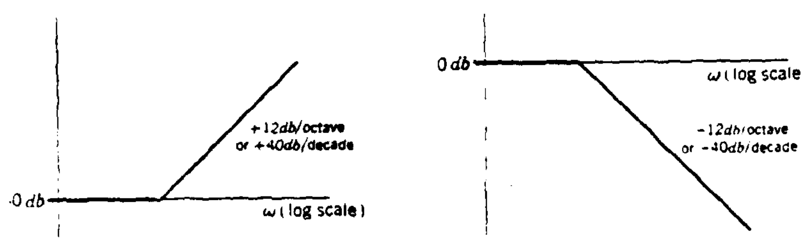
ζ = damping ratio

ω_n = natural frequency

When making a Bode plot, it is necessary to take into account the effect of each part of the transfer function. The following rules are used to plot a straight line approximation of the Bode plot of a system. The effect of a case (1) term moves the entire magnitude plot up or down as a function of 20 log K. Terms of the transfer function which contain real zeros or poles as in case (2) have certain frequency response characteristics as shown in Figure 3.4.1. A zero or pole at the origin, case (3), is plotted as a straight line with a slope of 20 dB/decade for a zero and -20 dB/decade for a pole. The approximation of a complex pole or zero is very similar to that of a real pole or zero, except for high frequencies. The case is shown in Figure 3.4.2. [6]



[illegible] 3.4.1 ASYMPTOTIC APPROXIMATION OF [illegible]
POLE OR [illegible]



Zero Case

Pole Case

FIGURE 3.4.2 [illegible] APPROXIMATI[illegible]
[illegible] OR ZEROS

Once each part of the transfer function has been plotted as a straight line approximation, the complete gain curve may be approximated by adding the gains of each part for every frequency. This does not yield the final Bode plot, though. The actual Bode plot representations of cases (2) and (4) are not the same as the straight line approximations.

There are rules which govern the correction of asymptotic Bode gain plots to get the actual plot. For case (2), gain corrections are a function of the location of the pole or zero as shown in Figure 3.4.3. For case (4), complex conjugate pairs of poles or zeros, the difference between the asymptotic plot and the actual curve depends on the value of ζ, with a variety of shapes realizable for the actual curve. Figure 3.4.4 shows how a Bode plot may change as a function of ζ.

After all corrections for errors in asymptotic plotting have been made, the resulting graph represents the Bode plot of the system. For a real system, the transfer function is unknown and the Bode plot is derived experimentally by measuring input and output magnitudes at various frequencies. Once a Bode plot is drawn for an actual system, the sequence of events leading from transfer function to Bode plot described above can be reversed to find a transfer function from the frequency response characteristics of the system.

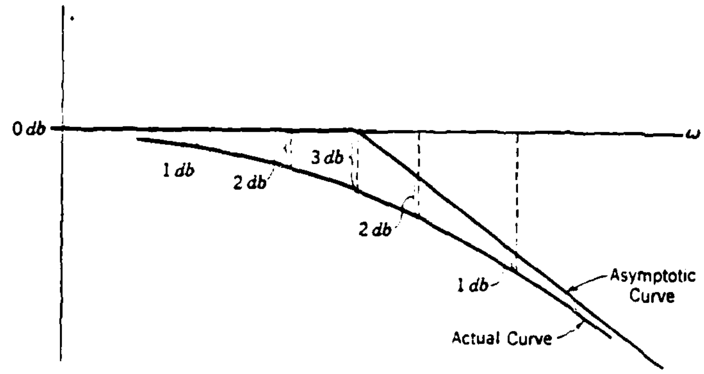


FIGURE 3.4.3 CORRECTIONS TO ASYMPTOTIC APPROXIMATION
OF A REAL POLE OR ZERO

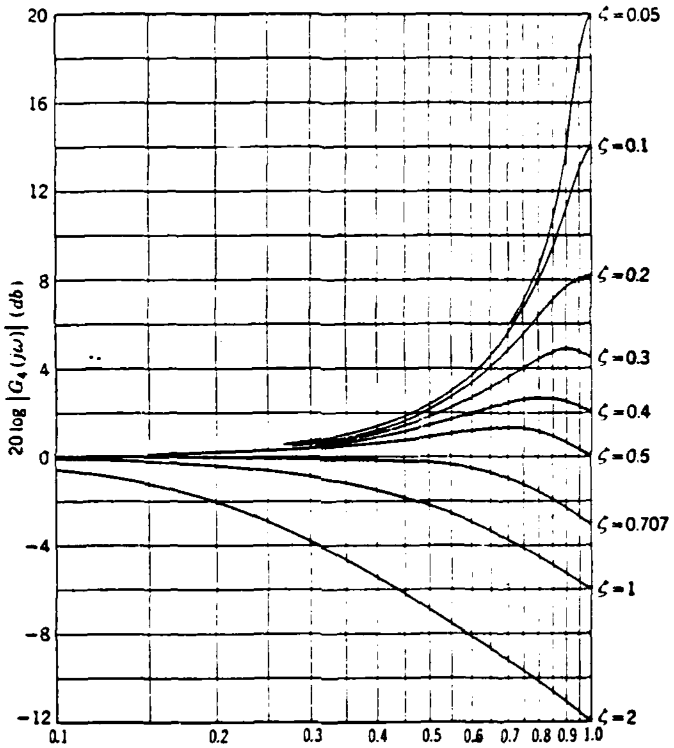


FIGURE 3.4.4 CORRECTIONS TO ASYMPTOTIC [illegible]
OF A COMPLEX PAIR OF POLES OR ZEROS

For a Bode representation of DINO to be obtained, a sine-wave input and a method of reading the output had to be devised. Bang-bang actuators (like those on DINO) do not readily lend themselves as input devices for Bode analysis. There is no easy way to vary the amplitude of a bang-bang input to make it appear sinusoidal. Therefore, an approximation of a sine-wave must be considered.

The easiest periodic input to create from bang-bang actuators is a square wave, as shown in Figure 3.4.5 (a). A series analysis would show that a square wave is actually made up of sine-waves of many frequencies superimposed upon each other. If a square wave is subjected to a low-pass filtering device of some kind, then the output would be a closer approximation of a pure sine wave. DINO's hydraulic pistons act as filtering devices which eliminate many of the overtones associated with a square wave. Therefore, the assumption that a square wave input to this system approximates a sine wave is not unwarranted.

There is a problem with using a square wave approximation of a sine-wave input. This problem is reading the output magnitude when the frequency of the input is less than the natural frequency of the system. The output is then complicated with transients which cause more than one peak to occur during each excitation period.

Other possible inputs were also considered. One was a frequency modulated input shown in Figure 3.4.5 (b). The input, in this case, would consist of a series of short pulses. The number of pulses contained within each sampling period would vary as a constant multiple of the magnitude of a sampled sine-wave. The number of sampling times within each period of the input would have to be chosen carefully so that there would be enough samples to approximate a sine-wave. Thus, the sampling rate would have to be much faster than the desired input frequency. The problem with this type of approximation was that, to make this method effective, the pulse width of each pulse within the series had to be no more than a set maximum governed by the desired frequency input and the number of samples necessary to make the approximation valid. The problem arose from the fact that the system would not follow a pulse of less than a minimum fixed time. The maximum pulse width governed by the desired frequency input and sampling rate was much less than the minimum fixed time needed for the system to follow a pulse input. The net result was that the system mechanically ignored the entire input.

A second alternative to the square wave approximation was a pulse width modulated input similar to that shown in Figure 3.4.5 (c). Similar to frequency modulation, the pulse width applied to the system within a sampling period would be proportional to the magnitude of a sampled sine-wave.

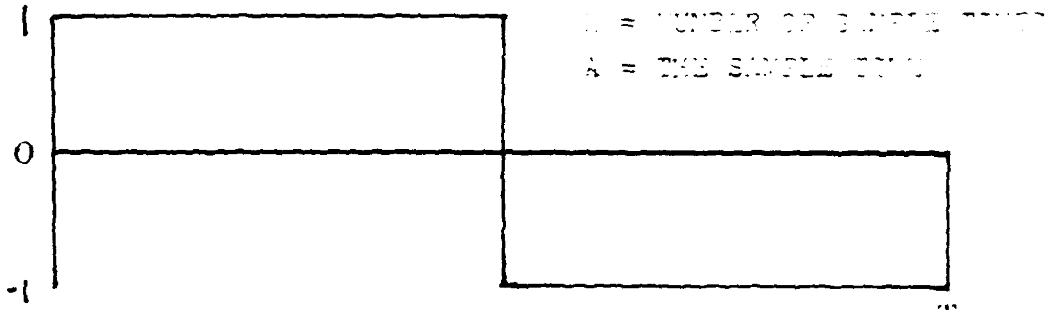


FIGURE 3.4.5(a) SQUARE WAVE INPUT

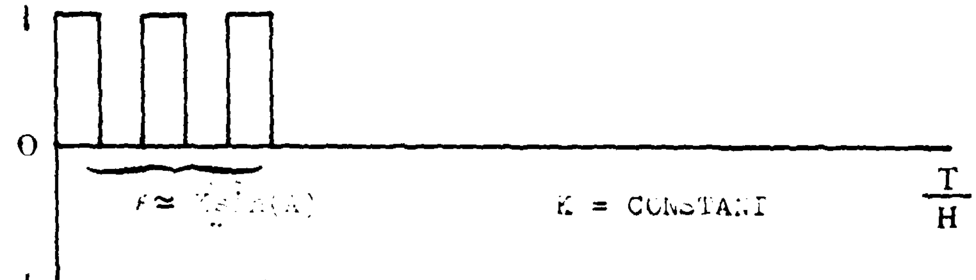


FIGURE 3.4.5(b) FREQUENCY MODULATED INPUT

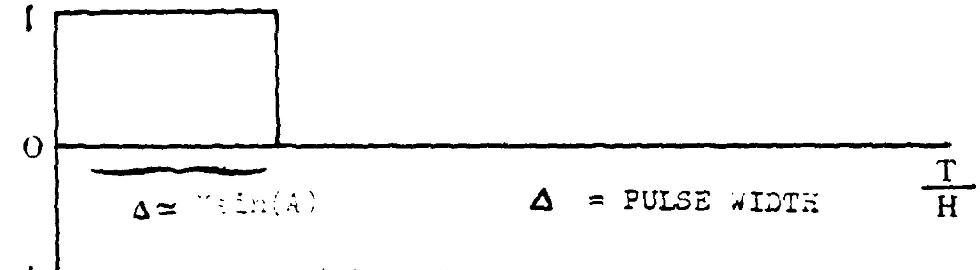


FIGURE 3.4.5(c) PULSE WIDTH MODULATED INPUT

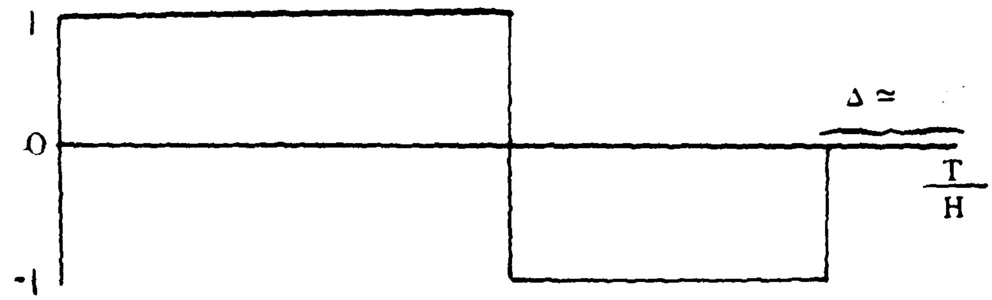


FIGURE 3.4.5(d) [illegible] INPUT

Again, the sampling rate would be much faster than the desired frequency input. This method had the same problems as the frequency method, but not quite as severe. The system would not ignore the entire input, but only those pulses which were less than the minimum width. Thus, the actual input to the system would appear as a sine wave except when the pulse width was too narrow, at which time the input would appear to be zero.

The final method considered was an off-time modulation. In this method, a very fast square wave would be input where the period of the wave would be equal to the desired sampling time. In order to vary the input as a sine-wave, one half of the square wave would be reduced in width by a constant multiple of a sampled sine-wave, as shown in Figure 3.4.5 (d). This method developed the same problems as the frequency and pulse width methods. To get the system to react properly (taking the minimum pulse width into account), the maximum frequency available was too low for the purposes of analysis. An additional problem arose as well. If a square wave was input from the computer to the actuators, the system would not necessarily remain stationary or oscillate about a fixed point. In the DINO system, there was a bias to the right. This meant that when the piston controlling horizontal motion was instructed to push the manipulator to the right, it always moved considerably farther than when it was instructed to pull the manipulator to the left for the same amount of time. Therefore, when the square wave was complete on both halves of the sampling period, the system would not remain stationary, but would move to the right. This drift would greatly hamper the analysis of the output.

The square wave approximation was best suited for the purposes of this project. It could be used to evaluate high frequency inputs which the others could not. No digital sampling was necessary, and the frequency of the input could be varied easily. The problem associated with reading the output due to unwanted transients at low frequencies could be solved by picking a common point at each frequency where the magnitude would be evaluated. For analysis, the maximum peak was used as the magnitude of the output sine-wave when there was more than one peak.

With a square wave input from the computer, it was easy to measure the magnitude and the frequency of the input at most frequencies. Measuring the mechanical output was more difficult. The method chosen was based on the desires of the investigator and the availability of measuring equipment.

It was desired to measure position, velocity, and acceleration along the path of motion in the horizontal plane, with the Bode output of velocity as the main objective. The method chosen was to measure acceleration directly, using an accelerometer, and to obtain velocity and position by using real-time analog computer integration.

An accelerometer was mounted on the end effector (hand) of the manipulator at right angles to the radial line connecting the hand and the axis of rotation for the horizontal plane. Its purpose was to measure acceleration along the path of motion in the horizontal plane. The output of the accelerometer was input into an analog integrator to get velocity along the path of motion. The velocity output was again integrated to give position along the path of motion in the horizontal plane.

There were many problems in obtaining an acceptable output. The output of the accelerometer was plagued with high frequency noise due to the hydraulic pump vibrating through the frame. The velocity and position output were victims of analog drift. This was due to a constant voltage bias output from the accelerometer while the accelerometer was at rest. The bias was easily countered by an equal but opposite constant voltage supplied to the input of the analog integrator. The problem with this approach was that the bias voltage output from the accelerometer changed as a function of the position of the end effector with respect to the axis of rotation. The compensatory voltage input had to be adjusted manually or by analog feedback. The feedback approach affected the output response of the system and was found to be inferior to manually adjusting the constant voltage correction. Manual adjustment was less efficient, but it yielded more acceptable output data.

The analog drift was very bad for the position output due to double integration. Because of the difficulty of interpretating the output, the position Bode plot was never investigated in any detail. On the other hand, the velocity output was quite adequate for examination. Using both methods described above to compensate for the analog drift, Bode plots were obtained for velocity. These are shown in Figure 3.4.6 to 3.4.9.

A comparison of the plots showed that for low frequencies, the method of countering the analog drift had little influence on the output. For high frequencies, though, the feedback method of compensating for the analog drift showed some undesired filtering characteristics.

Bode plots were also made of acceleration. As shown by Figures 3.4.10 and 3.4.11 these were not as helpful as the velocity Bode plots. This was mainly due to the difficulty in reading the magnitude of the output due to high frequency noise interference from the hydraulic pump.

The purpose of obtaining Bode plots was to verify the general form of the transfer function found in the dynamic approach. In order to do this, the knowledge of the relation between Bode plots and root locations in the s-plane, as discussed previously, had to be applied.

A model will not match perfectly every characteristic of an experimentally derived Bode plot; therefore, it was important to find the general characteristics of the Bode

