AN INVESTIGATION INTO THE COMPUTER CONTROL OF TRANSMISSION LINE DYNAMICS FOR A SERVO - CONTROLLED HYDRAULIC SYSTEM

BY

JAECHEON LEE

Bachelor of Science in Agriculture
Seoul National University
Seoul, Korea
1980

Master of Science
Korea Advanced Institute of
Science & Technology
Seoul, Korea
1985

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 1989

AN INVESTIGATION INTO THE COMPUTER CONTROL

OF TRANSMISSION LINE DYNAMICS

FOR A SERVO - CONTROLLED

HYDRAULIC SYSTEM

Thesis Approved:

Thesis Adviser

Dean of the Graduate College

ACKNOWLEDGEMENTS

"Bless the Lord, O my soul; And all that is within me, bless His holy name!" - Psalms 103:1.

I wish to express sincere appreciation to the principal advisor, Dr. Ing T. Hong, for his encouragement and advice throughout my graduate program. Many thanks also go to Dr. P. M. Moretti, and Dr. R. L. Lilley for serving my graduate committee. Their suggestions and advices were very valuable in this study. I am grateful to my former adviser, Kyung-Woong Kim at the Korea Advanced Institute of Science and Technology. He awaked me to reflect upon the attitude and the mind of a researcher whenever I met difficulty.

I am indebted to Dr. Richard K. Tessmann for his kind guidance and advice in the tests. I would especially like to express my sincere gratitude to Arrington Mikle, Ed Stanton, and S. C. Chung. Without their supports and encouragements the study would not have been possible. I extended my thanks to Linda C. Gray for her checking of English grammer; Man S. Hur and Joong H. Baek for their friendship.

I am very grateful to my lovely wife, Dong-sook, for her encouragements, understanding, and praying. I wish to express my deepest appreciation to my mother and sisters who supported all the way and helped me, and dedicate this endeavor to my father in the heaven.

Finally, I would like to own the pleasure of achievement with my friends who were praying for me, Young B. Jang, Jung H. Lee, Sae H. Kim, Mrs. Ho J. Jung, Mrs. Eun K. Lim, as well as the whole Korean church's family.

"I can do all things through Christ which strengtheneth me." - Philippians 4:13.

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure		Page

Figure | Page

Figure Page

Figure Page

Figure Page

NOMENCLATURE

A	inner section area of a tube
A_1, A_2	constants
A_p	pressure sensing area of a piston
a_i	parameters for state variables ($i = 1,2,3,...$)
B_1, B_2, B_3	constants
C	fluid capacitance
C_{f1}	internal leakage coefficient in the cylinder
C_{f2}	external leakage coefficient of the cylinder
C_o	constant ($= \Delta t/\Delta x$) or propogation speed of sound
c_i	parameters for sliding plane ($i = 1,2,3,...$)
c_n	discrete output response
D_i	tube inner diameter
D_r	piston diameter
$\hat{E}$	estimated parameter vector
e	error signal
e_n	discrete error signal
F_c	forces exerting into the piston
f_c	friction coefficient
F_L	friction force
$F(s)$, $G(s)$	Laplace transfer function

f	pressure wave frequency
h	discrete time step
I	input pulse or current
i	node number
j	time step number
J	performance index function for optimization
K_{ce}	flow-pressure coefficient
K_d	derivative gain in PID controller
K_i	integrational gain in PID controller
K_p	proprotional gain in PID controller
K_q	flow gain, spool displacement-flow rate
K_i	flow gain, current-spool displacement
k_c	proportional gain constant
k_d	ratio, T/T_d
k_i	ratio, T/T_i
L	fluid inductance
L_d	response delay
L_m	load inductance
L_r	piston stroke
L_t	transmission line length
l	line length
M_f	fluid weight in a tube
M_t	total mass referred to the piston
M_s	fluid mass in a tube
M_w	total weight of piston and load
m	PID control signal
n	total node number

O	output response
P	projection operator
P_i	pressure at point i (i = 1,2,3,...)
P_s	supply pressure
Q_{c1}, Q_{c2}	flow rates in the cylinder
$Q_{d1}, Q_{d2}, Q_{d3}, Q_{d4}$	flow rates in the valve
Q_i	flow rate at point i (i = 1,2,3,...)
R	fluid resistence
R_t	total resistance in the system
R_s	slope of transient response
R_v	valve resistance
r	direction of radial axis
r_a	constant ($= \Delta t/\Delta x$)
r_n	discrete reference signal
s	Laplace transform operator
T	time step
T_d	time constant for derivative gain
T_i	time constant for integrational gain
t	time
t_i	time for input response
t_o	time for output response
u	fluid velocity in x direction
u	control input signal
V_2, V_3, V_4, V_5	fluid volume
V_a	total oil volume except line
V_t	total oil volume in the system
V_{ol}	fluid volume in a tube

v	fluid velocity in r direction
W_d	orifice width in the valve
w	frequency
w_h	hydraulic natural frequency
X_p	piston rod displacement
X_{p1}	piston displacement due to current
X_v	servovalve spool displacement
x	direction of x axis
x_i	optimization index variables ($i = 1,2,3,...$)
$\dot{x}_i$	state variables for a system ($i = 1,2,3,...$)
Y	shunt admittance
y	system output
Z	series impedence
Z_c	characteristic impedence
z	Z transform operator
α_i, β_i	state variables ($i = 1,2,3,...$)
β_e	effective bulk modulus
δ_h	hydraulic damping constant
δ_o	small positive scalar
ϕ	vector of discrete output, y and input u
ΔF	sum of forces
ΔP	pressure difference
$\Delta Q_2, \Delta Q_3, \Delta Q_4, \Delta Q_5$	sum of flow rates
Δt	a time step
Δx	one section line length
Γ	propogation constant

ψ_i	parameters for the input signal in VSS
μ	absolute viscosity of fluid
σ	sliding plane
∂	partial derivative symbol

CHAPTER I

INTRODUCTION

Problem Statement

The research work described in this thesis deals with a study on the control of transmission line dynamics in an electro-hydraulic servovalve controlled cylinder system in which long fluid transmission lines are connected between the servovalve and the double-rod cylinder.

As the lines are longer, the response delay in addition to the phase lag increases and the transient response of the cylinder becomes more oscillatory. This occurs because increased total oil volume makes the hydraulic damping constant smaller. Furthermore, as the current in the servovalve decreases, in other words, the flow rate to the cylinder reduces, the system damping constant also reduces to cause more oscillatory transient response, especially in the low speed range of the actuator. This is a critical problem in servovalve controlled actuator systems that incorporate long transmission lines. For example, the tracking rate on low speed of a Gun/Turret system in a Tank should follow the control signal of the Gunner's handle in order to hit the target quickly and accurately. But long transmission lines between the valve and the actuator worsen the transient response. In order to eliminate this problem, the servo-actuators which incorporating the servovalve and the actuator in a unit have been used. This units can avoids the transmission line dynamics between the servovalve and the actuator. However, the transmission line between the servovalve and the actuator is common in hydraulic system and it affects the system response. Therefore, transmission line should not be ignored for accurate analysis.

Surprisingly, even though there has been a continuing interest in fluid transmission line dynamics for last several decades, no attention has been given to the problem of controlling the transient response which occurs from long transmission line dynamics. Most studies have concentrated on the development of better models and solutions to transient problems of line dynamics, for example, wave propogation theory, analogies with electrical transmission theory, method of characteristics, various approximation techniques such as modal approximation, and approximated state variable represenation. Regardless of its importance no study to control the transient line dynamics in the servovalve - actuator system has been recorded until now. Now, it is the time to develop the technology to control the transmission line dynamics in a hydraulic system. Because the personal computer and the digital data controller, which could not be utilized effectively until now, are so fast that theoretical control algorithm can be implemented into real system.

Purpose of Study

The purpose of this thesis is to develop a successful modelling technique which will be able to faithfully represent hydraulic transmission line characteristics under practical condition. Further, dynamic control strategies, such as the optimal Proportional-Integral-Derivative (PID) control method as well as Variable Structure System (VSS) control technique, will be integrated with the mathematical model development to form an in-line controller to control the line dynamics as desired. A symmetric cylinder system coupled via transmission lines to an electrohydraulic servovalve was used.

Organization

This study is organized in two parts, One is a digital computer simulation performed to establish the theoretical methodology and the other includes experiments using a personal computer's data acquisition and control equipment to support the theoretical results.

Theoretical Methodology

The study begins with the construction of a modelling technique. The technique must be as close as possible, mathematically, to the real system in order to determine the effect of transmission line dynamics. Both of time domain responses and frequency domain responses are required. The study concerning application of the control theory follows.

For convenience in obtaining time domain responses, linearized transfer function method of expressing the servovalve controlled cylinder system has been used. In this method the transmission lines are considered as lumped masses which have no order differential equation mathematically. When the long transmission lines are included, distributed parameters modelling for the lines must be used in order to reduce the discrepancies between the simulation results and the real responses. Therefore, distributed parameters line modelling is considered. All hydraulic components are expressed by ordinary differential equations except the lines which must be expressed by partial differential equations. The problem of determining the transfer function for the total system results from the difficulty of the coupling method between the transmission line and the other hydraulic components. As a method of distributed parameters line modelling, the conventional implicit technique, using Finite Differential Method (FDM), requires a lengthy simulation time. The least time-consuming explicit method, using method of characteristics, becomes progressively more difficult to couple than the implicit techniques as the system becomes more complex. That is, if the transfer function method, which has been used until now, is adhered to. Therefore, for a general study of distributed parameters line modelling methods, a new quasi-explicit method, with average friction theory, is explored using FDM. After comparing the responses of the explicit method and to those of the quasi-explicit method, it was determined that the explicit method for line dynamics coupled with other hydraulic components by way of distributed components modelling should be used to obtain the entire system differential equations. In the time domain, the response analysis is performed.

The system's order must be identified, and the parameters' values estimated in order to apply the control theory. Pulse testing method is applied to identify the system. This can be applied theoretically and experimentally without special frequency analysing equipment. Frequency response analysis is performed using the linearized transfer function method. Both no-order lumped mass line and lumped parameter line modelling were used in order to compare with the results of system identification. The parameters of the system transfer function are estimated using the Orthogonal Projection Algorithm.

For control law, the Variable Structure System (VSS) control algorithm is applied. It is one of the simplest and fastest control algorithms and it is insensitive to the parameter variations. Therefore it can tolerate the inaccuracy of parameter evaluations. In order to evaluate the usefulness of the VSS control the results from the VSS control are compared with those using the conventional PID control in which the optimum gains were determined using the Powell optimization theory and Ziegler-Nichols method.

Experimental Verification

In support of the theoretical approach a servovalve controlled cylinder system was implemented.

A Linear Voltage Transducer was used to measure the displacement of the piston rod. A digital data control instrument using a personal computer and a digital oscilloscope were used for data acquisition and controlling the system.

Time domain transient responses were compared with theoretical simulation results. From the transient responses, the system was identified and parameters were established. PID, PI, and VSS feedback control algorithm were then applied.

CHAPTER II

LITERATURE REVIEW

Since many researchers had dealt exclusively with the frequency response characteristics of fluid transmission lines since 1950, Ezekiel and Paynter [1] obtained ordinary differential equations in hyperbolic operators first relating pressures and flows at two cross sections of a hydraulic line, and Oldenburger and Donelson [2] and Regetz [3] verified the validity of these equations in tests.

Odenburger and Donelson [4] simplified Paynter equations. They used rational approximations to the transfer function using infinite products of linear factors instead of using power series expansion in order to overcome mathematical instability.

Brown [5] derived the operator forms of the basic transmission parameters, the propogaion operator, and characteristic impedence for the computation of transient response. He revealed the propogation operator and the characteristic impedence completely determine any response of a uniform linear transmission line, given the end conditions for pressure and flow.

Zielke [6] revealed that the method of characteristics can be adapted to handle frequency-dependent wall shear stress of laminar flow. Brown [7] reformulated Zielke's results and implemented practically using digital computer.

Keller [8] showed the analogies with electrical transmission theory using hydraulic characteristics, i.e. viscosity, density and bulk modulus.

Healey and Hullender [9] modelled the transmission line in state variable form using finite dimensional modal approximation. The linear system coupled together was then solved using state transition methods.

Hullender and Woods [10] developed a technique for formulating the minimum order state space equations using new dependent variables. Using this technique the number of the state variables can be the same of the order of the system.

However no literature above regarded hydraulic line dynamics in a servo-controlled hydraulic system. Viersma [11] addressed the dynamics of hydraulic supply lines for servosystems. This work concerned the installation of gas-accumulators, and pressure control valves in the supply line, in order to maintain constant pressure to the servovalve. Viersma utilized lumped parameter, four-pole equations, for simulating line dynamics. However, his study addressed frequency response analysis and excluded the time domain response.

Watton published a series of papers addressing the transmission line dynamics in a servovalve controlled actuator system [12,13,14,15]. This study revealed the time domain response.

Using a variety of simulation techniques, Watton investigated the transmission lines in fluid power systems to determine the practicability of each approach when applied to some common fluid power circuit elements [12]. An explicit method of characteristics together with implicit methods, analogue simulation, and small signal linearization technique were compared for the combination of line and non-linear electrohydraulic elements and components. He deduced that the method of characteristics is easiest to program and the least time-consuming for time domain analysis. Howevere it becomes more difficult to handle than the implicit method as the fluid power system becomes more complex. He concluded that small oscillatory components superimposed upon the mean response are inherent to the implicit method and require extended simulation time.

Watton investigated the effect of transmission line dynamics in servovalve controlled actuators [13]. He used the linearized transfer function technique in the frequency domain and the method of characteristics, using lossless-line theory, in the time domain. He concluded that due to the effective low inertia of the actuator, even short lines may have a significant effect upon the response. The simulation techniques give consistent natural frequency prediction, when using a linearization approach in the frequency domain and the method of characteristics in the time domain.

In his study of servovalve controlled - single rod actuator system [14], he investigated the open-loop response of single-rod actuators coupled with interconnecting transmission lines using the linearized frequency response technique. He obtained some design criteria indicating the systems' transient response.

He then analyzed the transient response of an electrohydraulic servovalve coupled to an axial piston ball motor [15]. The frequency measurements coupled with a small signal linearized analysis were performed first to define the dynamic charactristics. The method of characteristics coupled with the motor-end boundary equations were used to predict the transient response. This study revealed that the servovalve dynamics, although apparently negligible compared to the major system dynamics, have a marked filtering effect on the transient pressures. He determined that the use of simulation technique using lumped line volume gives misleading results when the line is long.

Zongxia, Yigang and Jingchao studied the coupling method of lumped parameter components with distributed parameter model [16]. They considered a transmission line as a distributed parameter model. They used method of characteristics for distributed parameter line modelling and state space equations for identifying the system equation. However they concerned specific dynamic simulation software of hydraulic system - DSH (Dynamic Simulation of Hydraulic System) [17], did not reveal generalized simulation technique.

CHAPTER III

THEORETICAL APPROACH

An overview of the theoretical approach to the problem of controlling line dynamics is shown on Figure 1.

Both the linearized transfer function method and the distributed components modelling method of mathematical modelling are addressed. For convenience in obtaining the time domain response analysis, the transmission line is usually considered as a lumped mass of no order differential equation in order to develop the linearized transfer function of the whole system. A new technique of quasi-explicit method, using finite difference method (FDM) for the distributed parameter line modelling, is explored and the results are compared with those of the explict method of characteristics. A new coupling method is explored in which the line dynamics of the partial differential equation is coupled with other distributed, lumped parameter hydraulic components. Using the lumped parameter transmission line modelling, another linearized transfer function is obtained for the frequency response analysis. Using the distributed components modelling to determine the time domain responses of the system, the system can then be identified using pulse testing. The

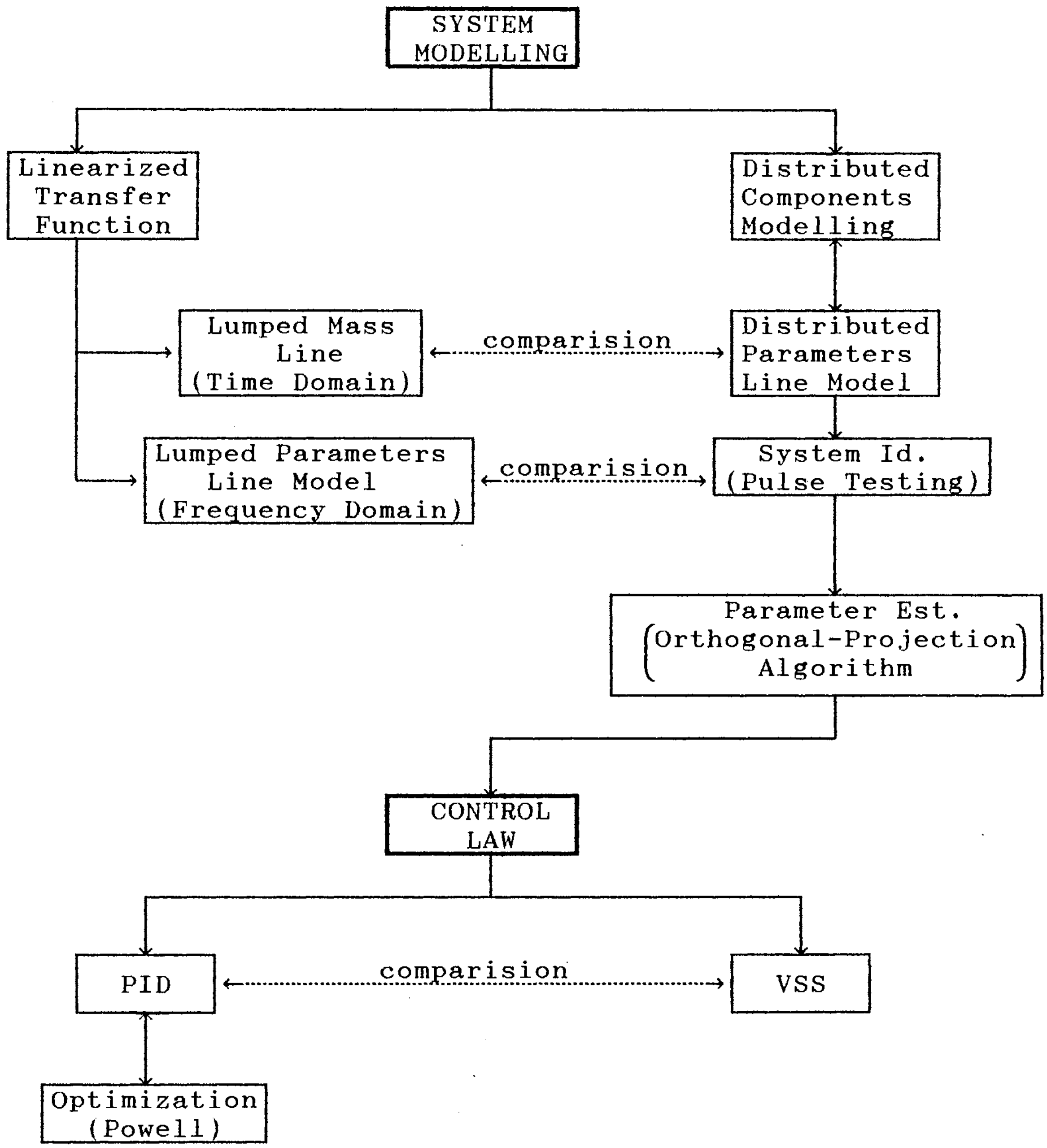

Figure 1. Computer Simulation Step Diagram

results of the system identification is compared with those of the frequency response analysis of linearized transfer function. After the order of the system is identified, the parameters are estimated by Orthogonal Projection Algorithm in order to apply the control theory.

Finally, control algorithms are applied to control the line dynamics in the electrohydraulic servovalve-cylinder system. The conventional PID controller is considered while the optimum controller gains are obtained using the Powell's optimization method and Ziegler-Nichols method. Variable Structure System (VSS) control algorithm is applied as well.

Mathematical Modelling of System

The basic servovalve-cylinder system is shown in Figure 2. There are two methods of mathematical modelling used in the lines between a servovalve and a cylinder. One is the lumped parameter modelling method and the other is distributed parameter modelling. The system equations are developed in the form of linearized transfer function considering the lines first as a lumped mass and later as a form of distributed components equations for time domain response analysis using a digital computer.