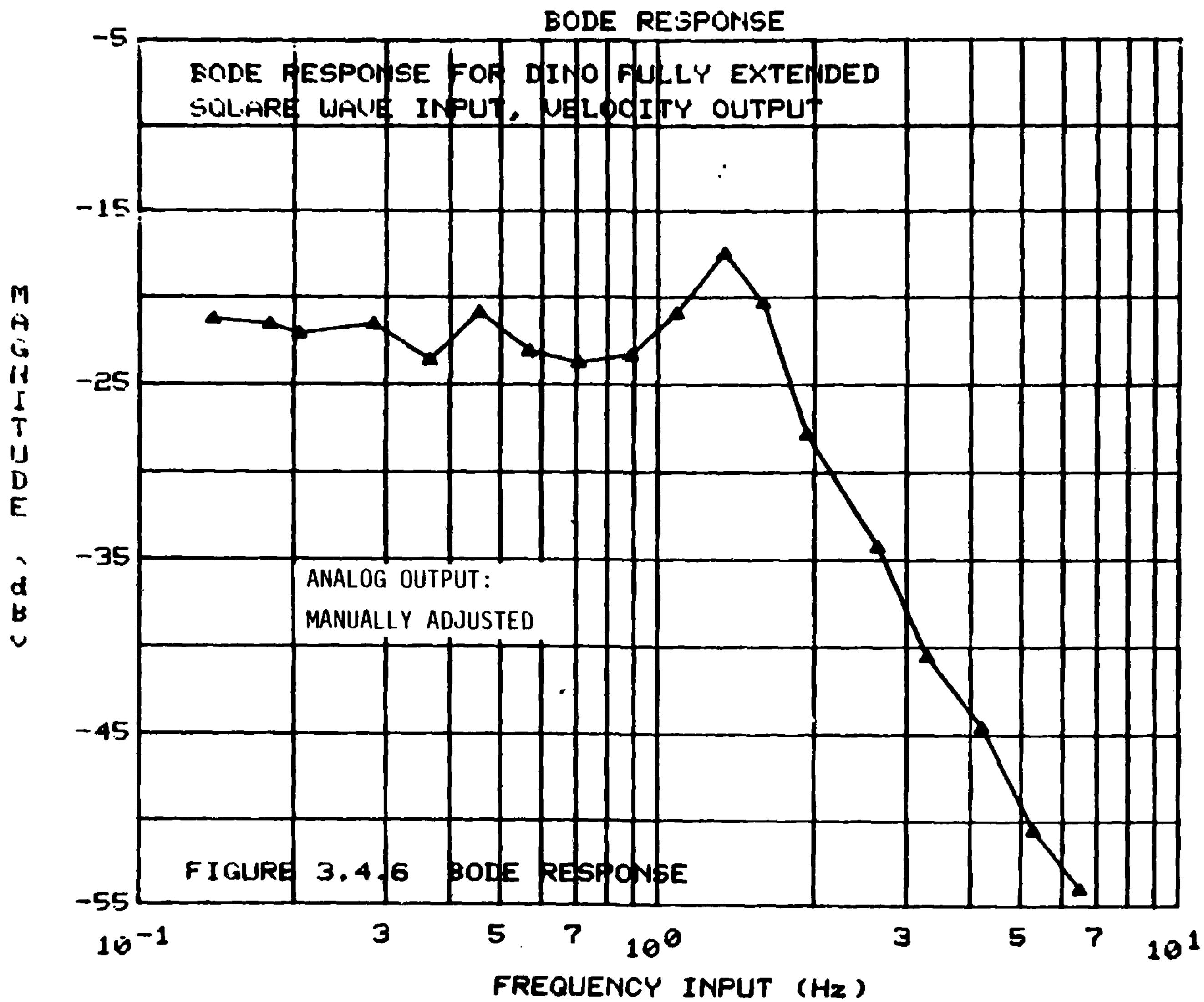


FIGURE 3.4.6 BODE RESPONSE

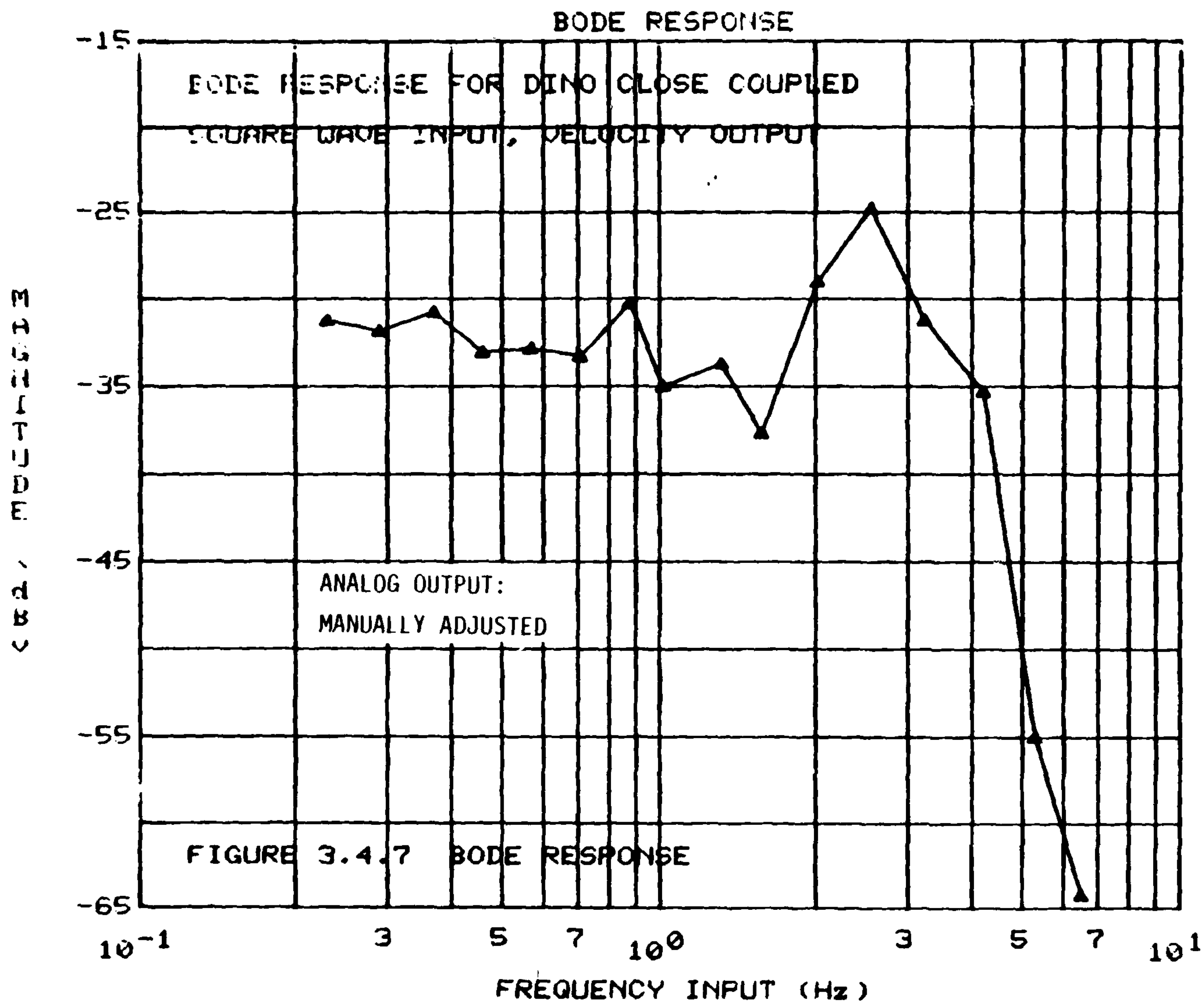


FIGURE 3.4.7 BODE RESPONSE

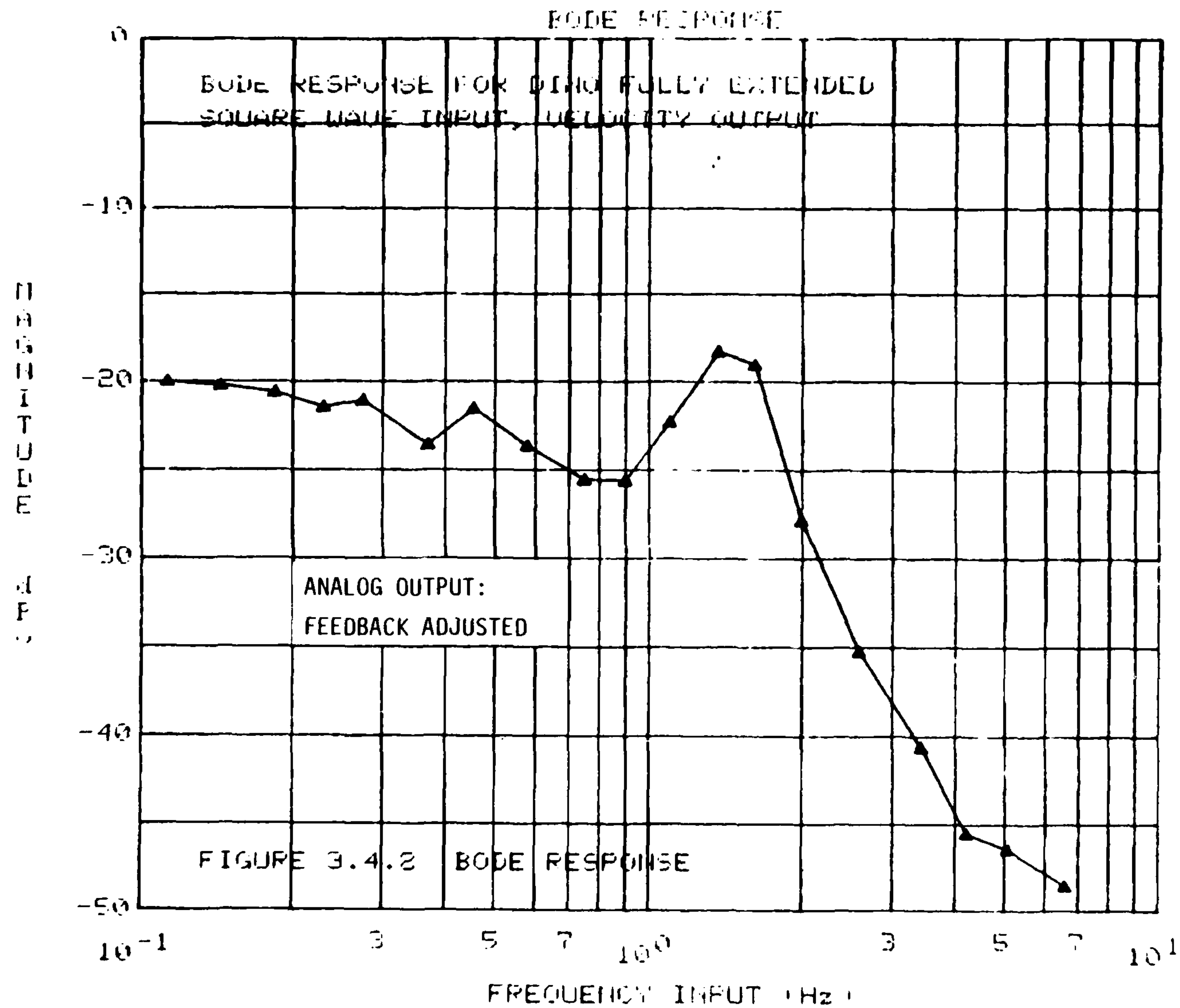


FIGURE 3.4.2 BODE RESPONSE

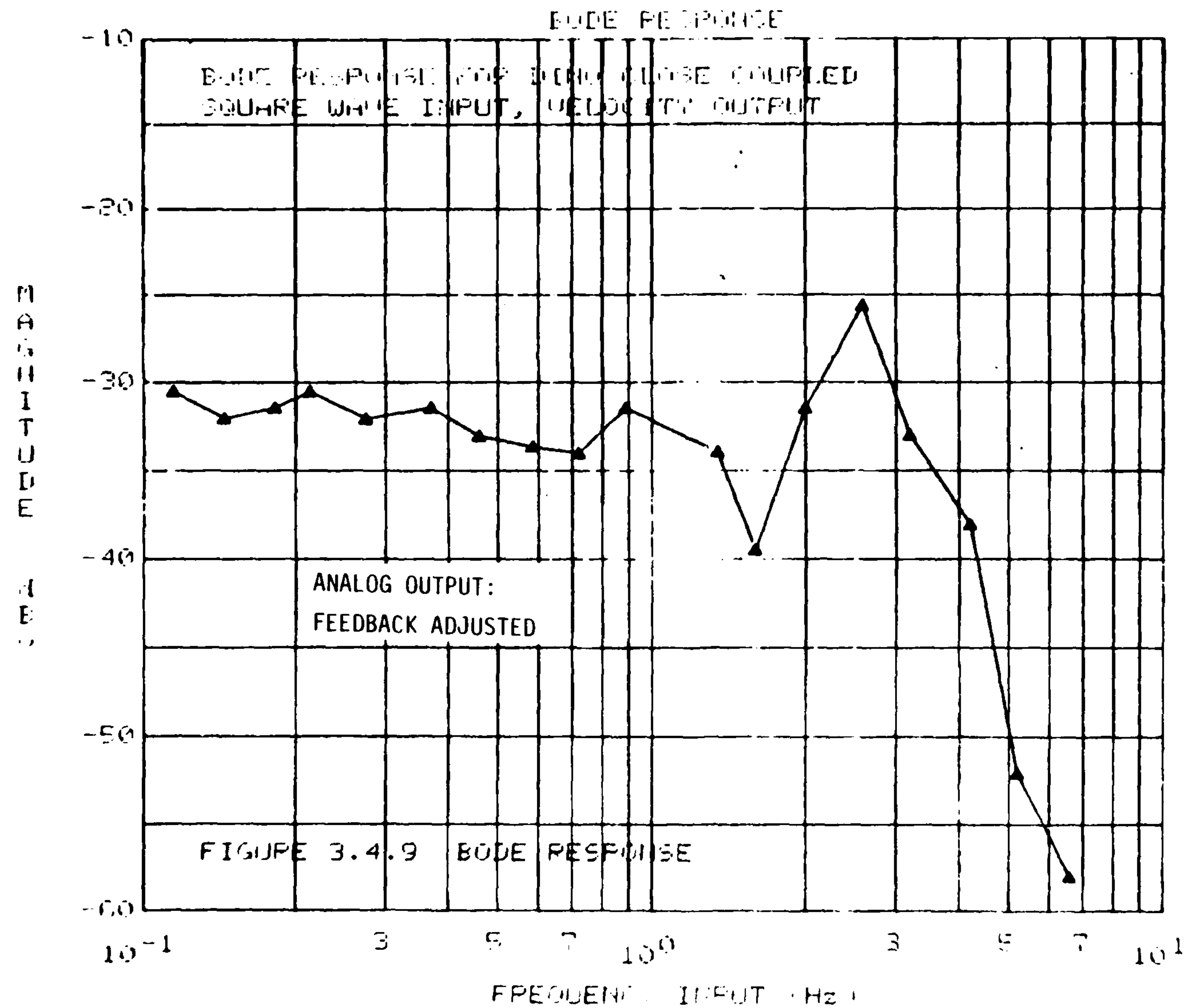


FIGURE 3.4.9 BODE RESPONSE

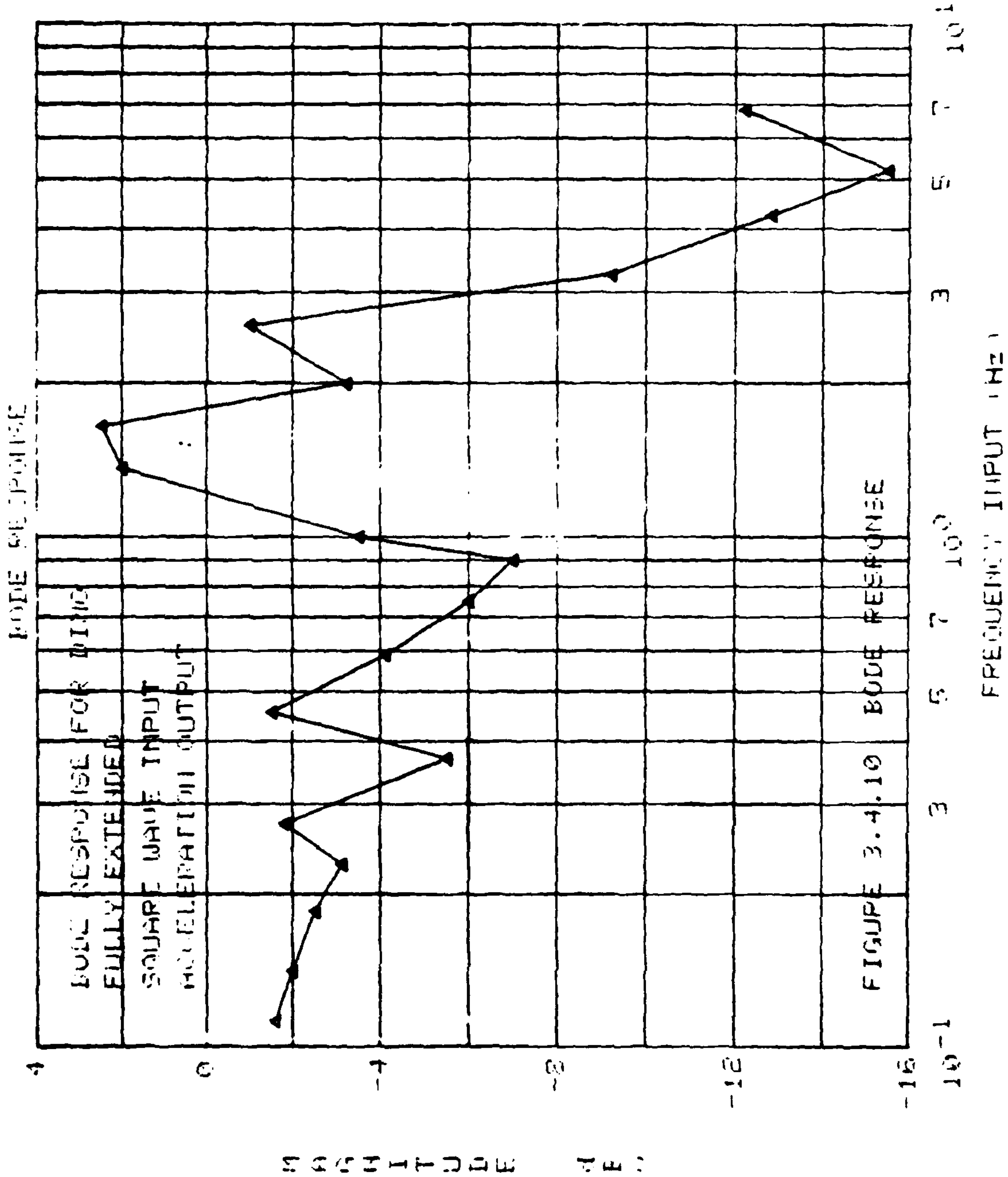


FIGURE 3.4.10 BODE RESPONSE

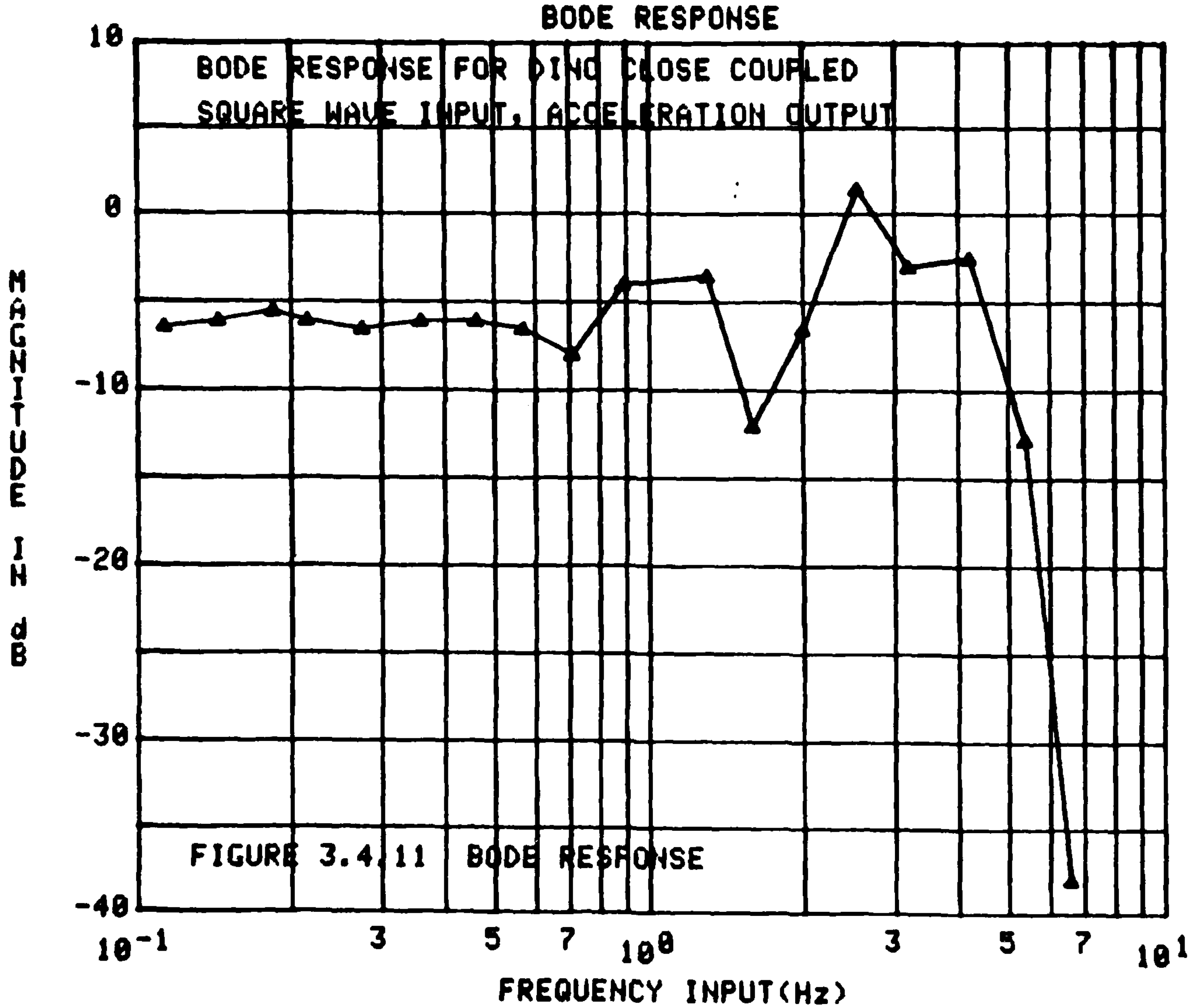


FIGURE 3.4.11 BODE RESPONSE

plot which most nearly agreed with the rules governing the model.

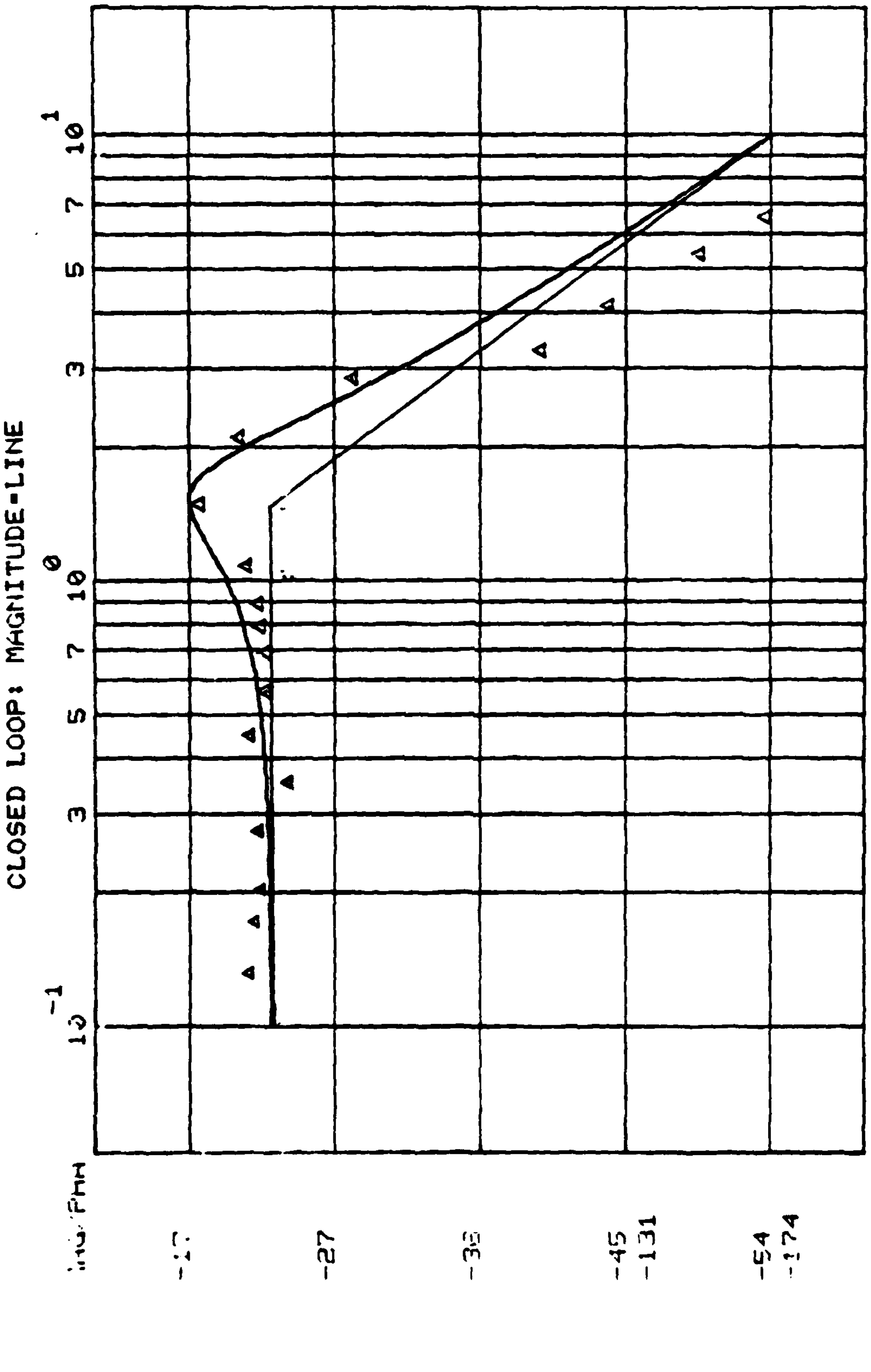
The first step in the analysis was to choose an approximate analytical Bode curve to approximate the experimental curve. Then an asymptotic approximation of the analytical Bode plot would lead to the transfer function. The experimental Bode plot chosen for analysis was that shown in Figure 3.4.6. Figure 3.4.12 shows the analytical Bode curve and the asymptotic approximation drawn through the data points of Figure 3.4.6. This plot exhibits the following characteristics: The final slope of the Bode plot was most nearly -40 dB/decade. The prominent resonant frequency occurred at approximately 1.3 Hz at a magnitude of approximately 4 dB above the low frequency asymptote. The low frequency response of the system indicated that there were no poles or zeros at the origin, and that the gain K was less than one.