Line Modelling

In order to derive an equation of unsteady state fluid flow in a straight line, the following assumption can be

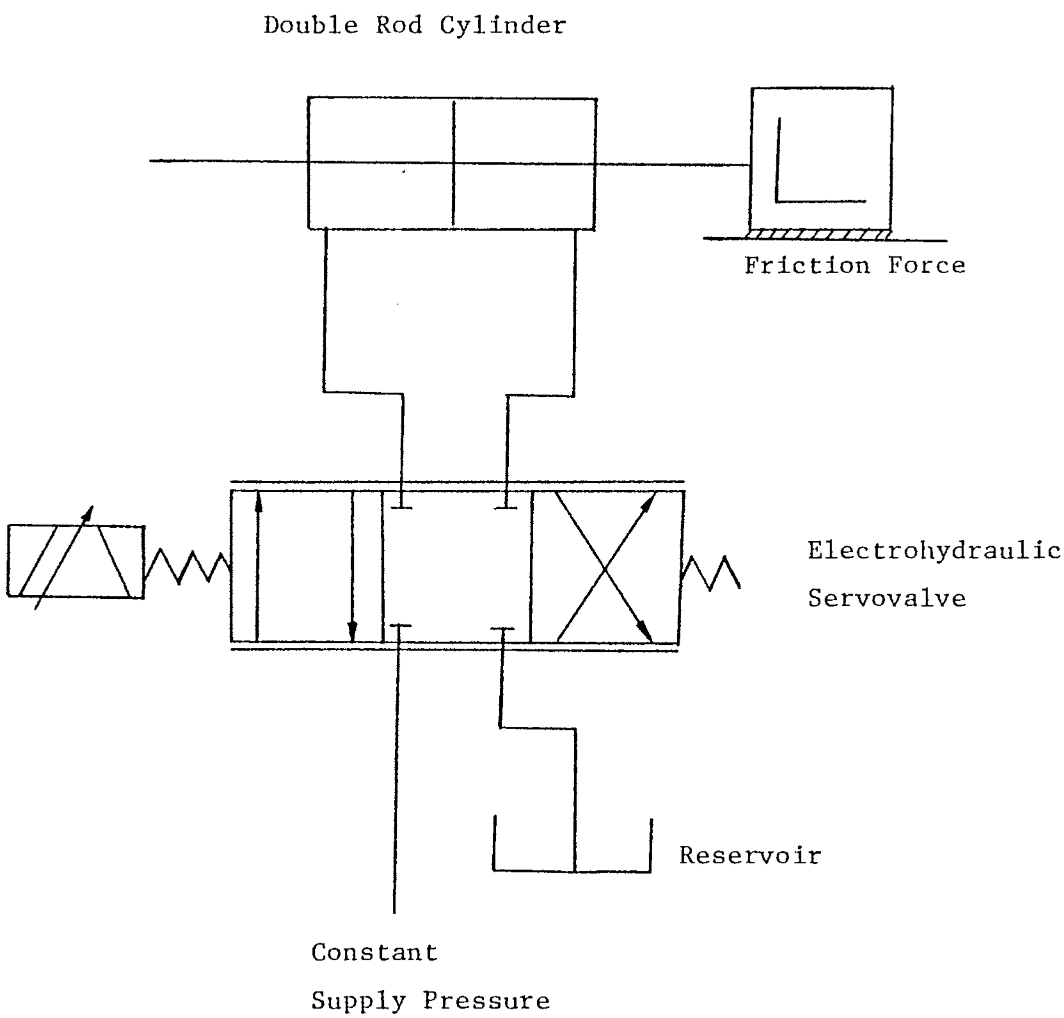

Figure 2. Basic System

made.

(1) The elasticity of the pipe walls may be ignored when compared with the compressibility of the fluid. Generally it is small for small-diameter pipes.

(2) Small temperature variation allows the fluid viscosity to be considered as constant.

(3) Due to rotational symmetry, both the velocity and the alteration of all dependent variables in the Φ (circumferential) direction are negligible.

(4) The flow is laminar.

(5) Comapratively long length and short inner diameter of the pipe.

It is convenient to use a cylindrical coordinats whose x-axis is identified with the center line of the pipe as shown in Figure 3. Complete Navier-Stokes' equations of cylindrical coordinates are given by Pai [18]. Using the preceding assumptions, equations for the deviations can be simplified in the following manner.

Equation of motion : x-direction

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial r}\right) = -\frac{\partial p}{\partial x}$$

$$+ \mu\left[\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{3}\frac{\partial}{\partial x}\left(\frac{\partial v}{\partial r} + \frac{v}{r}\right)\right] \qquad (1)$$

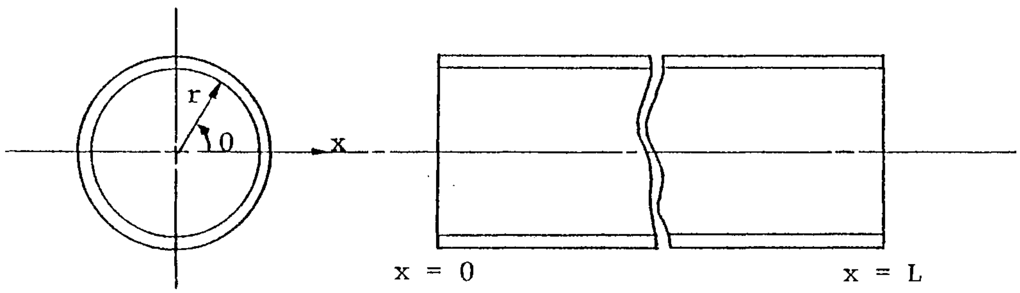

Figure 3. Transmission Line Coordinates

Equation of motion : r-direction

$$\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial r}\right) = -\frac{\partial p}{\partial r}$$

$$+ \mu\left[\frac{4}{3}\frac{\partial^2 v}{\partial r^2} + \frac{4}{3}\frac{1}{r}\frac{\partial v}{\partial r} - \frac{4}{3}\frac{v}{r^2} + \frac{\partial}{\partial x}\left(\frac{1\partial u}{3\partial r} + \frac{\partial v}{\partial x}\right)\right] \quad (2)$$

Continuity equation :

$$\frac{\partial \rho}{\partial t} + \rho\frac{\partial v}{\partial r} + \rho\frac{v}{r} + \rho\frac{\partial u}{\partial x} + v\frac{\partial \rho}{\partial r} + u\frac{\partial \rho}{\partial x} = 0 \quad (3)$$

Equation of state for a liquid :

$$\frac{\partial \rho}{\rho} = \frac{\partial p}{\beta e} \quad (4)$$

The following assumptions can be made also:

(i) since $u >> v$, we neglect equation (2). Neglecting this equation implies that the pressure is constant across a section of the tube and the pressure becomes a function only of x and t.

(ii) As proved in D'souza and Oldenburger's study [19], $\partial u/\partial t >> u(\partial u/\partial x)$ and $\partial u/\partial t >> v(\partial u/\partial r)$. Hence, in equation (1), the nonlinear convective acceleration terms on the left side may be ignored.

(iii) Also, all viscous terms can be ignored with the exception of $\partial^2 u/\partial r^2$ and $(1/r)\partial u/\partial r$.

(iv) In equation (3), the terms $v(\partial\rho/\partial r)$ and $u(\partial\rho/\partial x)$ can be ignored when compared with the other terms.

Using these preceding assumptions, equations (1) to (4) reduce to the following two differential equations :

$$\rho \frac{\partial u}{\partial t} = -\frac{\partial p}{\partial x} + \mu\left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r}\right) \tag{5}$$

$$\frac{1}{\beta_e}\frac{\partial p}{\partial t} + \frac{\partial v}{\partial r} + \frac{v}{r} + \frac{\partial u}{\partial x} = 0 \tag{6}$$

If one assumes there is no flow toward the radial direction, the above equation can be written as follows [20]:

$$\frac{\partial p}{\partial x} = -ZQ \tag{7}$$

$$\frac{\partial Q}{\partial x} = -Yp \tag{8}$$

where, the series impedance, Z,and the shunt admittance, Y are respectively:

$$Z = \frac{128\mu}{\pi\ Di^4} + \frac{\rho}{A}s = R + sL \tag{9}$$

$$Y = \frac{A}{\beta_e}s = sC \tag{10}$$

According to the wave propagation theory, three models are available for laminar flow. These models are defined as:

(1) Lossless line model, i.e. inviscid model.

(2) Average friction model, i.e., lossless line model plus laminar pipe friction effects.

(3) Distributed friction model which also considers heat transfer effects between the fluid and the pipe wall.

And, equations (7) through (10) corresponds to the average friction model.

Lumped Parameter Line Modelling

For the analysis of transmission lines, the fluid momentum and continuity equation are applied to an undetermined length of line and then integrated to produce the required solution when boundary conditions are included. Therefore, the approach is to consider a small length of line Δx as shown in Figure 4.

Equations (7) and (8) are then combined to form the wave equation:

$$\frac{\partial^2 P}{\partial x^2} = ZYP \tag{11}$$

$$\frac{\partial^2 Q}{\partial x^2} = ZYQ \tag{12}$$

Define the boundary conditions as:

at $x = 0$, $P = P_1$ and $Q = Q_1$

at $x = l$, $P = P_2$ and $Q = Q_2$

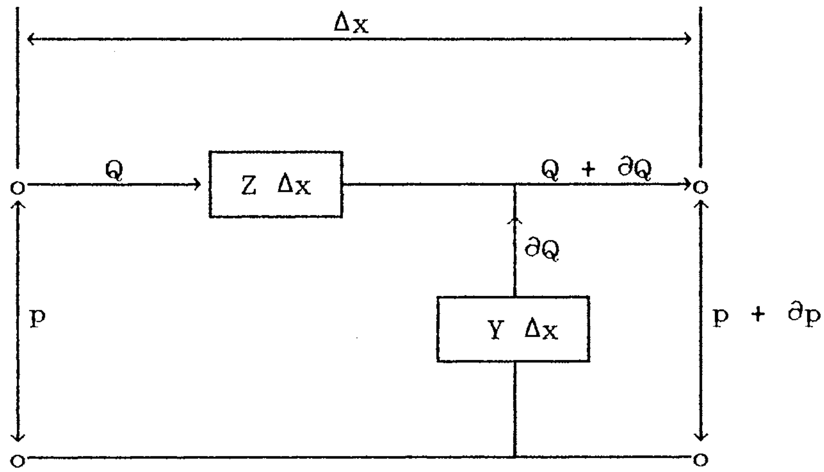

Figure 4. Element of a transmission line

Then the solution to equation (11) and (12) becomes,

$$P = \frac{(P_1 + Z_c Q_1)e^{-\Gamma x}}{2} + \frac{(P_1 - Z_c Q_1)e^{\Gamma x}}{2} \qquad (13)$$

$$Q = \frac{(Q_1 + P_1/Z_c)e^{-\Gamma x}}{2} + \frac{(Q_1 - P_1/Z_c)e^{\Gamma x}}{2} \qquad (14)$$

where the parmeters Z_c and Γ are defined as:

characteristic impedence, $Z_c = \sqrt{Z/Y}$ (15)

propagation constant, $\Gamma = \sqrt{ZY}$ (16)

By rearranging equation (13) and (14), the matrix hyperbolic form at the end of the line is determined as follows:

$$\begin{bmatrix} P_2 \\ Q_2 \end{bmatrix} = \begin{bmatrix} \cosh(\Gamma x) & -Z_c \sinh(\Gamma x) \\ \dfrac{-\sinh(\Gamma x)}{Z_c} & \cosh(\Gamma x) \end{bmatrix} \begin{bmatrix} P_1 \\ Q_1 \end{bmatrix} \qquad (17)$$

This four-pole equation has been commonly used to determine lumped parameter line modelling since Oldenburger and Goodson [21].

<u>Distributed Parameter Modelling</u>

<u>Finite difference method.</u>: There exist several methods of solving the preceding equations, (7) through (10), in time domain. Either implicit finite difference approximation or explicit method of characteristics could be used. However this study explores the new quasi-explicit

difference equations, using finite difference method in an attempt to simulate the whole hydraulic system by distributed components expressions.

If backward difference method and forward difference method are applied to express the time derivatives of P and Q respecively such that:

$$\frac{\partial P_i}{\partial t} = \frac{P_i^{j} - P_i^{j-1}}{\Delta t} \qquad (18)$$

$$\frac{\partial Q_i}{\partial t} = \frac{Q_i^{j+1} - Q_i^{j}}{\Delta t} \qquad (19)$$

Then the central difference method can be applied for the displacement derivatives, $\partial P/\partial x$ and $\partial Q/\partial x$ except at the boundary points where forward or backward difference method are applied. Thus:

for i = 2 ... n-1

$$P_i^{j} = -\frac{ra}{2C}\left(Q_{i+1}^{j} - Q_{i-1}^{j} \right) + P_i^{j-1} \qquad (20)$$

$$Q_i^{j+1} = -\frac{ra}{2L}\left(P_{i+1}^{j} - P_{i-1}^{j} \right) + Q_i^{j} - \frac{R\,\Delta t}{L}\,Q_i^{j} \qquad (21)$$

for i = 1

$$P_1^{j} = -\frac{ra}{C}\left(Q_2^{j} - Q_1^{j} \right) + P_1^{j-1} \qquad (22)$$

$$Q_1^{j+1} = -\frac{ra}{L}\left(p_2^j - p_1^j \right) + Q_1^j - \frac{R\ \Delta t}{L} Q_1^j \qquad (23)$$

for i = n

$$p_n^j = -\frac{ra}{C}\left(Q_n^j - Q_{n-1}^j \right) + p_n^{j-1} \qquad (24)$$

$$Q_n^{j+1} = -\frac{ra}{L}\left(p_n^j - p_{n-1}^j \right) + Q_n^j - \frac{R\ \Delta t}{L} Q_n^j \qquad (25)$$

where i represents the node number of n-1 line section and j represents the time step number, and r_a equals $\Delta t/\Delta x$.

Method of Characteristics. [22]: The characteristic grids are shown in Figure 5. Two characteristic equations must be written for each line. The line is divided n sections, and a time increment, Δt, is chosen such that,

$$C_o = \frac{\Delta X}{\Delta t} \qquad (26)$$

Thus, the time/displacement characteristic grid appears as shown in Figure 5. Using the finite difference approximation then enables the fundamental momentum equation (7) to be written for a particular point P in the grid:

Then, the fundamental momentum equation (7) becomes:

the positive characteristic C^+

$$P_P - P_A + C_o L(Q_P - Q_A) + R(Q_P + Q_A)\frac{\Delta X}{4}$$

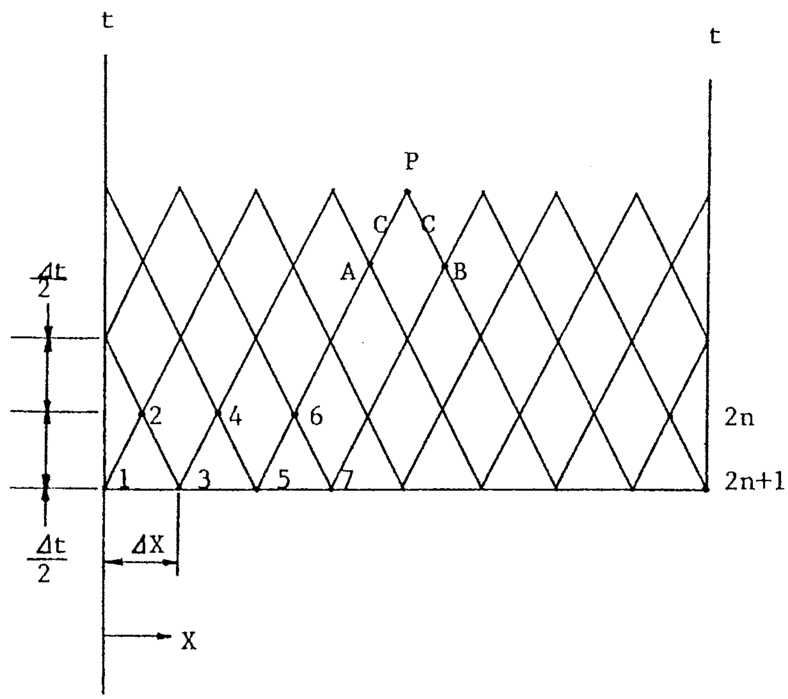

Figure 5. Characteristic Grids in A Line

the negative characteristic C^-

$$P_P - P_B - C_oL(Q_P-Q_B) - R(Q_P+Q_B)\frac{\Delta X}{4}$$

Using the preceding two statements, the pressure P_P and the flow rate Q_P at the desired point can be written in the following form:

$$P_P = \frac{P_A + P_B}{2} + \frac{1}{2}\left(C_oL - \frac{R\ \Delta X}{4}\right)(Q_A - Q_B) \qquad (27)$$

$$Q_P = \frac{\left(C_oL - \frac{R\ \Delta X}{4}\right)(Q_A + Q_B) + (P_A - P_B)}{2\left(C_oL + \frac{R\ \Delta X}{4}\right)} \qquad (28)$$

Equations (27) and (28) are used to determine the first time step of $\Delta t/2$. This enables the pressures and flows to be calculated for all stations, i is 2 to 2n. For the next time step of $\Delta t/2$, the equations are again used for i is 3 to 2n -1. The input boundary condition combined with the C^- characteristic allows calculation of the conditions at point 1. The load boundary condition together with the C^+ characteristic then allows calculation of the conditions at point (2n + 1). For example if the flow rate at inlet port and the pressure at outlet port are given, the pressure at inlet and the flow rate at outlet is determined in the following manner:

$$P_1 = P_2 + C_oL(Q_1 - Q_2) + R(Q_1 + Q_2)\frac{dX}{4} \quad (29)$$

$$Q_{2n+1} = \frac{\left(C_oL - R\frac{dX}{4}\right)Q_{2n} - (P_{2n+1} - P_{2n})}{C_oL + R\frac{dX}{4}} \quad (30)$$

System Modelling

Linearized transfer function

The functions of most components in a hydraulic system can be expressed in the nonlinear algebraic equations. One example is the pressure-flow characteristic of a servovalve. In making a dynamic analysis and applying control algorithms, it is necessary that the nonlinear equations be linearized. Merritt [23] revealed that the following transfer function equation can represent a servovalve-cylinder system. In this equation, the viscous damping coefficient of the piston and load is ignored because it is usually much smaller than the value of other parameters.

$$x_p = \frac{\frac{K_q}{A_p}x_v - \frac{K_{ce}}{A_p^2}\left(1 + \frac{V_t}{4\beta_e K_{ce}}s\right)F_L}{s\left(\frac{s^2}{\omega_h^2} + \frac{2\delta_h}{\omega_h}s + 1\right)} \quad (31)$$

By using the perturbation method applied by Watton [15], the system equation can be derived in another form as follows.

$$x_p = \frac{\frac{K_i}{A_p} i - \frac{K_{ci}}{A_p^2}\left(1 + \frac{V_t}{4\beta_e K_{ci}} s\right) F_L}{s\left(\frac{L_c C}{2} s^2 + \frac{L_c}{2R_t} s + 1\right)} \tag{32}$$

First, the following terms are defined:

$$A_1 = \frac{1}{\omega_h^2} \tag{33}$$

$$A_2 = \frac{2\delta_h}{\omega_h} \tag{34}$$

$$B_1 = \frac{K_q}{A_p} \tag{35}$$

$$B_2 = \frac{K_{ce}}{A_p^2} \tag{36}$$

$$B_3 = \frac{V_t}{4A_p^2\beta_e} \tag{37}$$

Then, equation (31) is expressed as,

$$x_p = \frac{\frac{B_1}{A_1} x_v - \left(\frac{B_2}{A_1} + \frac{B_3}{A_1} s\right) F_L}{s^3 + \frac{A_2}{A_1} s^2 + \frac{A_3}{A_1} s + \frac{A_4}{A_1}} \tag{38}$$

Since this is a linearized equation, the total piston displacement, x_p, is a sum of displacement due to valve displacement and that due to friction force, F_L. Therefore,

$$x_p = x_{p_1} + x_{p_2} \tag{39}$$

where

$$xp_1 = \frac{\frac{B1}{A1} x_v}{s^3 + \frac{A2}{A1} s^2 + \frac{1}{A1} s} \tag{40}$$

$$xp_2 = \frac{-\left(\frac{B2}{A1} + \frac{B3}{A1} s \right) F_L}{s^3 + \frac{A2}{A1} s^2 + \frac{1}{A1} s} \tag{41}$$

Equation (40) is expressed in a matrix form using state variables as follow.

$$\begin{bmatrix} \frac{d\alpha 1}{dt} \\ \frac{d\alpha 2}{dt} \\ \frac{d\alpha 3}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -\frac{1}{A1} & -\frac{A2}{A1} \end{bmatrix} \begin{bmatrix} \alpha 1 \\ \alpha 2 \\ \alpha 3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{B1}{A1} \end{bmatrix} x_v \tag{42}$$

Also equation (41) can be expressed in the following manner:

$$\begin{bmatrix} \frac{d\beta_1}{dt} \\ \frac{d\beta_2}{dt} \\ \frac{d\beta_3}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -\frac{1}{A_1} & -\frac{A_2}{A_1} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{B_3}{A_1} \\ \frac{1}{A_1}\left(\frac{A_2 B_3}{A_1} - B_2\right) \end{bmatrix} F_L \tag{45}$$

$$xp_2 = [\ 1\ \ 0\ \ 0\ ]\ [\ \beta_1\ \ \beta_2\ \ \beta_3\ ]^T \tag{46}$$

where the state variables, β_1, β_2 and β_3 are defined :

$$\beta_1 = xp_2$$

$$\beta_2 = \frac{dxp_2}{dt}$$

$$\beta_3 = \frac{d^2xp_2}{dt^2} + \frac{B_3}{A_1} F_L \tag{47}$$

From equations, (32) and(47), the following linearized state variable equations can be derived.

$$\begin{bmatrix} \frac{d\alpha_1}{dt} \\ \frac{d\alpha_2}{dt} \\ \frac{d\alpha_3}{dt} \\ \frac{d\beta_1}{dt} \\ \frac{d\beta_2}{dt} \\ \frac{d\beta_3}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{A_1} & -\frac{A_2}{A_1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -\frac{1}{A_1} & -\frac{A_2}{A_1} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{B_1}{A_1} & 0 \\ 0 & 0 \\ 0 & -\frac{B_1}{A_1} \\ 0 & \frac{1}{A_1}\left(\frac{A_2}{A_1} B_3 - B_2 \right) \end{bmatrix} \begin{bmatrix} x_v \\ F_L \end{bmatrix} \qquad (48)$$

$$x_p = [\ 1\ 0\ 0\ 1\ 0\ 0\ ]\ [\ \alpha_1\ \alpha_2\ \alpha_3\ \beta_1\ \beta_2\ \beta_3\ ]^T \qquad (49)$$

Distributed components modelling

The system can be considered as a composite of components as shown in Figure 6. The function of each component can be expressed in the following manner. Using these equations, the dynamic response of the entire system can be obtained [24,25].