By analysis of the analytic and asymptotic approximations as shown in Figure 3.4.12, it was found that a single complex pair of poles of the form

$$\frac{K}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \text{where } K < \omega_n^2$$

would yield Bode plots very similar to those found experimentally, if the proper values of ζ and ω_n were chosen.

Because of inaccuracies in the experimental Bode plots,


CLOSED LOOP: MAGNITUDE-LINE
-27
-45
-131
-54
-174
3
5
7
3
5
7

they were not used to actually determine the damping ratio (ζ) and the natural frequency (ω_n). Truxal's third general method of determining a transfer function, transient response analysis, was used to evaluate the constant values of the characteristic equation. [7]

3.5 Modifying the Dynamic Approach

Before discussing the transient approach, it might be helpful to explain how the Bode response simplification of the dynamic approach can be explained analytically. In the dynamic approach, the system was assumed to be made up of two significant inertial masses. If this assumption were changed, as the Bode analysis suggested, then only one inertial mass would satisfy the requirements of the lumped parameter system analysis. The transfer function would be derived using the model in Figure 3.5.1. The derivation of the dynamic transfer function with the new model is given below:

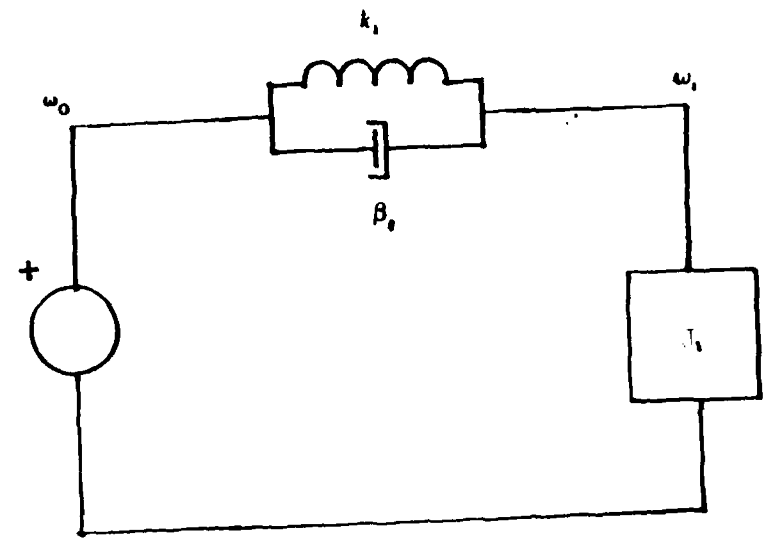
$$(\omega_0 - \omega_1)(\beta_1 + \frac{k_1}{s}) = \omega_1(sJ_1)$$

Therefore

$$\omega_0(\beta_1 + \frac{k_1}{s}) = \omega_1(\beta_1 + \frac{k_1}{s} + sJ_1)$$

The transfer function of the model is

$$\frac{\omega_1}{\omega_0} = \frac{\frac{\beta_1}{J_1}\left[s + \frac{k_1}{\beta_1}\right]}{s^2 + \frac{\beta_1}{J_1}s + \frac{k_1}{J_1}} \qquad 3\text{-}6$$



[illegible] 3.5.1 [illegible] MANIPULATOR

This analysis shows that, with the new assumption, the characteristic equation for velocity was of second order. If the zero was shown to be much farther to the left in the s-plane than the complex poles, then for the Bode analysis it could be ignored. This is shown below:

From 3-6

$$\frac{\beta_1}{J_1} = 2\zeta\omega_n \quad \text{and} \quad \frac{k_1}{J_1} = \omega_n^2$$

Therefore

$$\frac{\beta_1^2}{J_1^2} = 4\zeta^2 \frac{k_1}{J_1}$$

or

$$\beta_1 = 2\zeta\sqrt{k_1 J_1}$$

From 3-6 the zero is at $s + \frac{k_1}{\beta_1}$

$$\frac{k_1}{\beta_1} = \frac{k_1}{2\zeta\sqrt{k_1 J_1}} = \frac{1}{2\zeta}\sqrt{\frac{k_1}{J_1}} = \frac{\omega_n}{2\zeta}$$

If ω_n is assumed to be large and ζ is small, then the zero from equation 3-6 will be far to the left.

3.6 Transient Response Analysis

With a pure second order system as the model of the plant transfer function, the transient response approach was a convenient method of obtaining the necessary constants. The transient approach taken involved programming the transfer function of the model of the system on an analog computer

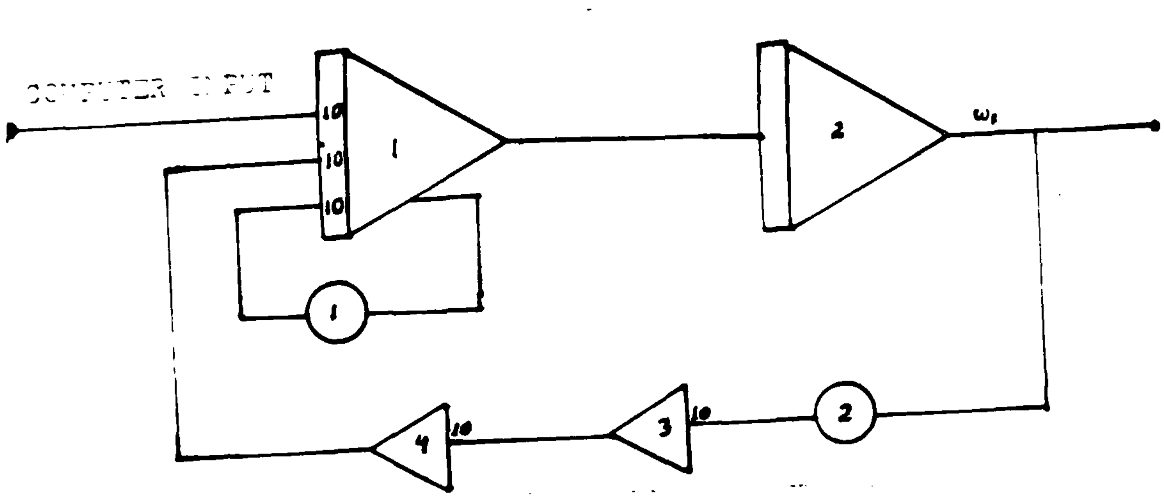


FIGURE 3.6.1 ANALOG PROGRAM OF MODEL [illegible]

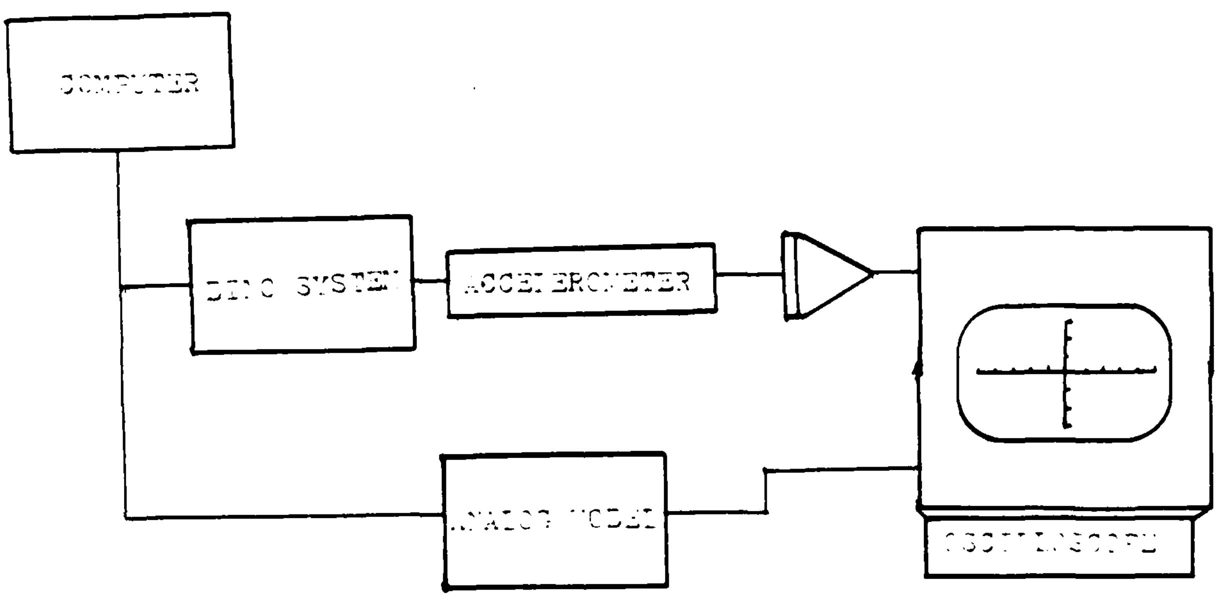


FIGURE 3. [illegible]

as shown in Figure 3.6.1. The desired constants of damping ratio and natural frequency were directly controlled by potentiometers, so that they could be varied throughout the analysis.

A unit step was then input simultaneously to the actual and modeled systems. The outputs of both systems were displayed on an oscilloscope, as shown in Figure 3.6.2. A visual comparison was made and the potentiometers of the analog program model were adjusted to fine tune the model output to closely approximate the output of the actual system. In this way, system constants were determined.

3.7 Regions of Motion

The values of the system constants did not remain constant throughout the entire range of motion. Instead, they varied as a function of the radial distance between the manipulator and effector and the axis of rotation. It was not obvious which form this function took, but there were regions of motion where the transfer function remained fairly constant. Assuming that the compensation method chosen would be sufficiently robust to compensate for minor changes in the location of roots of the characteristic equation, regions of motion were defined where the roots of the characteristic equation would be considered constant. This eliminated the need to define governing equations to describe how the roots would change as a function of end effector positioning.

There were two major regions chosen to separate the range of motion of the manipulator. The first, and the one which most closely resembled the pure second order model, was the fully extended region. This region was defined to be that region where either of the pistons which controlled the shoulder or elbow joints was extended outward beyond the midpoint of its scope. Any position which was not fully extended was considered to be close-coupled. Figure 1.2 shows the system in one of its close-coupled positions. Figure 3.3.1 shows a fully extended position, because the pistons which control the motion about the shoulder hinge are extended beyond the midpoint of their scope.

CHAPTER FOUR

COMPENSATOR DESIGN

The goal of this project was to enhance the performance of a mechanical robot arm driven by bang-bang actuators. System performance was only inadequate in the horizontal plane of motion, so it was this motion that was investigated. A model of the system equations of motion in the horizontal plane was derived using analytic and experimental methods. The transfer function took the following form:

$$\frac{K}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

The system input, as discussed earlier, could not automatically be considered constant. An assumption about the nature of the desired input had to be made. Since the compensation was designed to be deadbeat, it would be completed within one period of the natural frequency. With this in mind, it was possible to assume that cosecant (ϕ), from equation 3-2, would be constant. Therefore, the input function could be considered constant. A constant, bang-bang input would take the following form:

$r(t) = u(t-t_1) + u(t-t_2) - u(t-t_3) + \ldots\ldots$

Where $t_1 < t_2 < t_3 < \ldots < t_n$

$u(t)$ is a constant step input applied for $t > 0$

The object was to control the system with as simple an input as possible. The least complicated input was $u(t)$ (a constant step input). Figure 4.1 shows the output of a typically underdamped system with $u(t)$ as the input. The sample system dynamics were chosen to be

$$\frac{Y(s)}{R(s)} = \frac{100}{s^2 + s + 100}$$

As the case with flexible robot manipulators, the system was very underdamped, and a simple input, $u(t)$, would not suffice as a controlling input.

The second least complicated input was $u(t) - u(t-t_1)$. This was undesirable as a control because, assuming t_1 was finite, the steady state value of the output was zero.

The third least complicated input was $u(t) - u(t-t_1) + u(t-t_2)$. This input would yield the proper steady-state value for a control, and it also allowed for t_1 and t_2 to be varied independently. It was supposed that this control could be manipulated to compensate the system.

The first step to evaluate a compensation strategy based on the control above, was to look at system response to an input of $r(t) = u(t) - u(t-t_1)$.

$$\frac{Y(s)}{R(s)} = \frac{K}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad \text{assume } K = \omega_n^2$$

$$R(s) = \frac{1 - e^{-st_1}}{s}$$

$$Y(s) = \frac{\omega_n^2 (1 - e^{-st_1})}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} \qquad 4\text{-}1$$

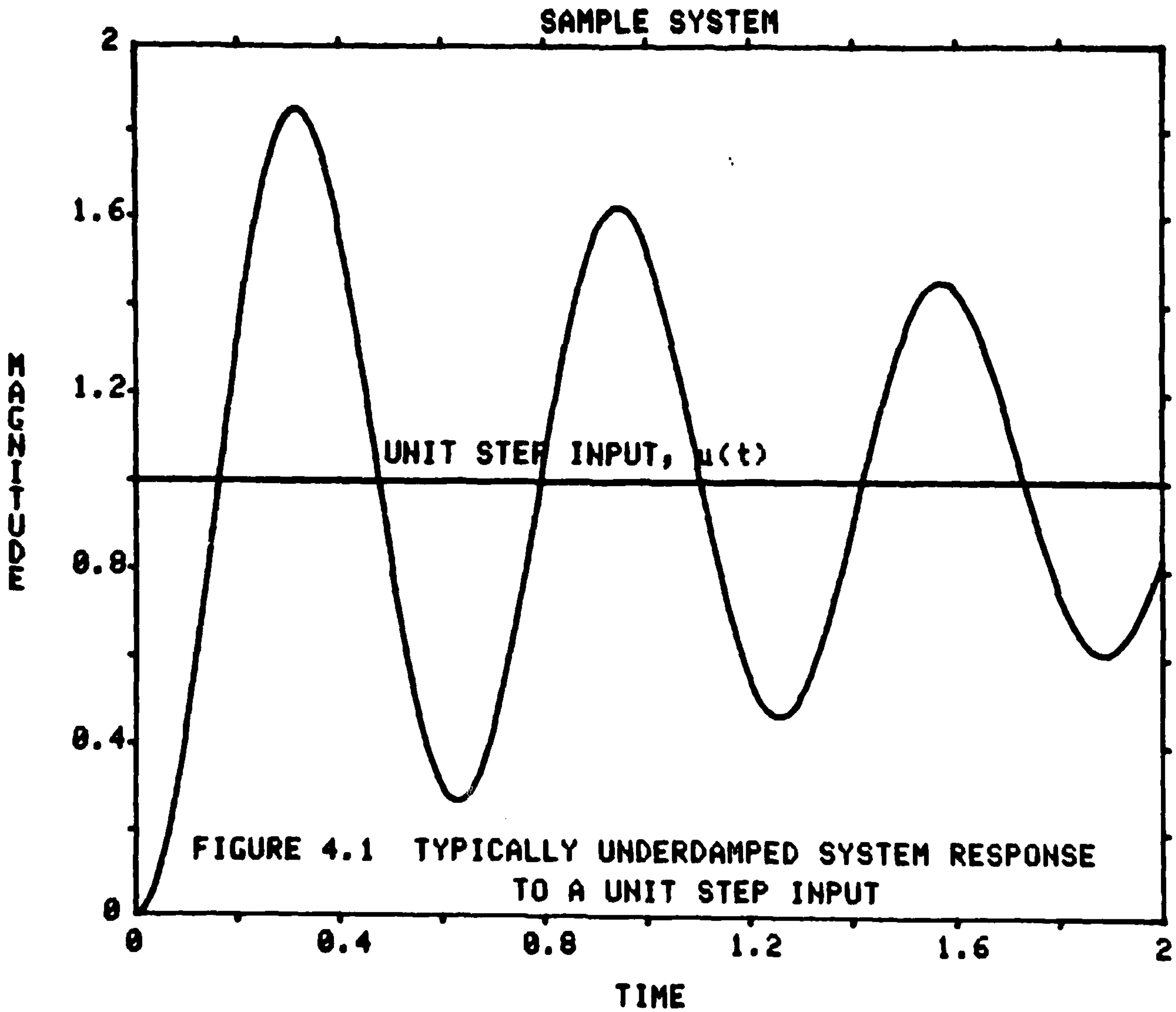


FIGURE 4.1 TYPICALLY UNDERDAMPED SYSTEM RESPONSE TO A UNIT STEP INPUT

Take inverse Laplace transform of 4-1

$$Y(t) = \left[1 + \frac{1}{\sqrt{1-\zeta^2}} e^{-\sigma t} \sin(\omega_d t - \phi)\right] u(t)$$

$$- \left[1 + \frac{1}{\sqrt{1-\zeta^2}} e^{-\sigma(t-t_1)} \sin(\omega_d (t-t_1) - \phi)\right] u(t-t_1) \quad 4\text{-}2$$

where $\sigma = \zeta\omega_n$

$\omega_d = \omega_n\sqrt{1-\zeta^2}$

$\phi = \tan^{-1}\left(\frac{\sqrt{1-\zeta^2}}{-\zeta}\right)$

For the fastest response, the control was to become effective during the first period, $t<(\frac{2\pi}{\omega_n})$. Figure 4.2 shows how the sample system reacted to varying t_1. The most significant point of each curve was t_p, the time of the first peak overshoot. It was at this point that the system had an acceleration of zero and constant velocity. If, at this point, the velocity was equal to the steady-state velocity, then a step input $u(t-t_2)$, where $t_2 = t_p$, it was believed, would maintain this velocity with zero acceleration. Thus, a compensator with a control of $u(t) - u(t-t_1) + u(t-t_2)$, where $t_2 = t_p$ and $y(t_2)$ equaled the steady state velocity, would bring the system to steady state velocity as fast as possible with no overshoot. A mathematical analysis showed the following:

To find t_p, assume $t > t_1$ in equation 4-2.