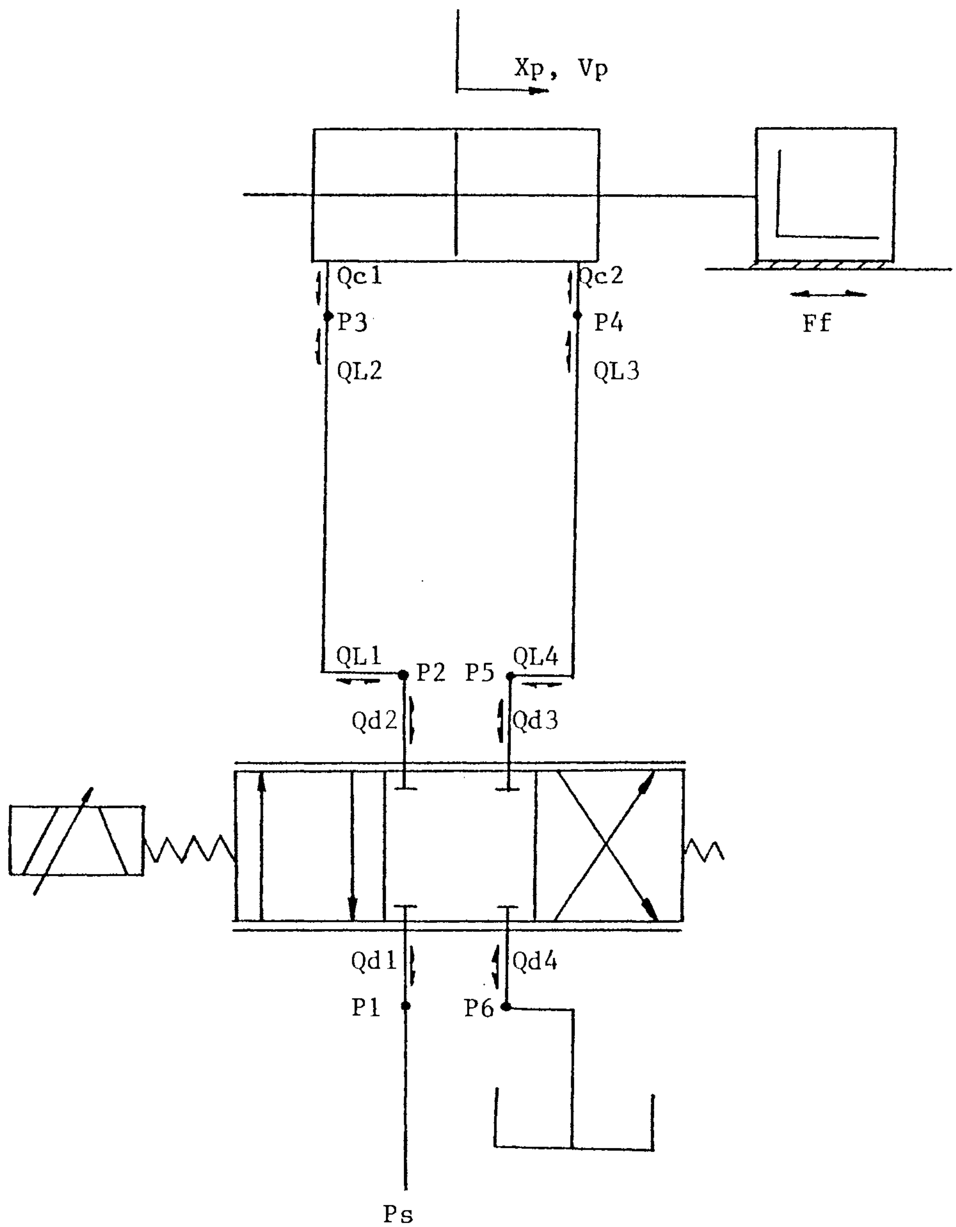

Figure 6. Distributed Components Modeling

for Pressure Source : $P1 = Ps$ (50)

for Servo Valve :

$$Q_{d1} = -\ C_{d1}\ X_v\ W_d \sqrt{\frac{2(P2-P1)}{\rho}} \qquad (51)$$

$$Q_{d3} = \ C_{d2}\ X_v\ W_d \sqrt{\frac{2(P6-P5)}{\rho}} \qquad (52)$$

$$Q_{d2} = -\ Q_{d1} \qquad (53)$$

$$Q_{d4} = -\ Q_{d3} \qquad (54)$$

for Cylinder :

$$Q_{c1} = -[A_pV_p + C_{f1}(P3-P4) + C_{f2}P_3] \qquad (55)$$

$$Q_{c2} = A_pV_p + C_{f1}(P3-P4) - C_{f2}P_4 \qquad (56)$$

$$F_c = A_p(P3-P4) \qquad (57)$$

for Friction Force :

$$F_f = -\ M_{tfc} \qquad (58)$$

for Supply & Return Line : Determine the flow rate QL_1, QL_2 for supply line and QL_3, QL_4 for return line using the equations of line dynamics.

Then, the total volume change at points 2 to 5 and the force acting on the piston at any instance can be obtained in the following manner:

at point 2 : $\Delta Q2 = Q_{d2} + Q_{L1}$ (59)

at point 3 : $\Delta Q3 = Q_{L2} + Q_{c1}$ (60)

at point 4 : $\Delta Q4 = Q_{c2} + Q_{L3}$ (61)

at point 5 : $\Delta Q5 = Q_{L4} + Q_{d3}$ (62)

at piston : $\Delta F = F_c + F_f$ (63)

Therefore, the pressure deviation occurres at points 2 through 5. The equations determine the acceleration and the speed of the piston.

at point 2 : $\frac{dP_2}{dt} = \frac{\beta_e}{V_2} \Delta Q_2$ (64)

at point 3 : $\frac{dP_3}{dt} = \frac{\beta_e}{V_3} \Delta Q_3$ (65)

at point 4 : $\frac{dP_4}{dt} = \frac{\beta_e}{V_4} \Delta Q_4$ (66)

at point 5 : $\frac{dP_5}{dt} = \frac{\beta_e}{V_5} \Delta Q_5$ (67)

at piston : $\frac{dV_p}{dt} = \frac{\Delta F}{M_s}$ (68)

$\frac{dX_p}{dt} = V_p$ (69)

The above equations, (50) through (69), can be solved using the Runge-Kutta 4th order formula.

System Identification and Parameter Estimation

In order to apply the control algorithm, the system must first be identified. The transfer function of the system is not known if the method of distributed components modelling is used. In order to identify the system, the direct sine-wave testing method and the pulse testing method have been used [26]. Direct sine-wave testing is a useful method of obtaining precise dynamic data. Damping, time constants, and the order of the system can all be accurately determined. The main disadvantage of direct sine-wave testing is that it can be time consuming. Pulse testings can be performed in only a fraction of the time that direct sine-wave tests require. However pulse testing analysis is particularly difficult due to the oscillatory behavior of the response at high values of frequency. In this study pulse testing method was used. After the order of the system is identified, the parameters are estimated using the Orthogonal Projection Algorithm [27].

System Identification

Pulse Testing

Consider a system with an input $I(t)$ and an output $O(t)$ as shown in Figure 7. Where the transfer function is:

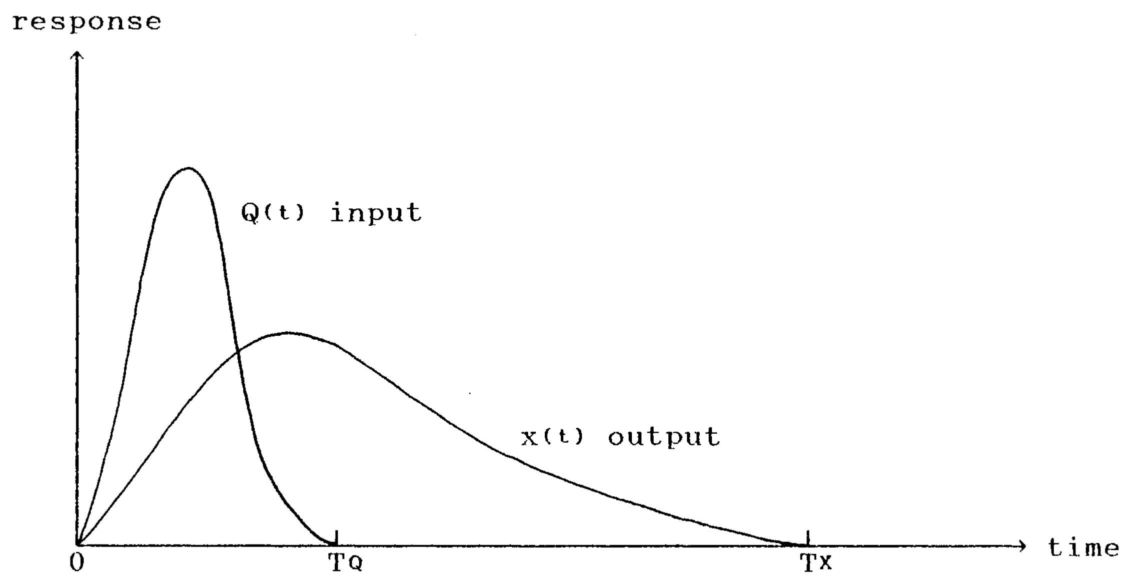

Figure 7. Pulse Test Input And Output Curves

$$G(s) = \frac{O(s)}{I(s)} \tag{70}$$

Using the definition of Laplace Transformation and substituting $s = i\omega$,

$$G(i\omega) = \frac{\int_0^\infty O(t)\, e^{-i\omega t}\, dt}{\int_0^\infty I(t)\, e^{-i\omega t}\, dt} \tag{71}$$

The numerator is the Fourier transformation of the time function O(t). The denominator is the Fourier transformation of the time function I(t). Therefore,

$$G(i\omega) = \frac{(AC + BD) + i(AD - BC)}{C^2 + D^2} \tag{72}$$

where $$A = \int_0^{to} O(t)\cos(\omega t)dt \tag{73}$$

$$B = \int_0^{to} O(t)\sin(\omega t)dt \tag{74}$$

$$C = \int_0^{ti} I(t)\cos(\omega t)dt \tag{75}$$

$$D = \int_0^{ti} I(t)\sin(\omega t)dt \tag{76}$$

Parameter Estimation

Orthogonal Projection Algorithm

The Orthogonal Projection Algorithm for parameter estimation is expressed in the following manner:

$$\hat{E}(t) = \hat{E}(t-1) + \frac{P(t-2)\ \Phi(t-1)}{1 + \Phi(t-1)^T P(t-2)\Phi(t-1)} [y(t) - \Phi(t-1)^T \hat{E}(t-1)] \quad (77)$$

$$P(t-1) = P(t-2) - \frac{P(t-2)\Phi(t-1)\Phi(t-1)^T P(t-2)}{1 + \Phi(t-1)^T P(t-2)\Phi(t-1)} \quad (78)$$

where $y(t)$ denotes the (scalar) system output at time t

$\Phi(t-1)$ denotes a vector that is a linear or nonlinear function of { y(t-1), y(t-2),... } and { u(t-1), u(t-2),...}

$\hat{E}(t)$ denotes estimated parameters vector

$P(t-1)$ projection operator vector for algorithm

with the initial estimate $\hat{E}(1)$ given and P(0) begine equal to the unit matrix.