Thus

$$Y(t) = Ae^{-\sigma t} \sin(\omega_d t - \phi) - Ae^{-\sigma(t-t_1)} \sin(\omega_d (t-t_1) - \phi) \quad 4\text{-}3$$

where $A = \frac{1}{\sqrt{1-\zeta^2}}$

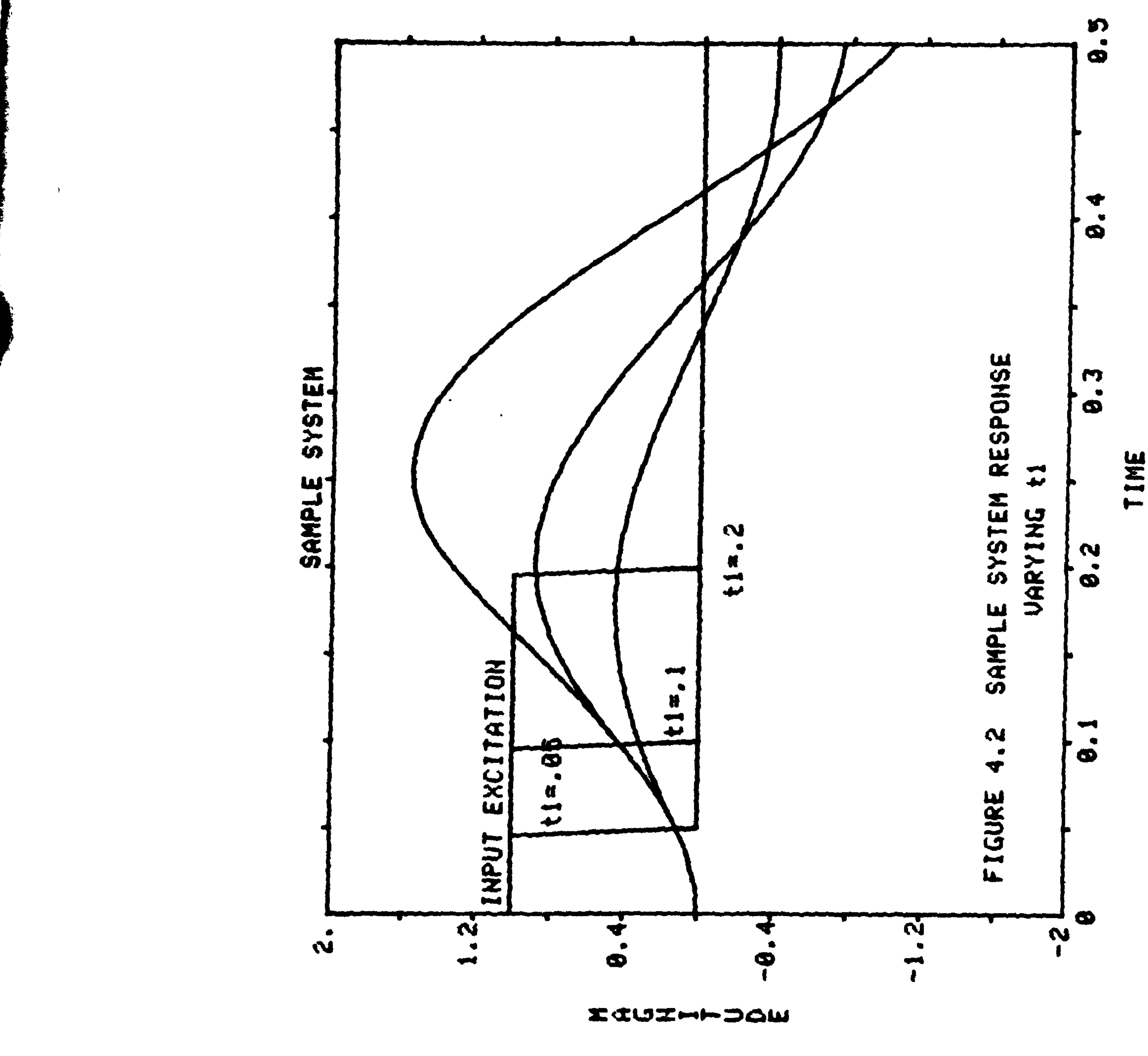


FIGURE 4.2 SAMPLE SYSTEM RESPONSE VARYING t1

Then take the time derivative of 4-3.

$$\frac{dy(t)}{dt} = \omega_d A e^{-\sigma t} \cos(\omega_d t - \phi) - \sigma A e^{-\sigma t} \sin(\omega_d t - \phi) \qquad 4\text{-}4$$

$$- \omega_d A e^{-\sigma(t-t_1)} \cos(\omega_d(t-t_1)-\phi) + \sigma A e^{-\sigma(t-t_1)} \sin(\omega_d(t-t_1)-\phi)$$

Set equation 4-4 equal to zero to find t_p.

To do this

Assume t_1 = constant

and expand sin and cos terms

$$\cos(\alpha - \beta) = \cos\alpha \cos\beta + \sin\alpha \sin\beta$$

$$\sin(\alpha - \beta) = \sin\alpha \cos\beta - \cos\alpha \sin\beta$$

then solve for t_p

$$t_p = \frac{\tan^{-1}\left[\dfrac{-\omega_d\cos(\phi)-\sigma\sin(\phi)+\omega_d e^{\sigma t_1}(\omega_d t_1+\phi)+\sigma e^{\sigma t_1}\sin(\omega_d t_1+\phi)}{\omega_d\sin(\phi)-\sigma\cos(\phi)-\omega_d e^{\sigma t_1}\sin(\omega_d t_1+\phi)+\sigma e^{\sigma t_1}\cos(\omega_d t_1+\phi)}\right]}{\omega_d} \qquad 4\text{-}5$$

Thus t_p, given the damping ratio and the natural frequency of the system, is a function of t_1. Substituting t_1 and t_p into equation 4-3 gives the magnitude of the response at t_p. A digital program which varied t_1 would be able to find t_p, where $y(t_p)$ was equal to the steady-state velocity. Therefore, given the damping ratio and the natural frequency, t_1 and t_2 could be found for the input which would act as a deadbeat compensator. Figure 4.3 shows that the implementation of this control to the sample system yielded a deadbeat response.

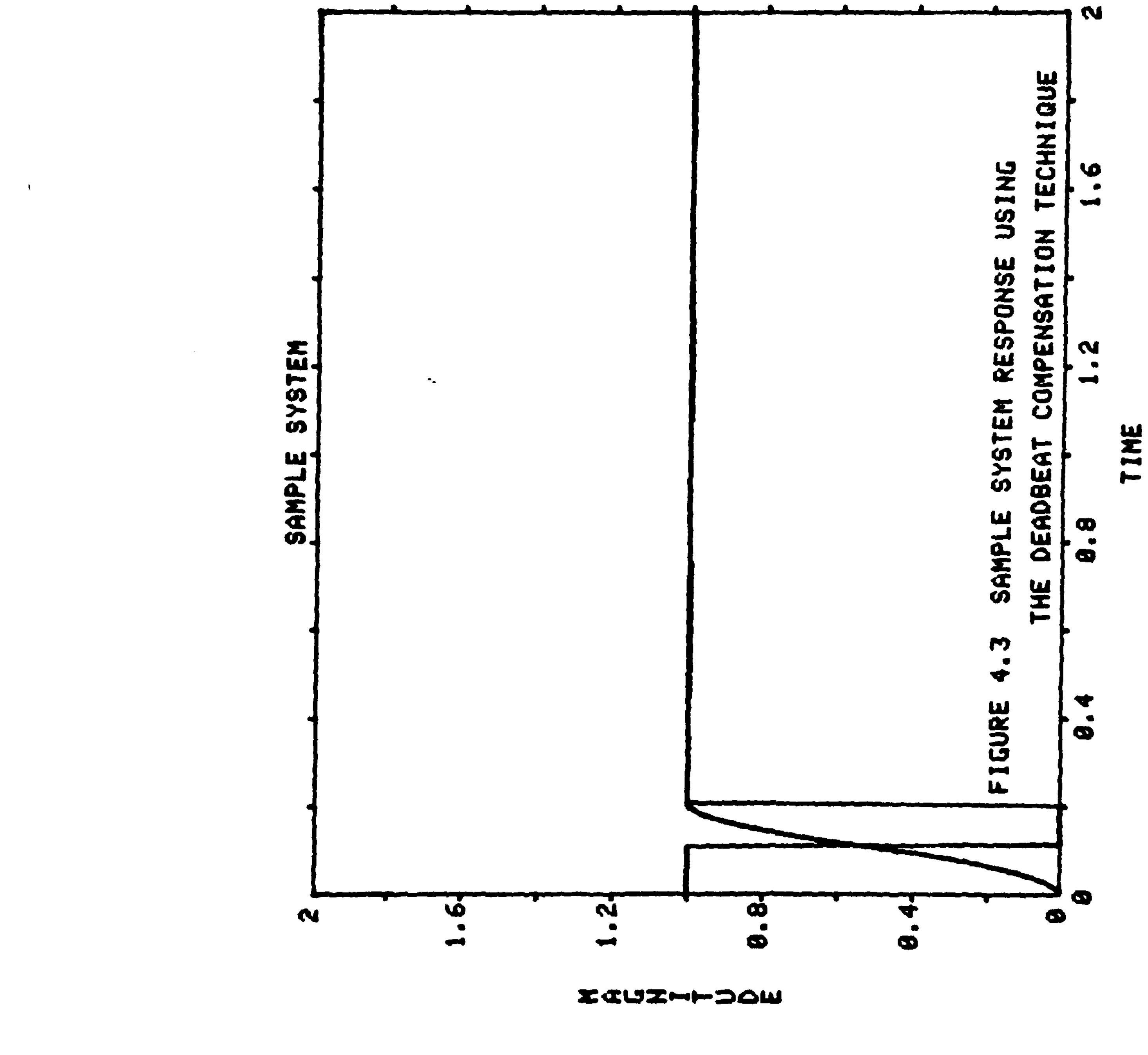


FIGURE 4.3 SAMPLE SYSTEM RESPONSE USING THE DEADBEAT COMPENSATION TECHNIQUE

CHAPTER FIVE

SENSITIVITY OF DEADBEAT CONTROLLER

The purpose of this chapter is to examine the sensitivity of the deadbeat controller to changes in the system parameters. The deadbeat controller relies heavily on the knowledge of system parameters. If the system is second order and bang-bang actuated, then it was shown that a simple deadbeat controller may be programmed which will compensate the system if the system parameters of damping ratio and natural frequency are known. What will happen, though, if the system parameters vary from the modeled values from which the deadbeat compensator was derived? The effect of changing system parameters on the ability of a compensator to control the system is known as sensitivity.

Deadbeat controllers have a reputation for being very sensitive to parameter changes. By varying the damping ratio and natural frequency of the sample system without changing the compensator, the sensitivity of this deadbeat technique can be evaluated.

Figures 5.1 through 5.6 show how the output is affected by changing the damping ratio and the natural frequency while keeping the compensating times constant. It can be seen that the output is indeed affected by parameter changes, but system performance is still improved, as compared with the uncompensated system.

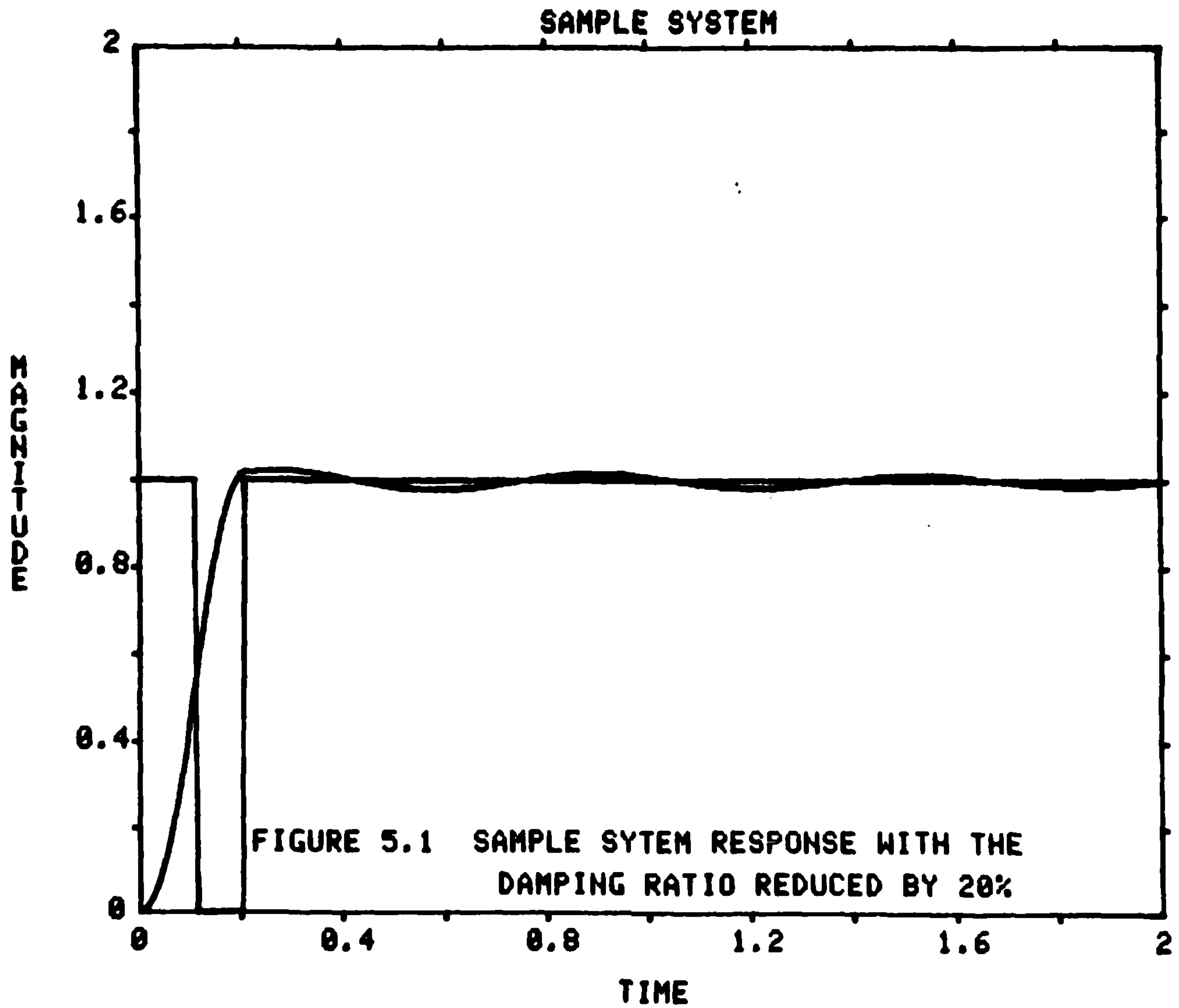


FIGURE 5.1 SAMPLE SYTEM RESPONSE WITH THE DAMPING RATIO REDUCED BY 20%

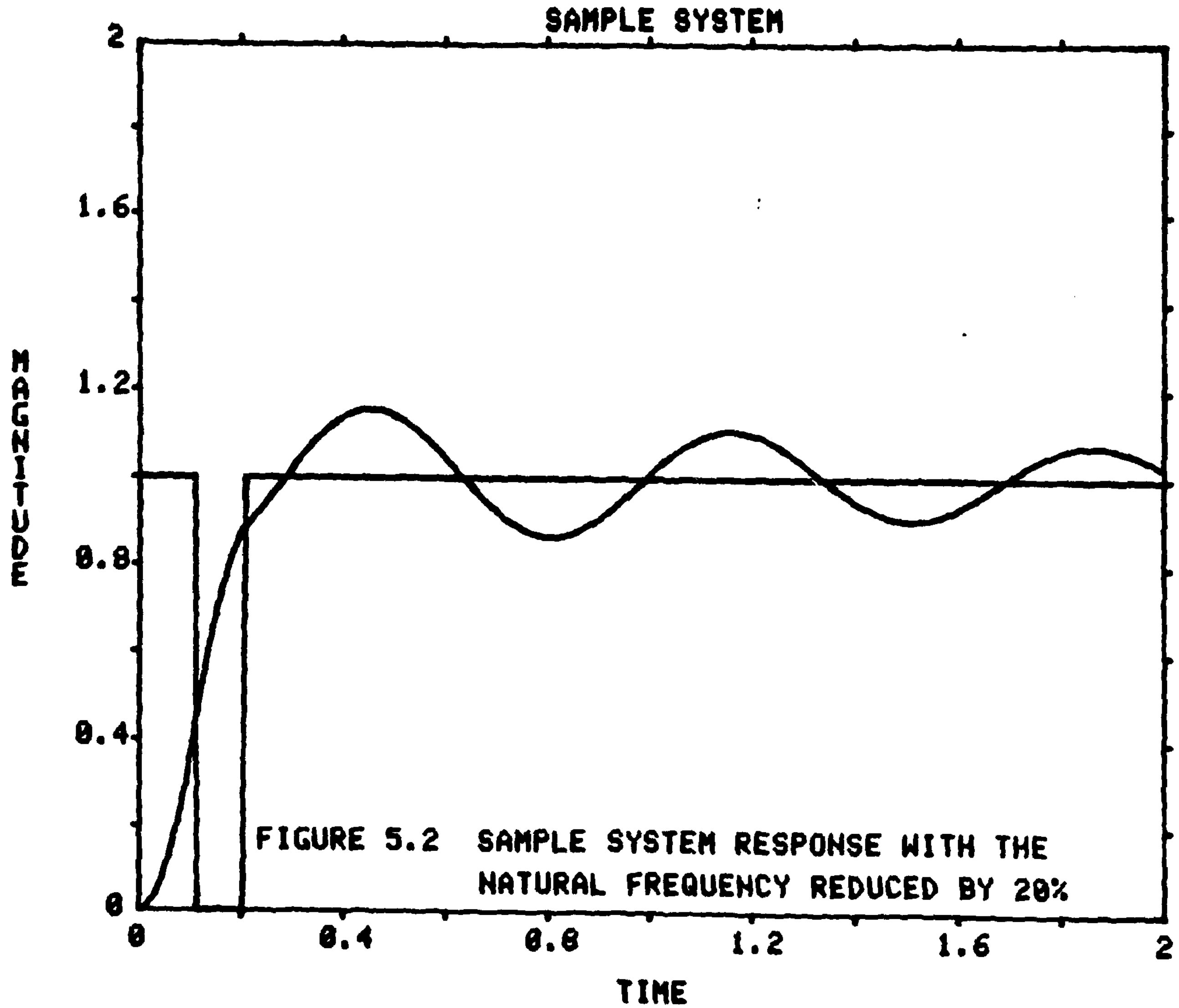


FIGURE 5.2 SAMPLE SYSTEM RESPONSE WITH THE NATURAL FREQUENCY REDUCED BY 20%

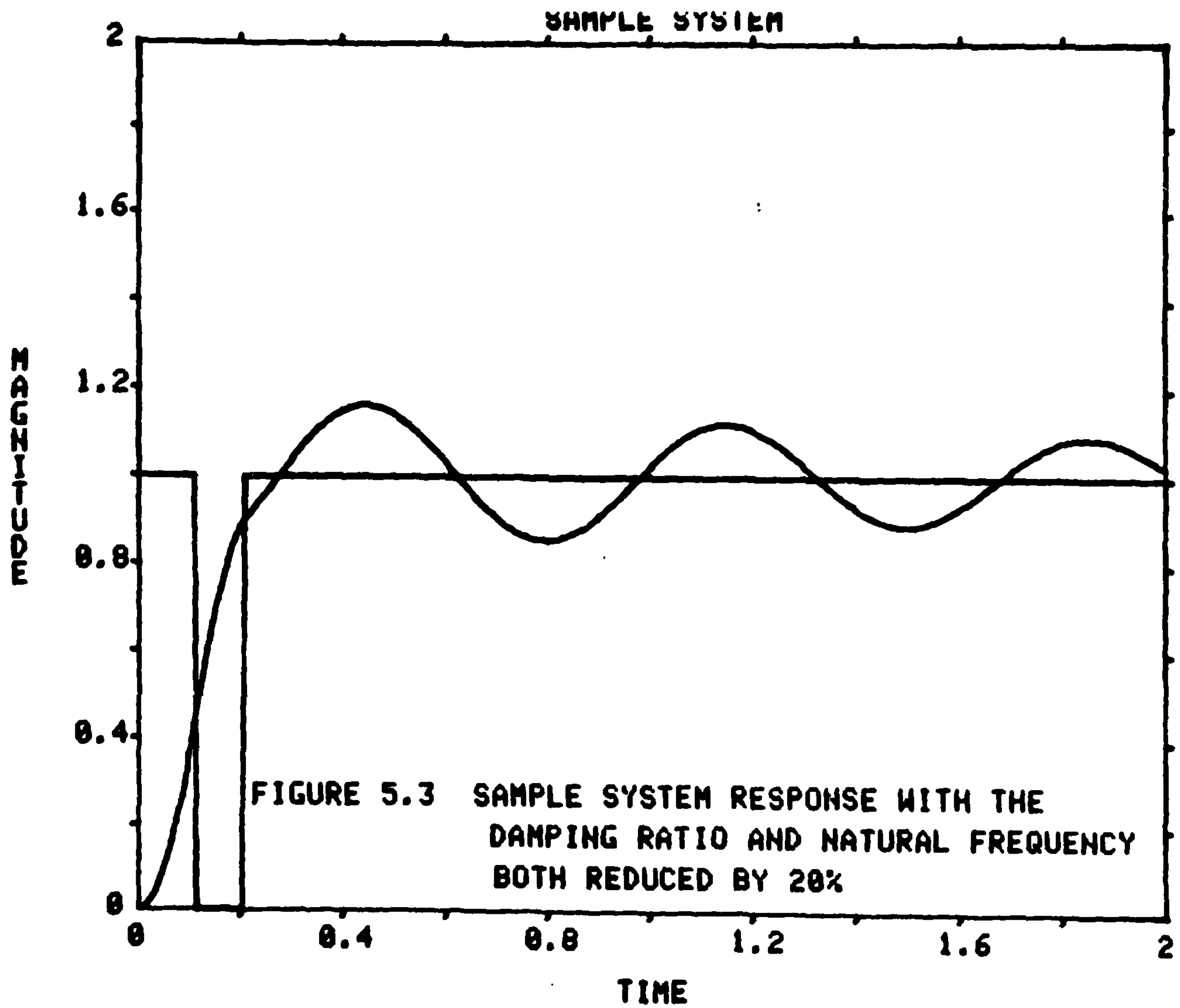


FIGURE 5.3 SAMPLE SYSTEM RESPONSE WITH THE DAMPING RATIO AND NATURAL FREQUENCY BOTH REDUCED BY 20%

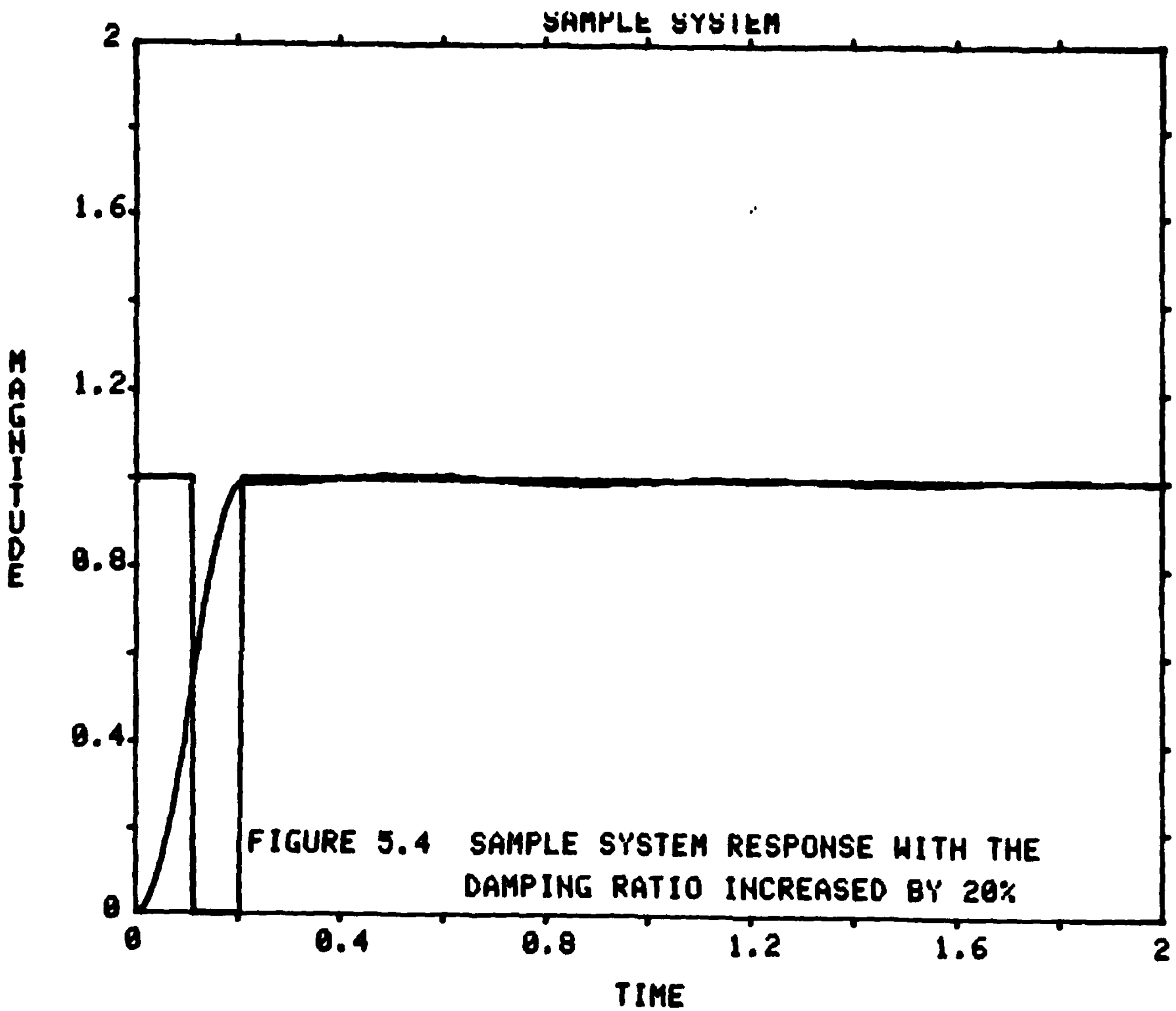


FIGURE 5.4 SAMPLE SYSTEM RESPONSE WITH THE DAMPING RATIO INCREASED BY 20%

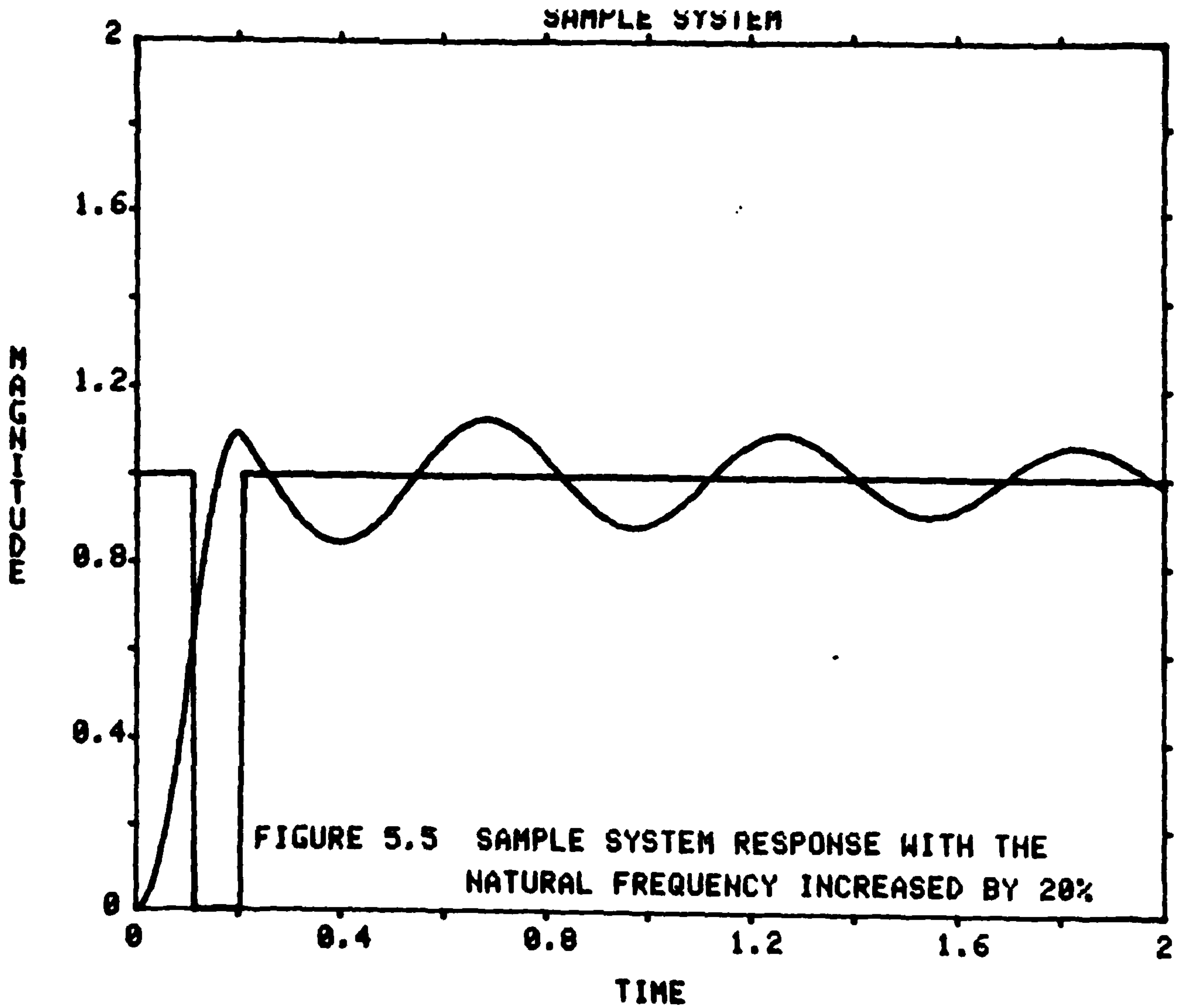


FIGURE 5.5 SAMPLE SYSTEM RESPONSE WITH THE NATURAL FREQUENCY INCREASED BY 20%

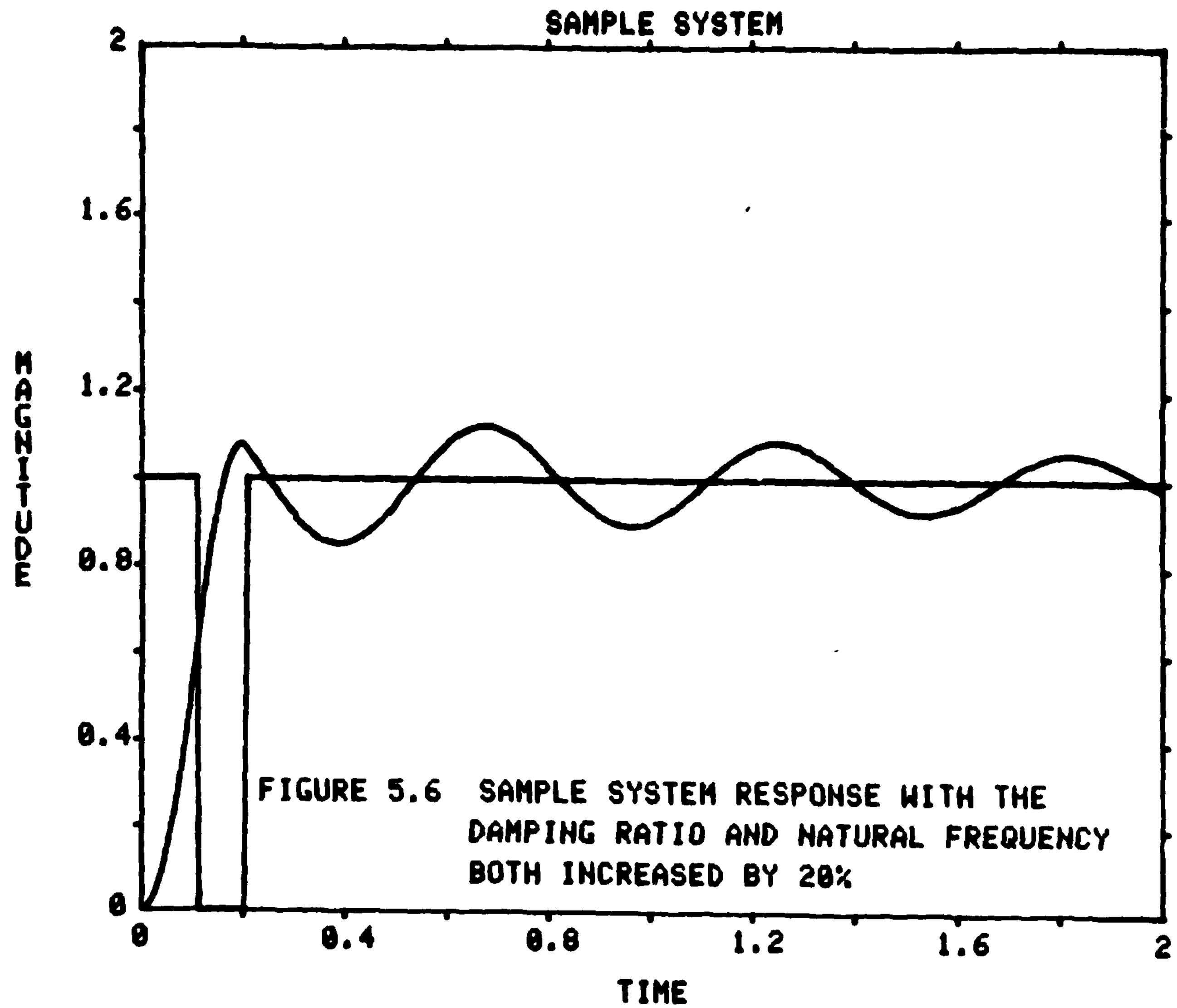


FIGURE 5.6 SAMPLE SYSTEM RESPONSE WITH THE DAMPING RATIO AND NATURAL FREQUENCY BOTH INCREASED BY 20%

In Chapter Three it was decided to choose regions of motion where the roots of the characteristic equation would be considered constant. This decision was based on the assumption that the compensation technique which would be developed would be relatively insensitive to minor parameter changes exhibited within the region. The study of sensitivity in this section has shown that this assumption was valid.

The effectiveness of the deadbeat controller also relies heavily on the ability of the computer to precisely measure the control times, t_1 and t_2. The times, t_1 and t_2, are the input times discussed in Chapter Four. If the computer was not able to achieve good resolution of t_1 and t_2, then the compensation would be adversely affected. Figures 5.7 through 5.9 show how the output of the system is affected by varying resolution of t_1 and t_2. Again, it can be seen that the output is affected by changing resolution of t_1 and t_2.

The actual resolution available for this project was dependent on how fast the microcomputer could run the controlling programs. In the programs which controlled the system, time was measured using program loops of the form

```
100  FOR I = 1 TO A
110  NEXT I
```

The resolution was a function of how fast the computer could run the loop when A = 1. It was found, by using