Control Scheme

The overview of a control scheme for the system is shown in Figure 8. Variable Structure System (VSS) Control Algorithm is used for establishing the control law, since it is one of the simplest and quickest control algorithms and it is insensitive to parameter variations [28]. The referenced model signal is given in such a way that the error, between the output response of the system and the model signal, manipulates the control law signaling the system response to follow the model signal as soon as possible. To compare the response using VSS control to the response using conventional Proportional-Integrational-Derivative (PID) control, the PID control algorithm was derived. The optimum gains of the PID controller can be obtained by using the Powell optimization method and Zigler-Nichols method.

PID Control

The PID control scheme is shown in Figure 9. The error signal is generated by determining the difference between the feedback signal and the input signal.

Discrete PID Controller Design

An anlalog PID control law, written in the form

$$m(t) = k_c \left[e(t) + \frac{1}{T_i} \int_{-\infty}^{t} e(t) + T_d \frac{de(t)}{dt} \right] \tag{79}$$

can be written in the finite difference form

$$m_n = k_c \left[e_n + \frac{1}{T_i} \sum_{j=-\infty}^{n} e_j \, T + T_d \frac{e_n - e_{n-1}}{T} \right] \tag{80}$$

But, the sampling time T is so small that the derivative term in equation (80) causes a total gain over the

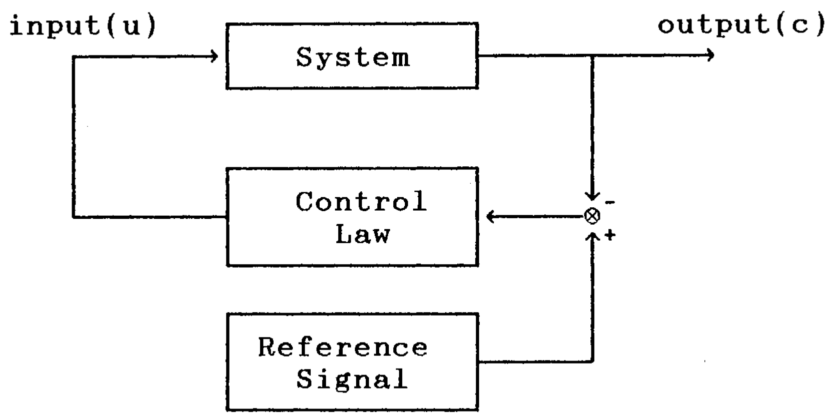

Figure 8. Variable Structure System Control Scheme

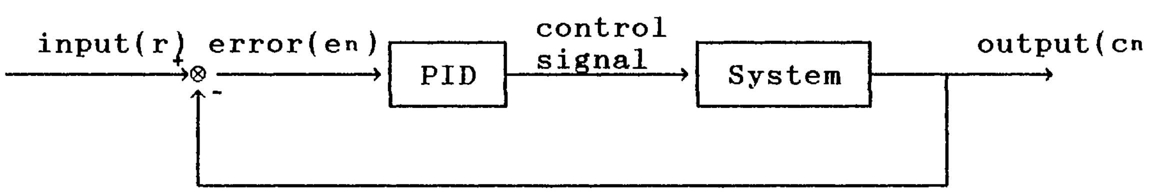

Figure 9. PID Control Scheme

saturation value for a sudden set-point change or at the instance of starting the test. The denominator T in the derivative term should be eliminated [29].

By applying equation (80) to the sample(n-1), and substracting the result from both sides of the equation (80), an output increment can be determined.

$$m_n - m_{n-1} = \Delta m_n \qquad (81)$$

$$= k_c\left[(e_n - e_{n-1}) + \frac{T}{T_i} e_n + \frac{T_d}{T}(e_n - 2e_{n-1} + e_{n-2})\right]$$

Substituting $e_n = r_n - c_n$ (82)

into equation (81) results in the following equation:

$$\Delta m_n = k_c[(C_{n-1} - C_n)$$

$$+ k_i(r_n - C_n) + k_d(2C_{n-1} - C_{n-2} - C_n)] \qquad (83)$$

where

$$k_i = \frac{T}{T_i} \qquad k_d = \frac{T_d}{T} \qquad (84)$$

Therefore, the algorithm can be written in such a way that the P, I and D adjustments are independent:

$$\Delta m_n = K_p(C_{n-1} - C_n) \qquad (85)$$

$$+ K_i(r_n - C_n) + K_d(2C_{n-1} - C_{n-2} - C_n)$$

<u>Powell's Optimization Method</u>

Powell's optimization algorithm is designed to determine the minimum of an unconstrained, multivariable, nonlinear function [30], and can be expressed in the following manner:

$$\text{Minimize} \quad J(x_1, x_2, x_3, \ldots. x_n).$$

The logic concept of Powell optimization method is revealedin Figure 10. The integral of the square of the error (ISE) is used as a perfomance index function, J. Xu and He addressed that the ISE criterion is the most desirable performance index in fluid power application in [31]. The optimizing variables are:

$$x_1 = K_p, \; x_2 = K_i, \; x_3 = K_d.$$

The definition of ISE is shown in Figure 11.

<u>Ziegler and Nichols Method</u> [32]

In order to usethe transient-response Ziegler and Nichols method, the steepest slope, R_s, and the delay time, L, are measured from a unit-step response of the open-loop system (see Figure 12). The gains for the PID controller are then obtained from Table I.