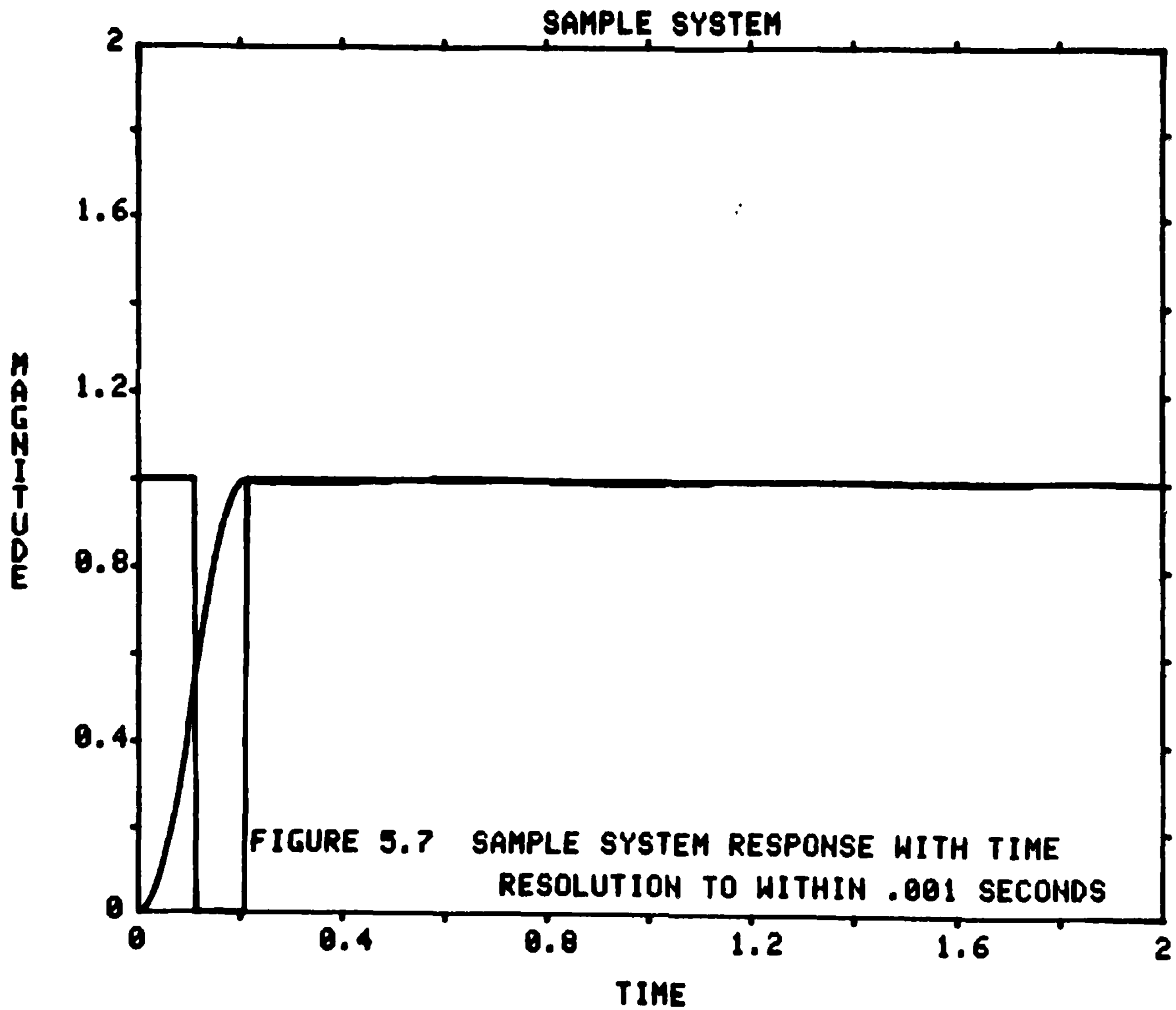


FIGURE 5.7 SAMPLE SYSTEM RESPONSE WITH TIME RESOLUTION TO WITHIN .001 SECONDS

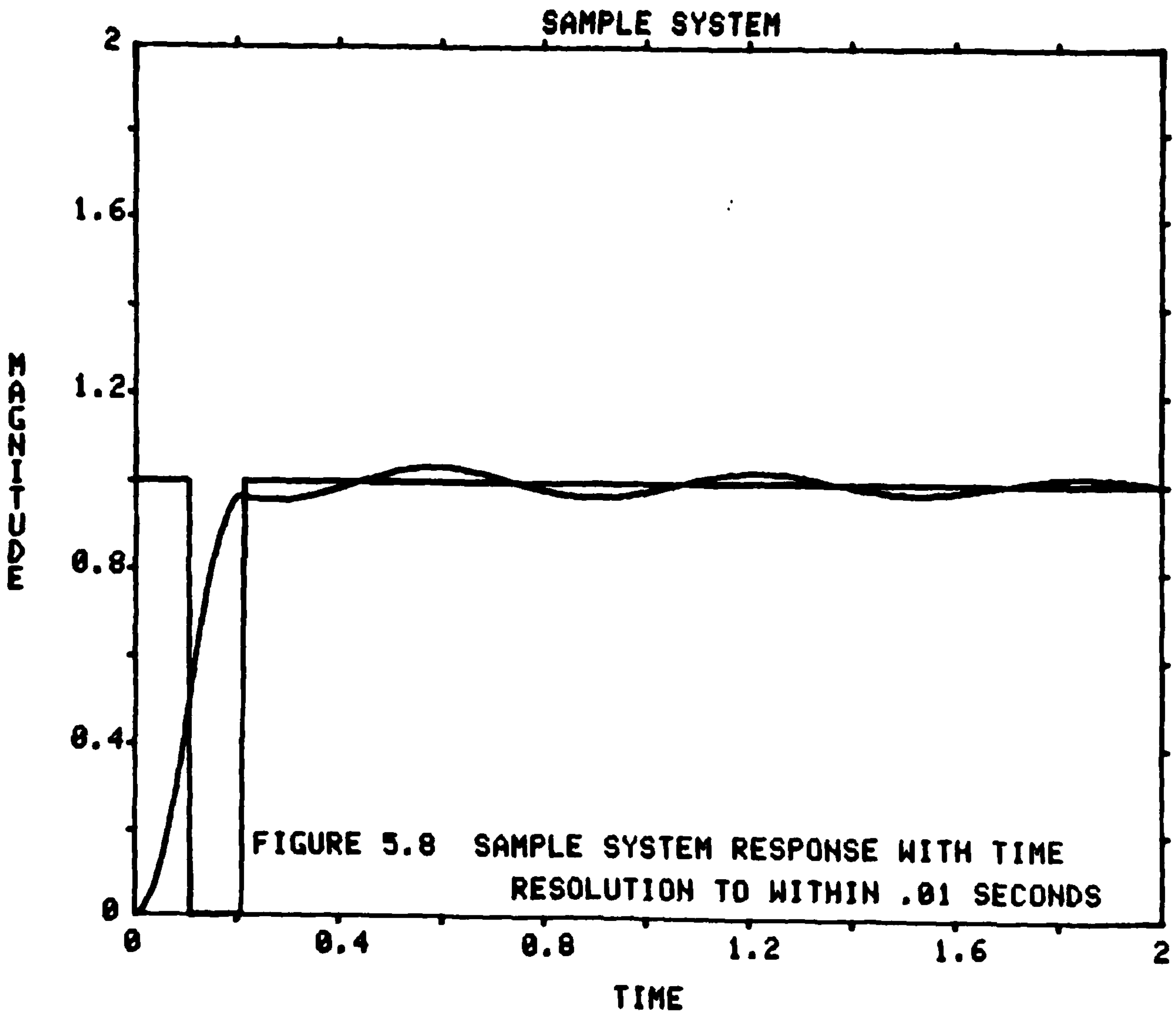


FIGURE 5.8 SAMPLE SYSTEM RESPONSE WITH TIME RESOLUTION TO WITHIN .01 SECONDS

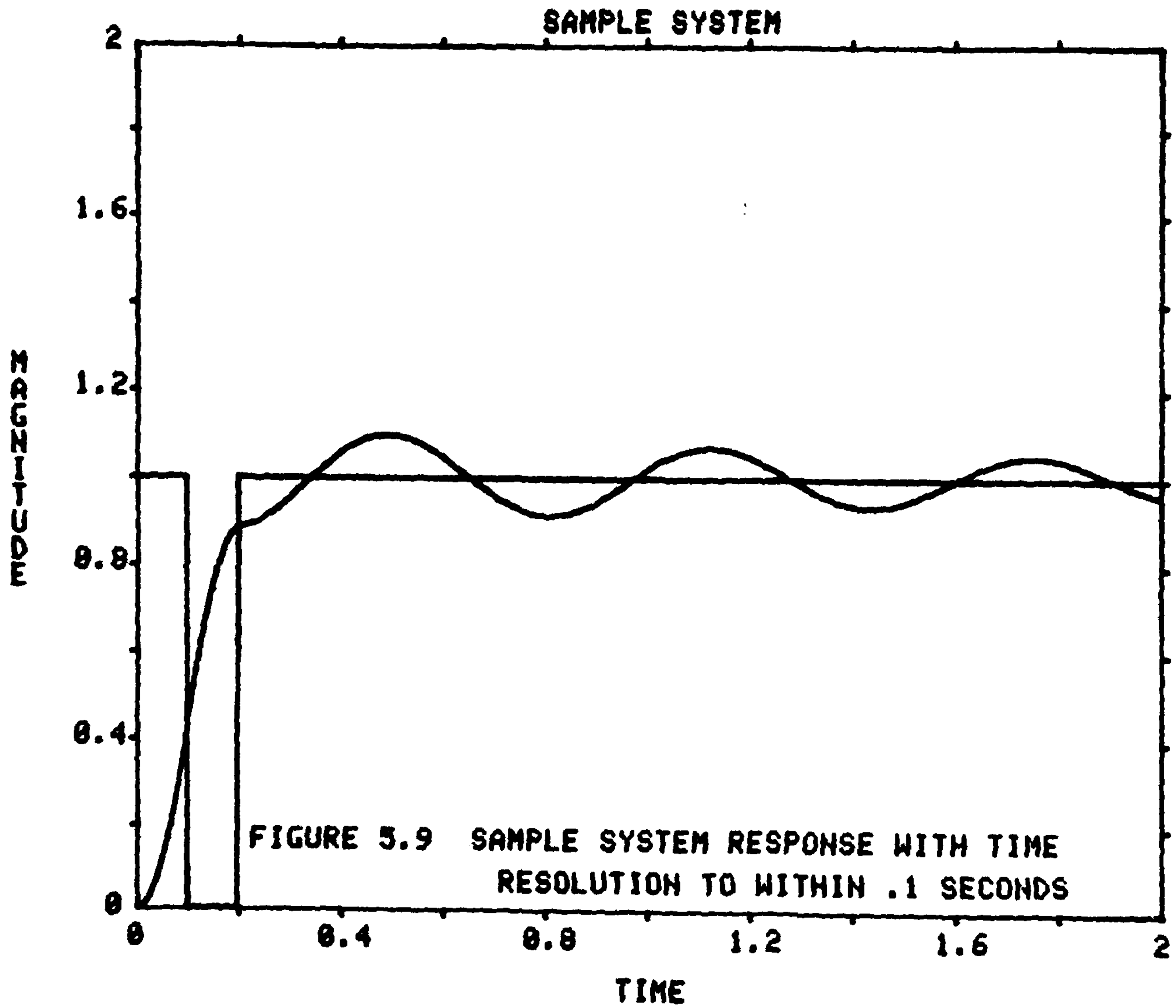


FIGURE 5.9 SAMPLE SYSTEM RESPONSE WITH TIME RESOLUTION TO WITHIN .1 SECONDS

the "SQBODE" program found in Appendix C and a period measuring device, that resolution of approximately 1.2 milliseconds could be achieved. Figure 5.7 shows that this resolution is quite sufficient.

CHAPTER SIX
IMPLEMENTATION

This chapter introduces the programs which were used to compensate the horizontal motion of the system. The practical problems associated with implementing the theoretical compensation technique developed in Chapter Four on the actual system are also discussed. Finally, this chapter studies the effectiveness of the compensator.

6.1 Computer Programs

For testing the control algorithm, the program "CONTROL," shown in Appendix A, was used. This program allowed the control times discussed in Chapter Four to be changed. The investigator could also input the direction of motion to be tested. The program only controlled motion in the horizontal plane. Different orientations of the shoulder, elbow and wrist joints had to be chosen and prepared manually before the program was run.

A more powerful program "TEACH," shown in Appendix B, was not used for testing, but for demonstration. This program allowed the user to input, or teach, a complete routine to the system. This program would then repeat the

the routine as many times as instructed using the compensation technique of Chapter Four. The program takes into account the region of motion to be compensated. Only the motion in the horizontal plane was compensated.

6.2 Practical Implementation

In Chapters Four and Five, it was shown that the deadbeat compensator for second order systems controlled by bang-bang actuators was an effective method of compensating highly underdamped second order systems. Implementation of the theory would work as long as the model used to simulate the system was an accurate estimation of the actual system. For the fully extended case, as defined previously, the control was implemented per theory, with the expectation of the following change to the controlling times. The input pulse widths calculated using the deadbeat criteria were not the same times used to control the horizontal motion of the actual system. This was due to the necessity to take into account a mechanical time lag not accounted for in the model. This time lag, due to a signal delay at the actuator and pistons which controlled motion in the horizontal plane, was simply added to the pulse width times to make the compensation effective. The improvement in system performance was substantial after this time lag was considered. The strip chart recordings in Figures 6.2.1 to 6.2.4 show the output with and without compensation.

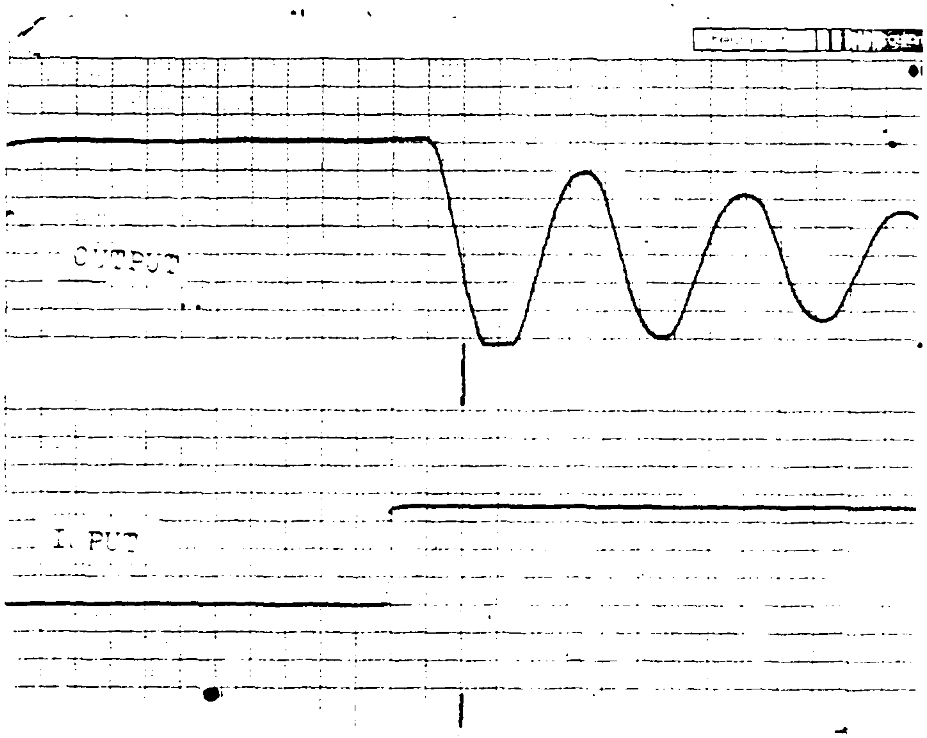


FIGURE 6.2.1 SYSTEM RESPONSE WITH NO COMPENSATION IN THE FULLY-EXTENDED REGION

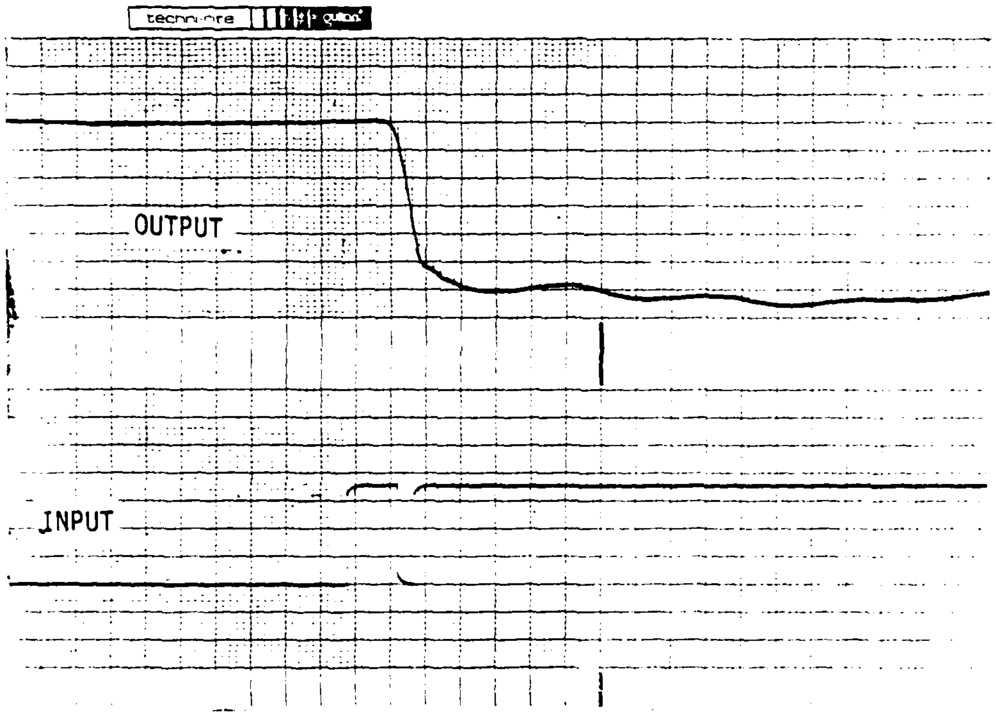


FIGURE 6.2.2 SYSTEM RESPONSE WITH COMPENSATION IN THE FULLY-EXTENDED REGION

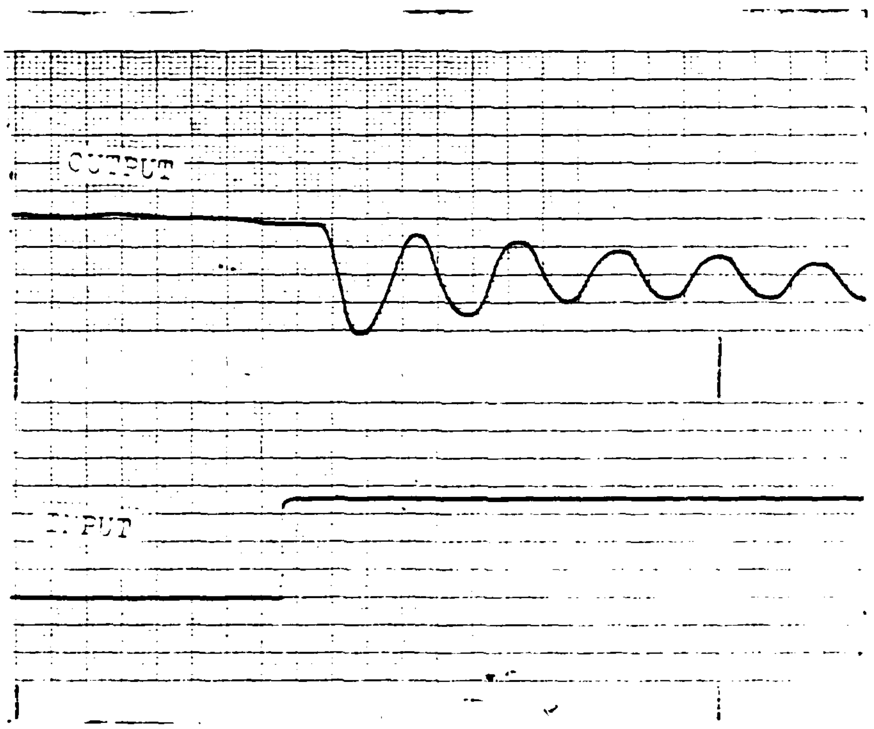


FIGURE 6.2.3 SYSTEM RESPONSE WITH NO COMPENSATION
IN THE CLOSE-COUPLED REGION

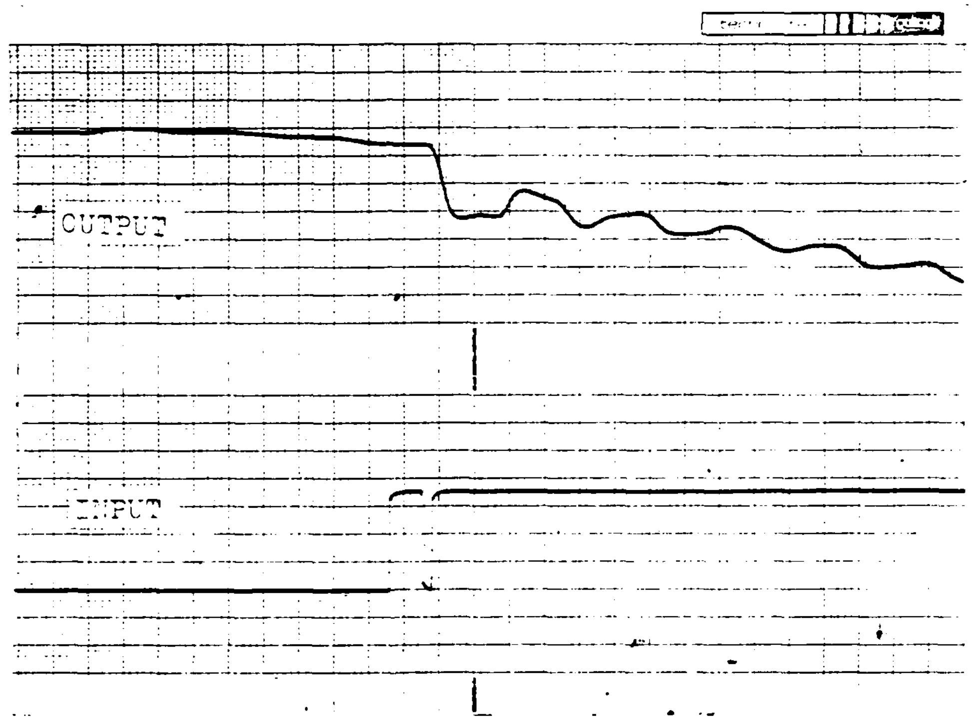


FIGURE 6.2.4 SYSTEM RESPONSE WITH COMPENSATION IN THE CLOSE-COUPLED REGION

The data for the strip chart recordings was taken at the center of the region to be compensated. It was found that system improvement for the fully extended region was substantial throughout the entire region when using the set of control values derived for the center of the region. The values used for the fully extended region were found to be mostly ineffective when applied to the close-coupled region, thus making it necessary to have a different set of control values for each region of motion.

The close-coupled case was also recorded by strip chart. (Figures 6.2.3 and 6.2.4) Again, the output was improved with compensation, but the compensated response was not perfectly deadbeat. A reason suspected for this situation was the breakdown of the initial linear assumptions of the model. Greater sensitivity due to slight parameter variations of the end effector positioning in the close-coupled region was also found to be a factor which influenced the imperfect response.

The simplified linear model most nearly matched the actual system in the fully extended region, and it was in this region that compensated system response was most nearly deadbeat. Nevertheless, as shown in Figure 6.2. a theoretical deadbeat response was not completely achieved. There were three possible contributing reasons for the minor differences which were observed between the actual and theoretical responses. The first reason was that the model was not completely accurate due to faulty linearity and lumped system assumptions. A second explanation for the deviation of the fully extended response from theoretical deadbeat response was that the assumption of a constant input was faulty. This was not considered a major factor because, as discussed earlier, the angular displacement during the time of the control was so small that the input excitation could be considered constant. The last, and most probable, reason for the imperfect response was the affect of the system zero shown in equation 3-6. This zero had been assumed to be far enough to the left to be overlooked. The deadbeat control algorithm was developed without consideration of this zero.

CHAPTER SEVEN

FURTHER IMPROVEMENTS

System improvement was significant throughout the range of motion of the manipulator, but the performance of the compensation technique could be improved further. In the opinion of the investigator, the assumptions made in the process of determining the transfer function were both helpful and valid. To make the model more complicated would not have facilitated development of a more cost-effective compensation technique. The system was controlled by a simple input; therefore, a simple model and a simple compensator were warranted.

There are ways to improve upon the results of this project. The controlling program used fixed pulse widths to control the manipulator within a given, predefined region. This meant that it was not adaptable to changing system parameters. A more powerful type of control would be to give the computer the ability to change its compensating algorithms based on changes in system parameters. This is known as adaptive control. If a compensator is not adaptive, then every time a parameter is significantly changed, such as the end effector load, the compensator will no longer be effective.

Adaptive control relies on state feedback information to compensate the system. For this system, there are two approaches which could lead to an effective adaptive controller, assuming that the general method of control would remain the same. In each case, a finite number of states would have to be fed back to the computer so that an algorithm could be applied to vary the compensation of the system.

The first approach would be to develop a more accurate model of the system which would take into account as many states as possible, such as oil pressure, air pressure, oil temperature, and end effector load. It would then be necessary to feed back these states so that the computer could evaluate the controlling times before activating the deadbeat controller. All of these calculations would be completed, and the control times set, before the compensation would take effect. The first approach would use a decision process which would be entirely outside the actual time of control.

The second approach for developing an effective adaptive controller would be to put all of the decision processes inside the time of control. In this case, t_1 and t_2 from Chapter Four would be considered functions of the states associated with motion, such as position, velocity, and acceleration. The feedback then would be used in real time, after the deadbeat controller had been activated, but before the first control time, t_1. An algorithm would use the feedback states to tell the computer when t_1 and t_2 should occur.

If a more complicated controlling technique is desired, then an adaptive controller based on a new compensation technique is possible. The system can be compensated by controlling a more complicated input pulse train, but as the pulse train becomes more complicated, so does the mathematical analysis and, more importantly, so does the physical implementation.

Non-linear digital control is also possible. This approach would be based on an assumption of system non-linearity due to saturation when pulse width exceeds sample time. The analysis for this approach is very complicated, and the literature which deals with this topic is sparse and somewhat dated.

FOOTNOTES

[1] Jasia Reichardt, Robots, (New York: Penguin Books, 1978), p. 7.

[2] Charles M. Close, and Dean K. Frederick, Modeling and Analysis of Dynamic Systems, (Boston: Houghton Mifflin Company, 1978), pp. 10-11.

[3] Close, pp. 5-6.

[4] John Truxal, Control System Synthesis, (New York: McGraw-Hill Book Company, Inc., 1955), pp. 344-45.

[5] Truxal, pp. 349-50.

[6] Truxal, pp. 350-53.

[7] Truxal, pp. 344-45.

BIBLIOGRAPHY

Close, Charles, M. and Dean K. Frederick. Modeling and Analysis of Dynamic Systems. Boston: Houghton Mifflin Company, 1978.

Reichardt, Jasia. Robots. New York: Penguin Books, 1978.

Truxal, John G. Control System Synthesis. New York: McGraw-Hill Book Company, Inc., 1955.

APPENDIX A

Compensator Test Program

"CONTROL"

This program was used to test the compensation technique. The control times are user inputs.

B:CONTROL.BAS 5-2-83

```
5 A% = 1
10 NUM0% = 0
20 NUM1% = 1
30 CALL PORTS
35 CALL CLRALL
40 PRINT "ENTER DIRECTION 1 OR 2"
50 INPUT D%
60 PRINT "ENTER T1"
70 INPUT L
80 PRINT "ENTER T2"
90 INPUT M
95 PRINT "ENTER N"
96 INPUT N
97 PRINT "ENTER T3"
98 INPUT O
99 PRINT "ENTER T4
100 INPUT P
101 CALL ENABLE(NUM1%)
110 CALL AXIS(A%,D%)
120 GOSUB 180
130 CALL AXIS(A%,NUM0%)
140 GOSUB 200
150 CALL AXIS(A%,D%)
151 GOSUB 300
153 CALL AXIS(A%,NUM0%)
154 GOSUB 400
155 CALL AXIS(A%,D%)
160 GOSUB 500
163 CALL AXIS(A%,NUM0%)
170 GOTO 40
175 END
180 FOR I = 1 TO L
190 NEXT I
195 RETURN
200 FOR J = 1 TO M
210 NEXT J
220 RETURN
300 FOR I = 1 TO N
310 NEXT I
320 RETURN
400 FOR I = 1 TO O
410 NEXT I
420 RETURN
500 FOR I = 1 TO P
520 NEXT I
530 RETURN
```

APPENDIX B

Compensator Deomonstration Program

"TEACH"

This program implements the compensation technique in a program designed for practical application. There are two options available. The user may choose a preset routine, which has eighteen positions, or may teach the manipulator a new routine to be repeated by the computer.

TEACH.BAS 5-2-83

```
100 REM  ****************************************************************
110 REM
120 REM          THIS PROGRAM EXECUTES A TEACH ROUTINE FOR DINO.
130 REM
150 REM          CREATED BY TOM LOOKE:  2 APRIL 1983
160 REM
170 REM  ****************************************************************
190 REM
200 REM
210 REM          ********************************************
220 REM          *                                          *
230 REM          *         INITIAL VARIABLE REGION          *
240 REM          *                                          *
250 REM          ********************************************
260 REM
270 REM
275              DIM A%(100)
280              DIM B%(100)
290              DIM C%(100)
295              DIM E%(100)
296              DIM F%(100)
300              AX1% = 1
310              AX2% = 2
320              AX3% = 3
324              AX4% = 4
325              AX5% = 5
330              DIR1% = 1
340              DIR2% = 2
350              CLR% = 0
360              I1 = 0
370              I2 = 0
380              I3 = 0
390              IT = 0
400              Z% = 0
405              X1 = 0
410              SET% = 1
420              X5 = 0
430              R = 0
440              ST = 0
450              REG = 0
480 REM
490 REM
500 REM          ***************  MAIN PROGRAM  ***************
501 REM
502              CALL PORTS
503              CALL CLRALL
504              CALL ENABLE(SET%)
505              PRINT "DO YOU WANT THE SAVED ROUTINE"
506              INPUT R2$
507              IF R2$ = "YES" THEN 600
510              PRINT "HOW MANY POSITIONS ARE IN THE ROUTINE"
```

TEACH.BAS 5-2-83

```
520            INPUT N
530            GOSUB 1070
532            PRINT "DO YOU WANT THE HORIZONTAL CONTROL REMOVED"
534            INPUT R1$
535            IF R1$ = "YES" THEN 538
536            REG = 0
537            GOTO 540
538            REG = 1
540            PRINT "HOW MANY TIMES DO YOU WANT THE ROUTINE RUN"
541            INPUT TX
542            FOR O = 1 TO TX
543            GOSUB 2000
544            NEXT O
545            PRINT "WOULD YOU LIKE THE ROUTINE RUN AGAIN"
546            INPUT U$
547            IF U$ = "YES" THEN 532
550            PRINT "WOULD YOU LIKE TO RUN A DIFFERENT ROUTINE"
552            INPUT T$
554            IF T$ = "YES" THEN 510
555 REM
558 REM
560 REM        **********  SAVED ROUTINE  *****************
600      A%(1) = 91
601      A%(1) = 91
602      A%(2) = 91
603      A%(3) = 230
604      A%(4) = 230
605      A%(5) = 230
606      A%(6) = 91
607      A%(7) = 91
608      A%(8) = 91
609      A%(9) = 230
610      A%(10) = 230
611      A%(11) = 230
612      A%(12) = 230
613      A%(13) = 230
614      A%(14) = 230
615      A%(15) = 170
616      A%(16) = 170
617      A%(17) = 170
618      A%(18) = 91
651      B%(1) = 101
652      B%(2) = 101
653      B%(3) = 101
654      B%(4) = 240
655      B%(5) = 240
656      B%(6) = 101
657      B%(7) = 101
658      B%(8) = 101
659      B%(9) = 101
660      B%(10) = 20
```

```
661     B%(11) = 45
662     B%(12) = 230
663     B%(13) = 40
664     B%(14) = 230
665     B%(15) = 230
666     B%(16) = 60
667     B%(17) = 60
668     B%(18) = 101
701     C%(1) = 140
702     C%(2) = 140
703     C%(3) = 140
704     C%(4) = 213
705     C%(5) = 240
706     C%(6) = 240
707     C%(7) = 140
708     C%(8) = 140
709     C%(9) = 140
710     C%(10) = 238
711     C%(11) = 100
712     C%(12) = 213
713     C%(13) = 213
714     C%(14) = 100
715     C%(15) = 100
716     C%(16) = 100
717     C%(17) = 30
718     C%(18) = 140
751     E%(1) = 1
752     E%(2) = 2
753     E%(3) = 2
754     E%(4) = 1
755     E%(5) = 1
756     E%(6) = 1
757     E%(7) = 1
758     E%(8) = 2
759     E%(9) = 2
760     E%(10) = 1
761     E%(11) = 1
762     E%(12) = 2
763     E%(13) = 2
764     E%(14) = 2
765     E%(15) = 2
766     E%(16) = 2
767     E%(17) = 1
768     E%(18) = 2
801     F%(1) = 10000
802     F%(2) = 1000
803     F%(3) = 0
804     F%(4) = 0
805     F%(5) = 0
806     F%(6) = 0
807     F%(7) = 5000
```

TEACH.BAS 5-2-83

```
808      F%(8) = 0
809      F%(9) = 0
810      F%(10) = 0
811      F%(11) = 0
812      F%(12) = 0
813      F%(13) = 0
814      F%(14) = 0
815      F%(15) = 0
816      F%(16) = 0
817      F%(17) = 0
818      F%(18) = 0
850      N = 18
900      GOTO 532
910               END
950 REM
960 REM
970 REM
980 REM
990 REM
1000 REM          *******************************************
1010 REM          *                                         *
1020 REM          *              TEACH ROUTINE              *
1030 REM          *                                         *
1040 REM          *******************************************
1050 REM
1060 REM
1070              FOR I = 1 TO N
1080              PRINT "PLACE ROBOT IN POSITION ";I
1090              PRINT "THEN PRESS THE ZERO KEY AND RETURN"
1100              INPUT Y
1110              CALL ADC08(AX1%,A%(I))
1120              CALL ADC08(AX2%,B%(I))
1130              CALL ADC08(AX3%,C%(I))
1140              NEXT I
1150              RETURN
1900 REM
1910 REM
1920 REM          ******************************************
1930 REM          *                                        *
1940 REM          *            REPEAT ROUTINE              *
1950 REM          *                                        *
1960 REM          ******************************************
1970 REM
1980 REM
2000              FOR I = 1 TO N
2001              I1 = 0
2002              I2 = 0
2003              I3 = 0
2004              IT = 0
2005              Z% = 0
2006              X1 = 0
```

```
2007           X5 = 0
2008           R = 0
2009           ST = 0
2030           IF A%(I) > 127 THEN A%(I) = A%(I) - 255
2040           IF B%(I) > 127 THEN B%(I) = B%(I) - 255
2050           IF C%(I) > 127 THEN C%(I) = C%(I) - 255
2051           IF I = 14 THEN 2053
2052           GOTO 2059
2053           GOSUB 9000
2059           IF B%(I) < 0 AND C%(I) > 0 THEN 2065
2060           GOTO 2500
2061 REM
2062 REM       *************  CLOSE COUPLED  *************
2063 REM
2065           X1 = 1
2070           GOSUB 3100
2080           GOSUB 3200
2090           GOSUB 3300
2100           IF IT<>3 THEN 2070
2110           GOSUB 7000
2115           CALL AXIS(AX4%,CLR%)
2150           GOTO 2545
2200           RETURN
2300 REM
2350 REM       *************  FULLY EXTENDED  ************
2400 REM
2500           X1 = 0
2510           GOSUB 3100
2520           GOSUB 3200
2530           GOSUB 3300
2540           IF IT<>3 THEN 2510
2542           GOSUB 7000
2543           CALL AXIS(AX4%,CLR%)
2545           PRINT "POSITION ";I
2546           FOR J = 1 TO 1000
2547           NEXT J
2548           NEXT I
2550           RETURN
2600 REM
2700 REM
2800 REM
2900 REM
3000 REM       ******************************************
3010 REM       *                                        *
3020 REM       *     MOVE AXIS TWO TO DESIRED POSITION  *
3030 REM       *                                        *
3040 REM       ******************************************
3050 REM
3060 REM
3100           CALL ADC08(AX2%,E%)
3110           IF E% > 127 THEN E% = E% - 255
```

```
3115           IF E% < -127 THEN E% = E% + 255
3120           IF ABS(E% - B%(I)) < 5 THEN 3500
3130           IF E% > B%(I) THEN 3160
3140           CALL AXIS(AX2%,DIR2%)
3150           RETURN
3160           CALL AXIS(AX2%,DIR1%)
3170           RETURN
3180 REM
3182 REM
3184 REM       ******************************************
3186 REM       *                                        *
3188 REM       *    MOVE AXIS THREE TO DESIRED POSITION *
3190 REM       *                                        *
3192 REM       ******************************************
3194 REM
3196 REM
3200           CALL ADC08(AX3%,F%)
3210           IF F% > 127 THEN F% = F% - 255
3215           IF F% < -127 THEN F% = F% + 255
3220           IF ABS(F% - C%(I)) < 5 THEN 3600
3230           IF F% > C%(I) THEN 3260
3240           CALL AXIS(AX3%,DIR2%)
3250           RETURN
3260           CALL AXIS(AX3%,DIR1%)
3270           RETURN
3280 REM
3282 REM
3284 REM       ******************************************
3286 REM       *                                        *
3288 REM       *     MOVE AXIS ONE TO DESIRED POSITION  *
3290 REM       *                                        *
3292 REM       ******************************************
3294 REM
3296 REM
3300           CALL ADC08(AX1%,D%)
3305           IF ST = 1 THEN 3395
3310           IF D% > 127 THEN D% = D% - 255
3315           IF D% < -127 THEN D% = D% + 255
3320           IF ABS(D% - A%(I)) < 15 THEN 3900
3340           IF D% > A%(I) THEN 3370
3350           CALL AXIS(AX1%,DIR1%)
3360           X5 = 1
3365           RETURN
3370           CALL AXIS(AX1%,DIR2%)
3380           X5 = 2
3390           RETURN
3395           CALL AXIS(AX1%,CLR%)
3397           RETURN
3400 REM
3410 REM
3470 REM       *************  NULL AXIS TWO  **************
```

TEACH.BAS 5-2-83

```
3480 REM
3500          CALL AXIS(AX2%,CLR%)
3510          IF I2 = 1 THEN 3550
3520          I2 = 1
3530          IT = IT + 1
3550          RETURN
3560 REM
3570 REM
3580 REM      ************  NULL AXIS THREE  *************
3590 REM
3600          CALL AXIS(AX3%,CLR%)
3610          IF I3 = 1 THEN 3650
3620          I3 = 1
3630          IT = IT + 1
3650          RETURN
3700 REM
3750 REM
3800 REM      **************  NULL AXIS ONE  *************
3850 REM
3900          IF REG = 1 THEN 6000
3905          IF I1 = 1 THEN 3990
3910          IF X5 = 0 THEN 3920
3915          GOTO 3950
3920          I1 = 1
3930          IT = IT + 1
3940          RETURN
3950          IF X1 = 0 THEN 3966
3960          GOSUB 4000
3964          GOTO 3970
3966          GOSUB 4500
3970          I1 = 1
3980          IT = IT + 1
3990          CALL AXIS(AX1%,CLR%)
3991          RETURN
3992 REM
3994 REM
3996 REM      ****  SPECIAL ROUTINE TO NULL AXIS ONE  ****
3998 REM
4000          CALL AXIS(AX2%,CLR%)
4002          CALL AXIS(AX3%,CLR%)
4003          R = 50
4004          FOR J = 1 TO 50
4005          CALL AXIS(AX1%,CLR%)
4020          CALL AXIS(AX1%,DIR1%)
4030          FOR K = 1 TO R
4040          NEXT K
4050          CALL AXIS(AX1%,CLR%)
4060          CALL AXIS(AX1%,DIR2%)
4080          FOR K = 1 TO R
4090          NEXT K
4100          NEXT J
```

TEACH.BAS 5-2-83

```
4105           CALL AXIS(AX1%,CLR%)
4110           ST = 1
4150           RETURN
4200 REM
4300 REM
4350 REM       ****  ADDED ROUTINE FOR FULLY EXTENDED  ****
4400 REM
4500           IF X5 = 2 THEN 4600
4510           CALL AXIS(AX1%,CLR%)
4520           GOSUB 5000
4530           CALL AXIS(AX1%,DIR1%)
4540           GOSUB 5100
4550           CALL AXIS(AX1%,CLR%)
4560           RETURN
4600           CALL AXIS(AX1%,CLR%)
4610           GOSUB 5000
4620           CALL AXIS(AX1%,DIR2%)
4630           GOSUB 5100
4640           CALL AXIS(AX1%,CLR%)
4650           RETURN
5000           FOR J = 1 TO 240
5010           NEXT J
5020           RETURN
5100           FOR J = 1 TO 79
5110           NEXT J
5115           ST = 1
5120           RETURN
5800 REM
5900 REM
5950 REM       ************  NULL AXIS ONE  **************
5960 REM
6000           CALL AXIS(AX1%,CLR%)
6010           IF I1 = 1 THEN 6050
6020           I1 = 1
6030           IT = IT + 1
6050           RETURN
6100 REM
6900 REM       ***********  MOVE AXIS FIVE  **************
6950 REM
7000           IF E%(I) = 2 THEN 7200
7001           IF E%(I) = 0 THEN 7060
7010 REM
7020 REM       *****  CLOSE THE HAND  ********************
7030 REM
7040           CALL AXIS(AX5%,CLR%)
7045           CALL AXIS(AX5%,DIR1%)
7050           GOSUB 8000
7060           RETURN
7070 REM
7080 REM       *****  OPEN THE HAND  *********************
7090 REM
```

TEACH.BAS 5-2-83

```
7200          CALL AXIS(AX5%,CLR%)
7205          CALL AXIS(AX5%,DIR2%)
7210          GOSUB 8000
7220          RETURN
7300 REM
7410 REM
7700 REM
7800 REM      ***********  EXTRA DELAY  *****************
7900 REM
8000          FOR J = 1 TO F%(I)
8100          NEXT J
8200          RETURN
9000          CALL AXIS(AX4%,DIR1%)
9010          RETURN
```

APPENDIX C

Bode Input Program

"SQBODE"

This program was used to generate a square wave approximation of a sine-wave input to the system.

APPENDIX C

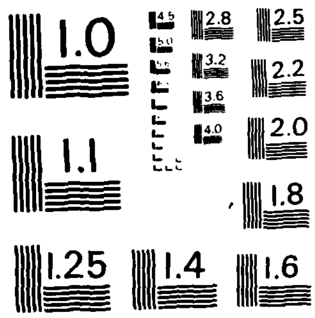
B:SQBODE.BAS 5-2-83

```
10 NUMO% = 0
20 NUM1% = 1
30 CALL PORTS
40 PRINT "ENTER AXIS 1 TO 5:"
50 INPUT A%
60 PRINT "ENTER TIME DELAY:"
70 INPUT F
80 PRINT "ENTER NUMBER OF PERIODS:"
90 INPUT N
100 PRINT
110 PRINT
115 CALL CLRALL
120 CALL ENABLE(NUM1%)
130 FOR I = 1 TO N
140 D% = 1
150 CALL AXIS(A%,D%)
160 GOSUB 260
165 D% = 0
170 CALL AXIS(A%,D%)
180 D% = 2
190 CALL AXIS(A%,D%)
200 GOSUB 260
210 CALL CLRALL
220 NEXT I
230 CALL ENABLE(NUMO%)
240 GOTO 40
250 END
260 FOR J = 1 TO F
290 NEXT J
300 RETURN
310 END
```

AD-A134 185 STABILITY ENHANCEMENT OF A FLEXIBLE ROBOT MANIPULATOR (U) NAVAL ACADEMY ANNAPOLIS MD T D LOOKE 24 JUN 83 USNA-TSPR-126 2/2

UNCLASSIFIED F/G 12/1 NL

END
DATE
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

APPENDIX D

Control Time Derivation Program

"TRIDENT 2"

Given the damping ratio and natural frequency as user inputs, this program varied t_1 to find t_p where the value of $y(t_p)$ was equal to the steady state output. Equation 4-5 was used to find t_p given t_1, and equation 4-3 was used to find $y(t_p)$ given t_1 and t_p.

B:TRIDENT2.BAS

5-2-83

```
50 A = 0
60 B = .1
70 C = .2
115 PRINT "INPUT 2*ZETA*WN, WN^2"
116 INPUT D,X
117 W = X^.5
118 Z = D/(2*W)
130 S = Z*W
140 Y = W*(1-Z^2)^.5
150 P = 3.14159265# - ATN(Y/S)
155 E = W/Y
180 F = -Y*COS(P)-S*SIN(P)+Y*EXP(S*B)*COS(Y*B+P)+S*EXP(S*B)*SIN(Y*B+P)
190 G = Y*SIN(P)-S*COS(P)-Y*EXP(S*B)*SIN(Y*B+P)+S*EXP(S*B)*COS(Y*B+P)
320 T = (3.14159265# - ATN(ABS(F/G)))/Y
330 M = E*EXP(-S*T)*SIN(Y*T-P)-E*EXP(-S*(T-B))*SIN(Y*(T-B)-P)
340 IF ABS(M-1) < .001 GOTO 410
350 IF M < 1 GOTO 380
355 C = B
360 B = A+(B-A)/2
370 GOTO 180
380 A = B
390 B = B+(C-B)/2
400 GOTO 180
410 PRINT "YMAG = ",M
420 PRINT "WHEN T1 = ",B
430 PRINT "AND  T2 = ",T
435 GOTO 50
440 END
```

APPENDIX E

Driver Routines for Hydraulic Arm

"AXES"

This appendix contains the machine language subroutines which were used to drive the manipulator.