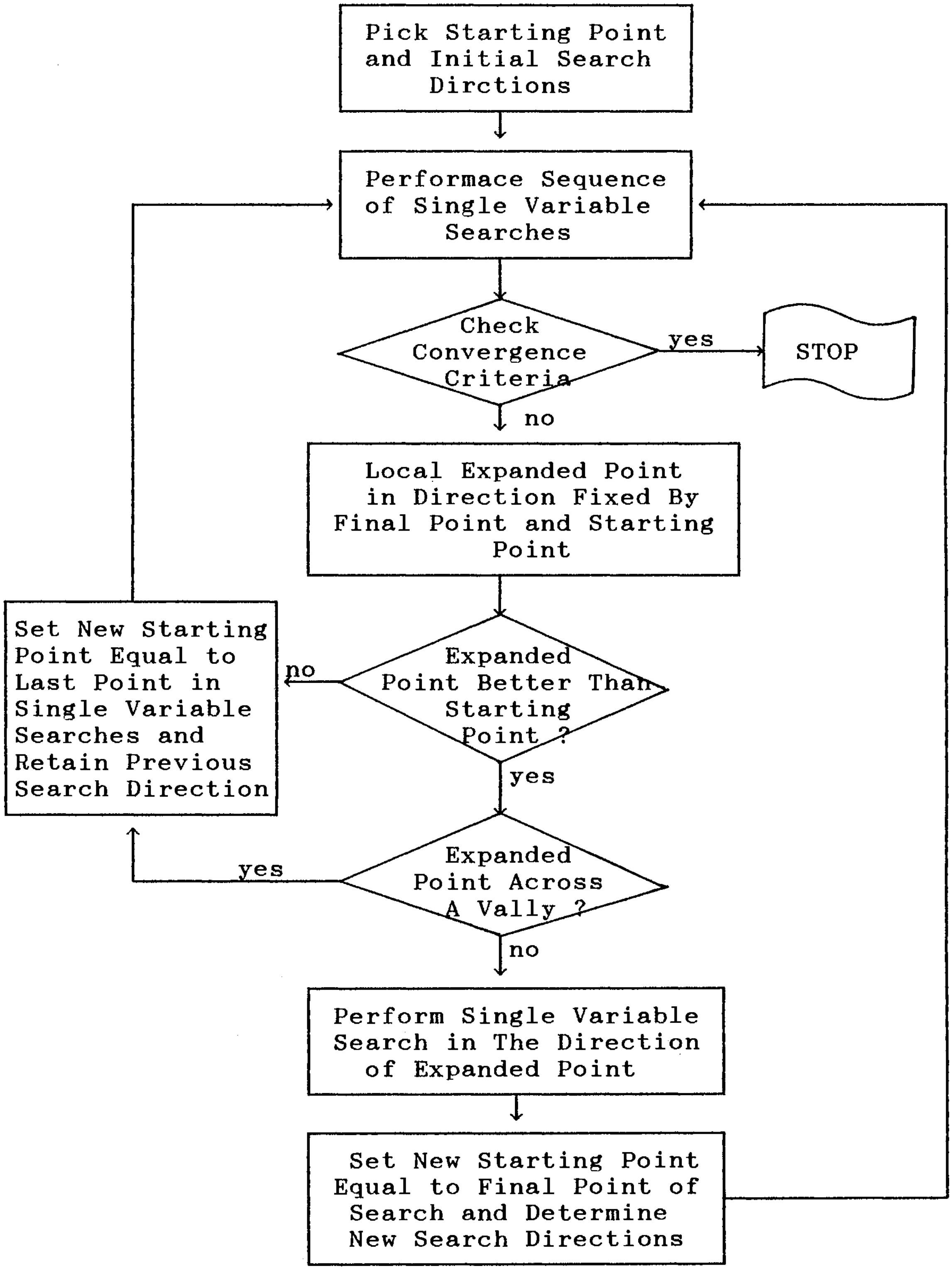

Figure 10. Powell Logic Diagram

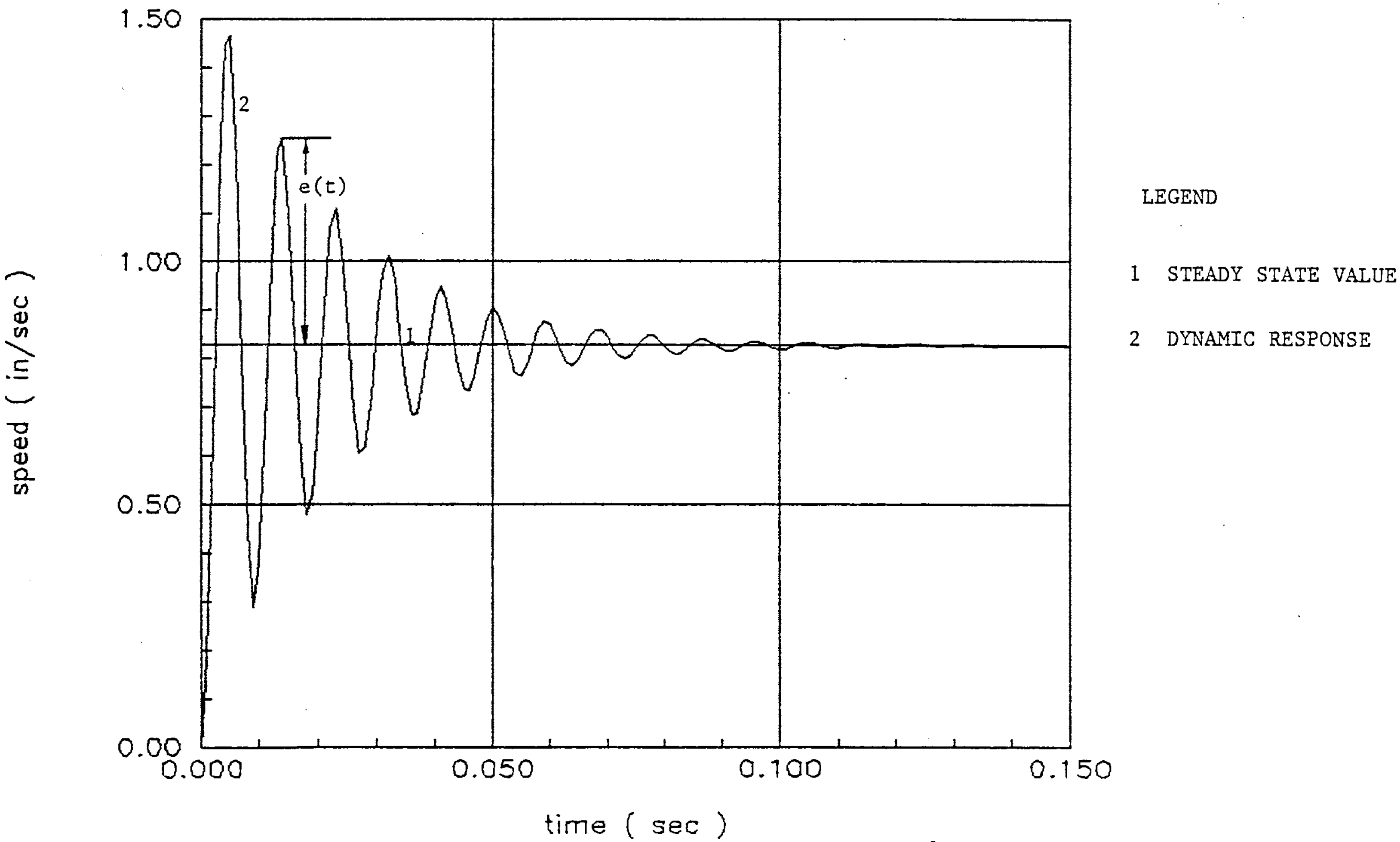

Figure 11. Integral of Square of the Error (ISE) = $\int_0^\infty e^2(t)\,dt$

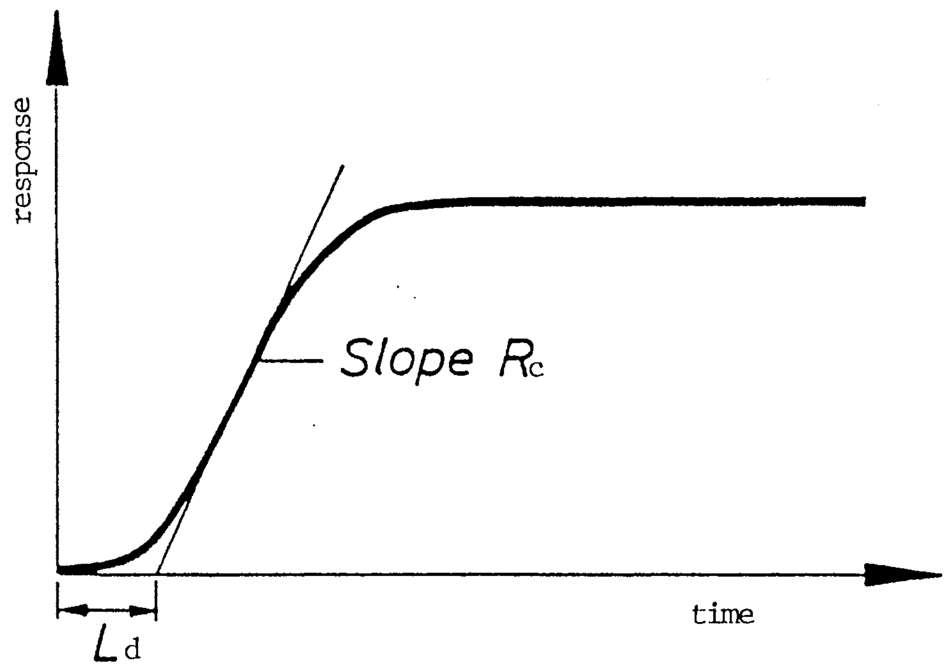

Figure 12. Transient Response for Ziegler-Nichols method

TABLE I

PID CONTROLLER GAIN BY ZIEGLER AND NICHOLS METHOD

Controller	Kp	Ki	Kd
P	$\frac{1}{R_s L_d}$	-	-
PI	$\frac{0.9}{R_s L_d}$	$\frac{0.3}{R_s L_d^2}$	-
PID	$\frac{1.2}{R_s L_d}$	$\frac{0.6}{R_s L_d^2}$	$\frac{0.6}{R_s}$

Variable Structure System Control

The main distinctive feature of VSS [33] is that changes can occur in the structure of the system during the transient process. The structure of a VSS is changed intentionally, in accordance with some preassigned algorithm or law of structural change; the times at which these changes occur (and the type of structure formed) are determined in accordance with the current value of error signal and its derivatives (see Figure 13). Until the response hits the sliding regime, linear feedback control dominates. The sliding regime has a property that the corresponding motion of the system is independent of changes in the plant parameters and of external

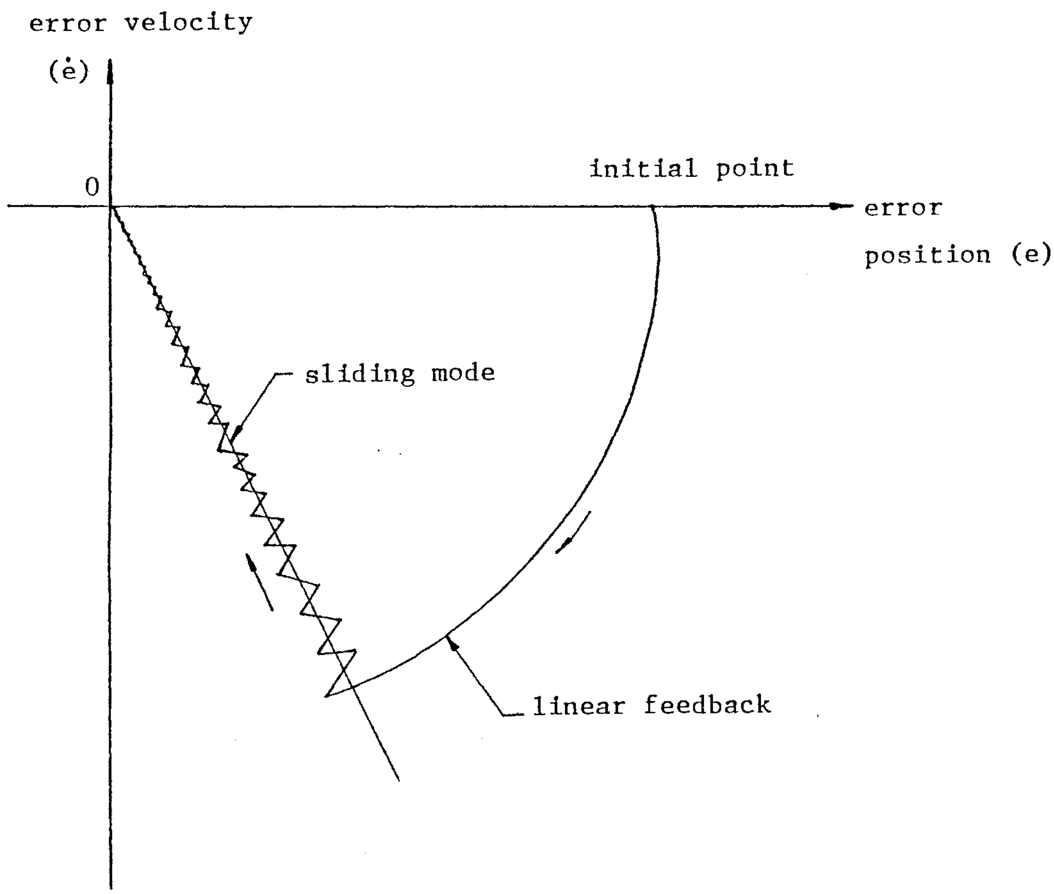

Figure 13. Variable Structure System Control

disturbances. Once the response hits the sliding regime, the response moves in a sliding regime toward the origin of the error and its derivatives. Using the changes introduced in the structure of the system, the typical problem for automatic control system, between static accuracy (stability, noise immunity) and speed of response (dynamic accuracy) can be resolved at the same time. The control algorithm of the VSS used for the system of no-disturbance and time-invariant parameters follows [34]:

The following differential equations describe the free motion of the system.

$$\dot{x}_i = x_{i+1}, \qquad i = 1,...,n-1$$

$$\dot{x}_n = -\sum_{i=1}^{n} a_i x_i + u \tag{86}$$

where x_1 is the difference between the reference input u and the output signal, a_i are constants.

Then, the sliding plane can be expressed as:

$$\sigma = \sum_{i=1}^{n} c_i x_i = 0 \qquad c_i : \text{constants}, \quad c_n = 1 \tag{87}$$

The control signal u is chosen as a piecewise linear function of x_i with discontinuous coefficients

$$u = \sum_{i=1}^{k} \Psi_i x_i + \delta_0 \operatorname{sgn}(\sigma) \qquad 1 \le k \le n-1 \tag{88}$$

$$\Psi = \begin{cases} a_i, & \text{if } x_i\sigma > 0 \\ b_i, & \text{if } x_i\sigma < 0 \end{cases}$$

a_i, b_i, δ_0 : constants

δ_0 : small positive scalar

$$\operatorname{sgn}(\sigma) = \begin{cases} +1, & \text{if } \sigma > 0 \\ -1, & \text{if } \sigma < 0 \end{cases}$$

The necessary and sufficient conditions for a sliding plane $\sigma = 0$ to exist are

$$a_i \geq c_{i-1} - a_i - c_i c_{n-1} + c_i a_n \tag{89}$$

$$b_i \leq c_{i-1} - a_i - c_i c_{n-1} + c_i a_n, \quad i = 1,\ldots,k$$

$$c_0 = 0$$

$$\frac{c_{i-1} - a_i}{c_i} = c_{n-1} - a_n, \qquad i = k+1,\ldots,n-1 \tag{90}$$

CHAPTER IV

DIGITAL SIMULATION RESULTS

Time Domain Analysis

Linearized Transfer Function with Lumped Mass of Line

Equations (48) and (49), expressed in a matrix form of lineralized transfer function, are solved using the typical hydraulic components values cited as below:

TABLE II

DATA FOR SIMULATION

Parameters	Symbol	Value	Unit
Bulk modulus	β_e	1.5×10^5	psig
Density of oil	ρ	0.78×10^{-4}	$lb\text{-}sec^2/in^4$
Absolute viscosity	μ	2.8×10^{-6}	$lb\text{-}sec/in^2$
Supply pressure	P_s	500	psig
Total oil volume except line	V_a	5	in^3
Total weight of piston and load	M_W	100	lbf
Friction Coefficient	F_c	0.1	-

TABLE II (Continued)

Parameters	Symbol	Value	Unit
Tube inner diameter	D_i	0.25	inch
Tube length	L_t	10	inch (short)
		100	inch (long)
Piston diameter	D_r	1.378	inch
Cylinder inner diameter	D_c	1.82	inch
Valve gain	K_i	0.0012	inch/mA
Orifice width in the valve	W_d	0.1	inch

It is assummed that the external and internal leakage in the cylinder is negligible. See Appendix A for computer program.