AXES.MAC 5-2-83

```
;       ***********************************************************
;       ***********************************************************
;
;               DRIVER ROUTINES FOR HYDRAULIC ARM
;
;
;               WRITTEN BY RICK BOYER, 8 JUNE 1982
;               UP-DATED  16 SEPT. 82   RNB
;
;
;       ROUTINES AVAILABLE;
;
;               AXIS --->  CALL AXIS(A,B)
;
;                       A --> AXIS NUMBER 1 TO 5 (INTEGER)
;                       B --> TO CLEAR AXIS ENTER = 0
;                             ENTER 1 OR 2 TO MOVE IN
;                             DIRECTION REQUIRED.        (INTEGER)
;
;
;               PORTS --->  CALL PORTS
;
;                       SETS UP THE 8255 PIO PORTS FOR
;                       PROPER DATA FLOW PATHS.
;
;
;               CLRALL --->  CALL CLRALL
;
;                       CLEARS ALL THE PORTS (A,B,C) TO
;                       ZERO. THIS WILL ALSO INITIALIZE
;                       ALL THE TEMP STORAGE LOCATIONS.
;
;
;               ENABLE --->  CALL ENABLE(A)
;
;                       A --> 1 TURNS ON DRIVE LOGIC    (INTEGER)
;                             0 TURNS OFF DRIVE LOGIC   (INTEGER)
;
;
;
;       ***********************************************************
;       ***********************************************************
;
        CPM EQU 00H
        PORTA SET 0E4H
        PORTB SET 0E5H
        PORTC SET 0E6H
        CNTL SET 0E7H
        PUBLIC AXIS
        PUBLIC CLRALL
        PUBLIC ENABLE
```

AXES.MAC 5-2-83

```
        PUBLIC PORTS
;
TEMP:   DS      4
TEMP1:  DS      4
TEMP2:  DS      4
TEMP3:  DS      4
TEMP4:  DS      4
;
AXIS:
        CALL ERRCHK
        MOV A,M
        STA TEMP4
        CPI 01
        JZ AX1
        CPI 02
        JZ AX2
        CPI 03
        JZ AX3
        CPI 04
        JZ AX4
        CPI 05
        JZ AX5
        LXI H,MESS2
        MVI B,LMESS2
        CALL MSG
        CALL CLRALL
        CALL CROUT
        CALL CPM
;
;       ******************
;
AX1:
        MVI A,01H
        STA TEMP3
        LDA TEMP
        CPI 00
        JZ CLR
        CPI 01
        JZ AX1A
        MVI A,01H
        CALL SETA
        OUT PORTA
        RET
AX1A:   MVI A,01H
        CALL SETB
        OUT PORTB
        RET
;
;       ********************
;
AX2:
```

```
        MVI A,02H
        STA TEMP3
        LDA TEMP
        CPI 00
        JZ CLR
        CPI 01H
        JZ AX2A
        MVI A,02H
        CALL SETA
        OUT PORTA
        RET
AX2A:   MVI A,02H
        CALL SETB
        OUT PORTB
        RET
;
;       ******************
;
AX3:
        MVI A,04H
        STA TEMP3
        LDA TEMP
        CPI 00
        JZ CLR
        CPI 01H
        JZ AX3A
        MVI A,04H
        CALL SETA
        OUT PORTA
        RET
AX3A:   MVI A,04H
        CALL SETB
        OUT PORTB
        RET
;
;       ******************
;
AX4:
        MVI A,08H
        STA TEMP3
        LDA TEMP
        CPI 00
        JZ CLR
        CPI 01H
        JZ AX4A
        MVI A,08H
        CALL SETA
        OUT PORTA
        RET
AX4A:   MVI A,08H
        CALL SETB
```

```
        OUT PORTB
        RET
;
;       *****************
;
AX5:
        MVI A,10H
        STA TEMP3
        LDA TEMP
        CPI 00
        JZ CLR
        CPI 01H
        JZ AX5A
        MVI A,10H
        CALL SETA
        OUT PORTA
        RET
AX5A:   MVI A,10H
        CALL SETB
        OUT PORTB
        RET
;
;       *******************
;
MESS2:DB CR,LF,LF,'ERROR!!! AXIS # OUT OF RANGE  1 TO 5',CR,LF,LF
LMESS2 EQU $-MESS2
;
;       ******************************
;
CLR:
        LDA TEMP3
        MOV B,A
        LDA TEMP1
        ANA B
        JZ CLRB
        LDA TEMP1
        XRA B
        STA TEMP1
        OUT PORTA
        RET
CLRB:   LDA TEMP2
        ANA B
        JZ CLRC
        LDA TEMP2
        XRA B
        STA TEMP2
        OUT PORTB
CLRC:   RET
;
SETA:
        MOV B,A
```

```
        LDA TEMP1
        ORA B
        STA TEMP1
        CALL CHECK
        LDA TEMP1
        RET
;
SETB:
        MOV B,A
        LDA TEMP2
        ORA B
        STA TEMP2
        CALL CHECK
        LDA TEMP2
        RET
;
CHECK:
        LDA TEMP1
        MOV B,A
        LDA TEMP2
        ANA B
        JNZ ERROR2
        RET
;
;
ERROR2:
        LXI H,MESS4
        MVI B,LMESS4
        CALL MSG
        LDA TEMP4
        CALL NMOUT
        LXI H,MESS3
        MVI B,LMESS3
        CALL MSG
        CALL CLRALL
        CALL CROUT
        CALL CPM
MESS4:DB CR,LF,LF,'ERROR!!!  AXIS #'
LMESS4 EQU $-MESS4
MESS3:DB ' DRIVING IN BOTH DIRECTIONS',CR,LF,LF
LMESS3 EQU $-MESS3
;
;       ******************************
;
ERRCHK:
        LDAX D
        STA TEMP
        CPI 03H
        JP ERR1
        RET
;
```

```
ERR1:   LXI H,MESS1
        MVI B,LMESS1
        CALL MSG
        CALL CLRALL
        CALL CROUT
        CALL CPM
MESS1:  DB 'ERROR!!! AXIS DIRECTION OUT OF RANGE, ENTER 0 OR 1'
        LMESS1 EQU $-MESS1
;
;       **************************************
;
CLRALL:
        MVI A,00
        OUT PORTA
        OUT PORTB
        STA TEMP
        STA TEMP1
        STA TEMP2
        STA TEMP3
        STA TEMP4
        RET
;
;       **************************************
;
PORTS:
        MVI A,88H
        OUT CNTL
        RET
;
;       **************************************
;
ENABLE:
        MOV A,M
        CPI 00
        JNZ ON
        MVI A,00
        OUT PORTC
        RET
ON:     MVI A,01H
        OUT PORTC
        RET
;
;
;       ************************************************
;       ************************************************
;
        CR SET 0DH
        LF SET 0AH
;
CO:
        PUSH H
```

```
        PUSH D
        PUSH B
        MVI C,02H
        MOV E,A
        CALL 05H
        POP B
        POP D
        POP H
        RET
;
MSG:
        MOV A,M
        CALL CO
        INX H
        DCR B
        JNZ MSG
        RET
;
CROUT:
        MVI A,CR
        CALL CO
        MVI A,LF
        CALL CO
        RET
;
NMOUT:
        PUSH H
        PUSH D
        PUSH B
        PUSH PSW
        RRC
        RRC
        RRC
        RRC
        ANI OFH
        CALL PRVAL
        CALL CO
        POP PSW
        ANI OFH
        CALL PRVAL
        CALL CO
        POP B
        POP D
        POP H
        RET
;
PRVAL:
        ADI '0'
        CPI 3AH
        RM
        ADI 7
```

```
RET
END
```

APPENDIX F

FORTRAN User Subroutine Library

"USERLIB"

This set of FORTRAN subroutines was used in "TEACH" to read the values of the potentiometers through A/D converters. The Cromemco 8 bit A/D converter was the one used in "TEACH."

USERLIB.DOC 5-2-83

```
C***************************************************************************
C
C       *************************************************
C               W&SE FORTRAN USER SUBROUTINE LIBRARY
C       *************************************************
C
C
C               LAST UPDATED    11/17/81        W.M.L.
C
C***************************************************************************
C
C       *****************************
C               FORTRAN MODULES
C       *****************************
C
C
C       CDCCNV ----> HANDLER FOR THE CAL. DATA CORP. 12-BIT D TO A
C                    BOARD.  THIS PROGRAM ALLOWS THE USER TO ENTER
C                    THE VOLTAGE TO BE CONVERTED AS A REAL NUMBER
C                    IN THE RANGE -10.00 < XX.XX < 10.00. IT WILL
C                    CONVERT THIS REAL NUMBER  TO A BINARY
C                    NUMBER WHICH CAN BE ACCEPTED BY THE D TO A
C                    CONVERTER.
C
C       CALL SEQ        CALL CDCCNV (CHANNEL,DATA)
C
C       ARGUMENTS       CHANNEL - CHANNEL NUMBER BETWEEN
C                                 1 AND 4 (LOGICAL)
C                       DATA    - DESIRED OUTPUT VALUE (REAL)
C
C       EXTERNALS       DAC12   - CAL DATA DRIVER
C                       $CROUT,$MSG,$CO - TTY IO ROUTINES
C                       $AB,$CH,$DA,$L1,$MA,$ND,$SB,$ST,$TI,$W2
C                                - FORM FORTRAN LIBRARY
C
C               *       *       *       *       *       *
C
C       TECIN -----> FOR THE TECMAR 12-BIT A TO D CONVERTER BOARD.
C                    THIS ROUTINE ALLOWS THE USER CONVERT  ANALOG
C                    VOLTAGES RANGING FROM -10.00 < XX.XX < 10.00
C                    VOLTS. IT RETURNS THE CONVERTED VOLTAGE AS
C                    A REAL NUMBER IN THE ABOVE FORMAT.
C
C       CALL SEQ        CALL TECIN (CHANNEL,DATA)
C
C       ARGUMENTS       CHANNEL - CHANNEL NUMBER BETWEEN
C                                 0 AND 15 (LOGICAL).
C                       DATA    - RECEIVED REAL VALUE FROM
C                                 DRIVER.
C
C       EXTERNALS       ADC12   - TECMAR DRIVER
```

USERLIB.DOC 5-2-83

```
C                    $CROUT,$MSG,$CO - TTY IO ROUTINES
C                    $CA,$DA,$M9,$T1 - FROM FORTN. LIBRARY
C
C          *        *        *        *        *        *
C
C     CROMIN ----> FOR THE CROMEMCO 8-BIT A TO D BOARD.  THIS
C                 ROUTINE ALLOWS THE USER TO CONVERT ANALOG
C                 VOLTAGES RANGING FROM -10.00 < XX.XX < 10.00
C                 VOLTS. IT RETURNS THE CONVERTED VOLTAGE AS
C                 A REAL NUMBER IN THE ABOVE FORMAT.
C
C     CALL SEQ         CALL CROMIN (CHANNEL,DATA)
C
C     ARGUMENTS        CHANNEL - CHANNEL NUMBER BETWEEN
C                                1 AND 7 (LOGICAL).
C                      DATA    - RECEIVED REAL VALUE FROM
C                                DRIVER.
C
C     EXTERNALS        ADC08   - CROMEMCO A/D DRIVER
C                      $CROUT,$MSG,$CO - TTY IO ROUTINES
C                      $CA,$DA,$M9,$T1 - FROM FORTN. LIBRARY
C
C          *        *        *        *        *        *
C
C     CROMCO ----> HANDLER FOR THE CROMEMCO EIGHT BIT D TO A
C                 BOARD.VOLTAGE TO BE CONVERTED IS ENTERED
C                 AS A REAL NUMBER IN THE RANGE -10.00 <
C                 XX.XX < 10.00. THIS WILL BE CONVERTED TO
C                 AN EIGHT BIT BINARY NUMBER AND PASSED TO
C                 THE CROMEMCO DRIVER.
C
C     CALL SEQ         CALL CROMCO (CHANNEL, DATA)
C
C     ARGUMENTS        CHANNEL - CHANNEL NUMBER BETWEEN
C                                1 AND 7 (LOGICAL)
C                      DATA    - DESIRED OUTPUT VALUE (REAL)
C
C     EXTERNALS        DAC08   - CROMEMCO 7DAIO D/A DRIVER
C                      $CROUT,$MSG,$CO - TTY IO ROUTINES
C                      $AB,$CJ,$DB,$LI,$MA,$ND,$SB,$ST,$TI,$W2
C                                - FROM FORTRAN LIBRARY
C
C     PROGRAMMING NOTE         - THIS DEVICE USES  SAMPLE AND
C                                HOLD CIRCUITRY TO STORE OUTPUT
C                                VOLTAGE. IT MUST  BE REFRESHED
C                                OFTEN TO CORRECT DRIFT.
C
C          *        *        *        *        *        *
C
C**********************************************************************
;
```

```
;        ******************************
;               MACRO SUBROUTINES
;        ******************************
;
;        PRNTHX -----> PRINTS THE HEX EQUIV. OF THE INTEGER INPUT
;
;        CALL SEQ         CALL PRNTHX (IDATA)
;
;        ARGUMENTS        IDATA   -INTEGER TO BE DUMPED
;
;        EXTERNALS        $CROUT,$NMOUT,$PRVAL,$CO - TTY & CONVERSION SUBS
;
;              *       *       *       *       *       *
;
;        FNMOUT -----> PRINTS HEX EQUIV. OF BYTE WIDE NUMBER
;
;        CALL SEQ         CALL FNMOUT (LDATA)
;
;        ARGUMENTS        LDATA - BYTE WIDE DATA TO BE DUMPED
;
;        EXTERNALS        $NMOUT,$PRVAL,$CO - TTY IO ROUTINES
;
;              *       *       *       *       *       *
;
;        FCO    -----> PRINTS BYTE WIDE DATA IN ASCII
;
;        CALL SEQ         CALL FCO (LDATA)
;
;        ARGUMENTS        LDATA - BYTE TO BE TYPED (LOGICAL)
;
;        EXTERNALS        $CO - TTY CHARACTER OUT
;
;              *       *       *       *       *       *
;
;        FCI    -----> READS BYTE OF DATA FROM TERMINAL IN ASCII
;
;        CALL SEQ         CALL FCI (LDATA)
;
;        ARGUMENTS        LDATA - ASCII BYTE READ FROM TERMINAL
;
;        EXTERNALS        $CI - TTY CHARACTER IN
;
;              *       *       *       *       *       *
;
;        MULOUT ----> DRIVER FOR MULLEN CONTROL BOARD OUTPUT
;
;        CALL SEQ         CALL MULOUT (LDATA)
;
;        ARGUMENTS        LDATA - BYTE WIDE DATA TO RELAYS
;                                 (LOGICAL)
;
```

```
;       EXTERNALS       NONE
;
;            *       *       *       *       *       *
;
;       MULLIN -----> DRIVER FOR MULLEN BOARD INPUT
;
;       CALL SEQ        CALL MULLIN (LDATA)
;
;       ARGUMENTS       LDATA - BYTE WIDE DATA FROM OPTO-
;                               COUPLERS (LOGICAL)
;
;       EXTERNALS       NONE
;
;            *       *       *       *       *       *
;
;       ADC12  -----> DRIVER FOR TECMAR 12BIT A/D
;
;       CALL SEQ        CALL ADC12 (LCHAN,IDATA)
;
;       ARGUMENTS       LCHAN - CHANNEL NUMBER BETWEEN 0 AND
;                               15  (LOGICAL).
;                     : IDATA - SIGNED INTEGER REPRESENTING THE
;                               CONVERTED VALUE.
;
;       EXTERNALS       $CROUT,$MSG,$CO - TTY ROUTINES
;
;            *       *       *       *       *       *
;
;       DAC12  -----> DRIVER FOR CAL. DATA 12 BIT D/A
;
;       CALL SEQ        CALL DAC12 (LCHAN,IDATA)
;
;       ARGUMENTS       LCHAN - CHANNEL NUMBER BETWEEN 1 AND
;                               4 (LOGICAL);
;                       IDATA - SIGNED INTEGER VALUE
;
;       EXTERNALS       $CROUT,$MSG,$CO - TTY IO ROUTINES
;
;            *       *       *       *       *       *
;
;       ADC08  -----> DRIVER FOR CROMEMCO 8 BIT A/D
;
;       CALL SEQ        CALL ADC08 (LCHAN,LDATA)
;
;       ARGUMENTS       LCHAN - CHANNEL NUMBER BETWEEN 1
;                               AND 7 (LOGICAL)
;
;                       LDATA - BYTE WIDE DATA TO BE CONVERTED
;                               (LOGICAL)
;
;       EXTERNALS       $CROUT,,$MSG,$CO - TTY ROUTINES
```

```
;
;          *        *        *        *        *        *
;
;   DAC08  -----> DRIVER FOR CROMEMCO D/A CONVERTER
;
;   CALL SEQ        CALL DAC08 (LCHAN,LDATA)
;
;   ARGUMENTS       LCHAN - CHANNEL NUMBER BETWEEN 1
;                           AND 7 (LOGICAL).
;                   LDATA - BYTE WIDE DATA TO BE CONVERTED
;                           (LOGICAL).
;
;   EXTERNALS       $CROUT,$MSG,$CO - TTY IO ROUTINES
;
;          *        *        *        *        *        *
;
;   DDCOUT  -----> DRIVER FOR CROMEMCO DIGITAL IO OUTPUT
;
;   CALL SEQ        CALL DDCOUT (LDATA)
;
;   ARGUMENTS       LDATA - BYTE DATA TO INPUT (LOGICAL)
;
;   EXTERNALS       NONE
;
;          *        *        *        *        *        *
;
;   DDCIN  -----> DRIVER FOR CROMEMCO PIO INPUT
;
;   CALL SEQ        CALL DDCIN (LDATA)
;
;   ARGUMENTS       LDATA - BYTE WIDE DATA TO XMIT (LOGICAL)
;
;   EXTERNALS       NONE
;
;          *        *        *        *        *        *
;
;*********************************************************************
;
;   *******************************
;      USERLIB INTERNAL ROUTINES
;   *******************************
;
;   NOTE: THESE ROUTINES ARE NOT CALLABLE FROM FORTRAN.
;
;
;   $CRLF OR $CROUT -> GENERATES A CARRIAGE RETURN - LINE FEED
;                           SEQUENCE TO THE TERMINAL.
;                           CALLS $CO - TTY CHARACTER OUT
;
;   $NMOUT ----------> PCONVERTS AN 8 BIT BINARY NUMBER
;                           TO ASCII HEX AND PRINTS IT ON THE
```

```
;                               TERMINAL.
;                               CALLS $PRVAL,$CO - CONVERSION & TTY OUT
;
;       $PRVAL ----------> CONVERTS HEX DIGET TO ASCII.
;
;       $CNVBN ----------> CONVERTS ASCII DIGET TO HEX.
;
;       $MESS OR $MSG ---> PRINTS ASCII MESSAGE TO THE TERMINAL.
;                               CALLS $CO - TTY CHARACTER OUT
;
;       $CO -------------> PRINTS ASCII CHARACTER TO THE TERMINAL.
;
;       $CI -------------> READS ASCII CHARACTER FROM THE TERMINAL.
;
;*********************************************************************
;
;       TO BUILD USERLIB.REL , USE THE MICROSOFT LIBRARIAN
;       AS FOLLOWS TO LINK THE '.REL' MODULES:
;
;       >A LIB
;       *USERLIB=TECIN,CDCCNV,CROMIN,CROMCO,ADC08,DAC08
;       *USERLIB=ADC12,DAC12,DDCIN,DDCOUT,MULLIN,MULOUT
;       *USERLIB=PRNTHX,FNMOUT,FCI,FCO,NMOUT,MSG,CROUT
;       *USERLIB=CNVBN,PRVAL,CI,CO
;       */E
;
;       NOTE: MODULES MUST BE IN ABOVE ORDER SO THAT THEY
;             WILL BE FOWARD REFERENCED.
;             TO LINK WITH L80- CMNDFILE=OBJFILE,USERLIB/S
;
;*******************************************************************
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER U.S.N.A. - TSPR; no 126 (1983)	2. GOVT ACCESSION NO. A134185	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) STABILITY ENHANCEMENT OF A FLEXIBLE ROBOT MANIPULATOR.		5. TYPE OF REPORT & PERIOD COVERED Final: 1982/1983
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Looke, Thomas D.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS United States Naval Academy, Annapolis.		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS United States Naval Academy, Annapolis.		12. REPORT DATE 24 June 1983
		13. NUMBER OF PAGES 112
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release; its distribution is UNLIMITED.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

This document has been approved for public release; its distribution is UNLIMITED.

18. SUPPLEMENTARY NOTES

Accepted by the U. S. Trident Scholar Committee.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Automata
Robots

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer software programming technique was developed to compensate a highly oscillatory robot system controlled by a bang-bang input. The assumptions that the system was linear and had lumped parameter characteristics allowed a fifth order, simplified dynamic model to be derived. Analysis using frequency response methods led to further simplification of the model to a third order system. Based on the third order model, a technique was developed which would compensate the system with a form of deadbeat control. (OVER)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Simulation of the model driven by the compensated bang-bang input verified the deadbeat response. The technique was implemented on an 8080-based microcomputer system which controlled the input. Acutal system response to the compensated input was observed to be essentially free of the undesirable oscillatory motions, thus yielding an apparently rigid system.