Figures 14 and 15 show the dynamic responses of piston as the current to the servovalve changes from 5 mA to 20 mA as the length of the transmission lines, both supply and return flow, coupled between the valve and the cylinder are 10 inches and 100 inches. The steady state values, damping ratios, and the undamped natural frequencies are measured in the Table III, IV and V.

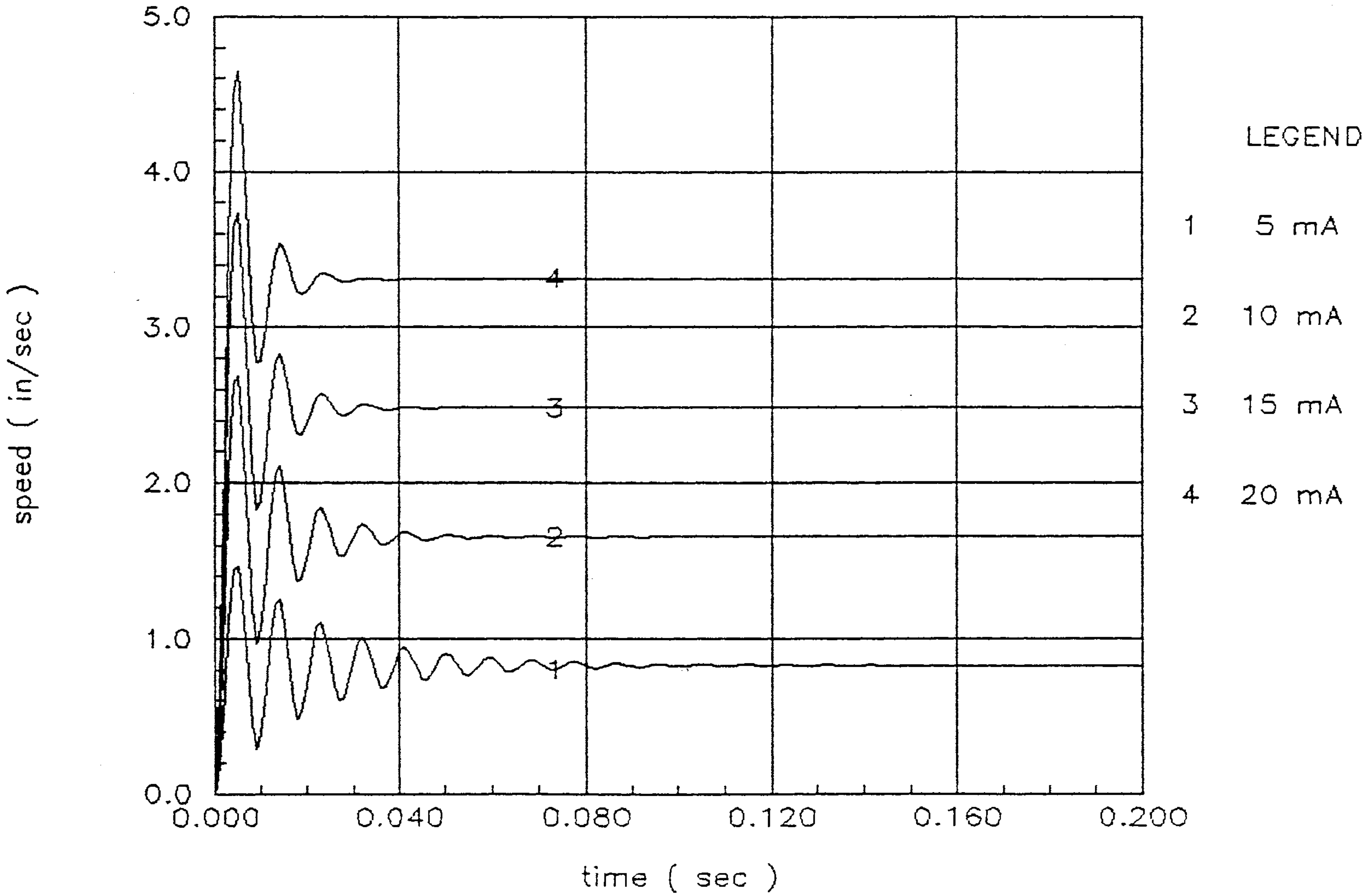

Figure 14. Dynamic Responses (L = 10 inch)

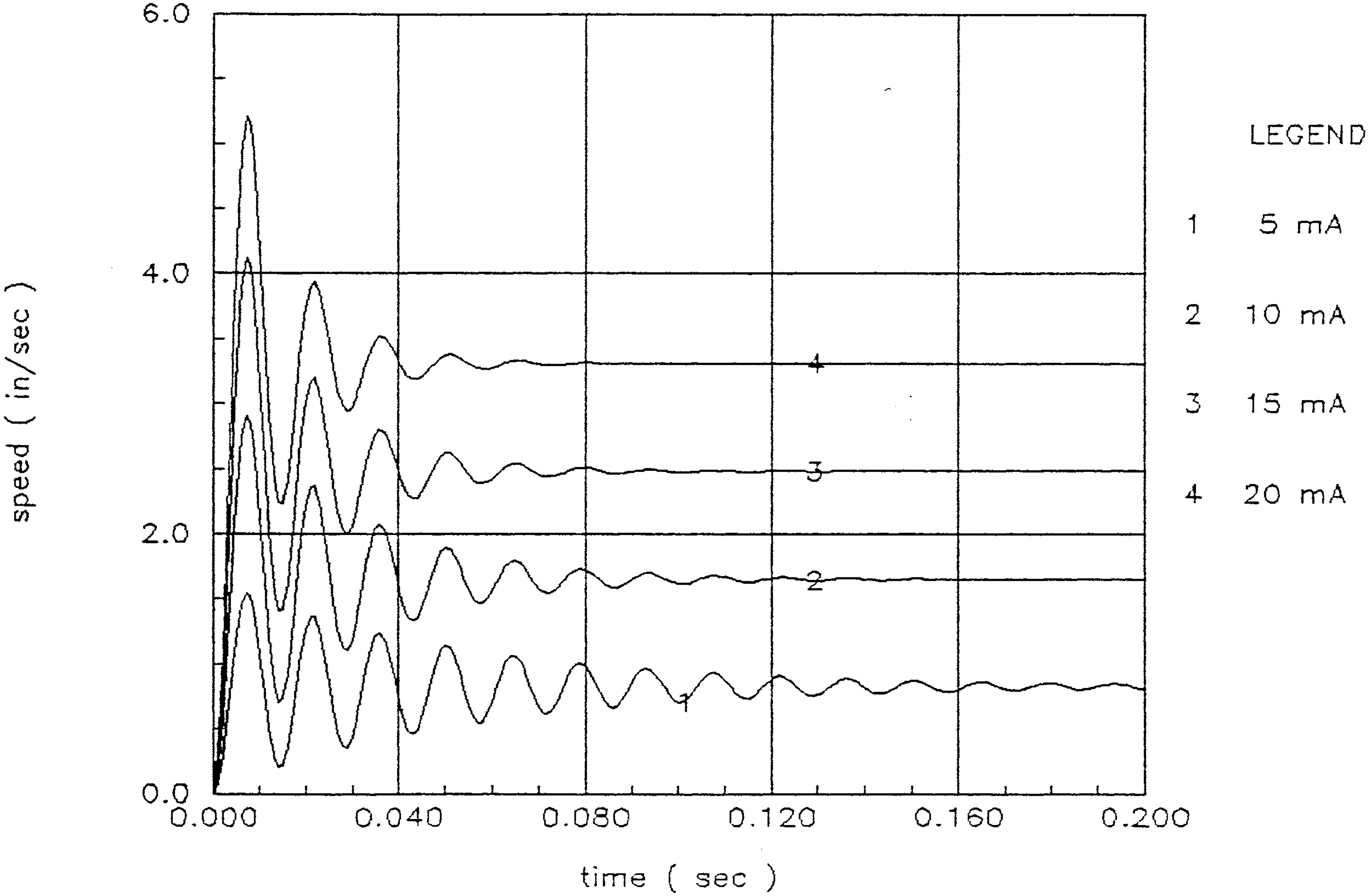

Figure 15. Dynamic Responses (L = 100 inch)

TABLE III

STEADY STATE VALUES
(in/sec)

I(mA) \ Lt(inch)	10	100
5	0.8272	0.8272
10	1.6543	1.6543
15	2.4815	2.4815
20	3.3086	3.3086

TABLE IV

DAMPING CONSTANTS

I(mA) \ Lt(inch)	10	100
5	0.0679	0.0431
10	0.1358	0.0863
15	0.2307	0.1294
20	0.2716	0.1725

TABLE V

UNDAMPED NATURAL FREQUENCIES
(Hz)

I(mA) \ Lt(inch)	10	100
5	109.94	69.85
10	109.94	69.85
15	109.94	69.85
20	109.94	69.85

The characteristics of damping constants and undamped natural frequencies depending on the line length and the servovalve current are illustrated in Figures 16 and 17. As the line between the servovalve and the actuator increases in length, and as the valve current decreases, damping constants reduce thus making the transient responses more oscillatory. This occures because the damping ratio is a function of the flow-pressure coefficient and the oil volume as expressed below:

$$\delta_h = \frac{K_{ce}}{A_p}\sqrt{\frac{\beta_e M_t}{V_t}} \qquad (91)$$

The flow-pressure coefficient can be written in the following manner:

$$K_c = \frac{C_d \underline{W}_d x_i I \sqrt{(1/\rho)(P_s - P_L)}}{2(P_s - P_L)} \qquad (92)$$

Therefore, the damping ratio decreases because the oil volume increases as the line length increases. Decreased flow-pressure coefficient due to decreased valve current also results in the reduction of the damping ratio. The undamped natural frequency, expressed in the following manner:

$$W_h = \sqrt{\frac{4 \beta_e D_m^2}{V_t J_t}} \qquad (93)$$

decreases as the oil volume increases when length of line

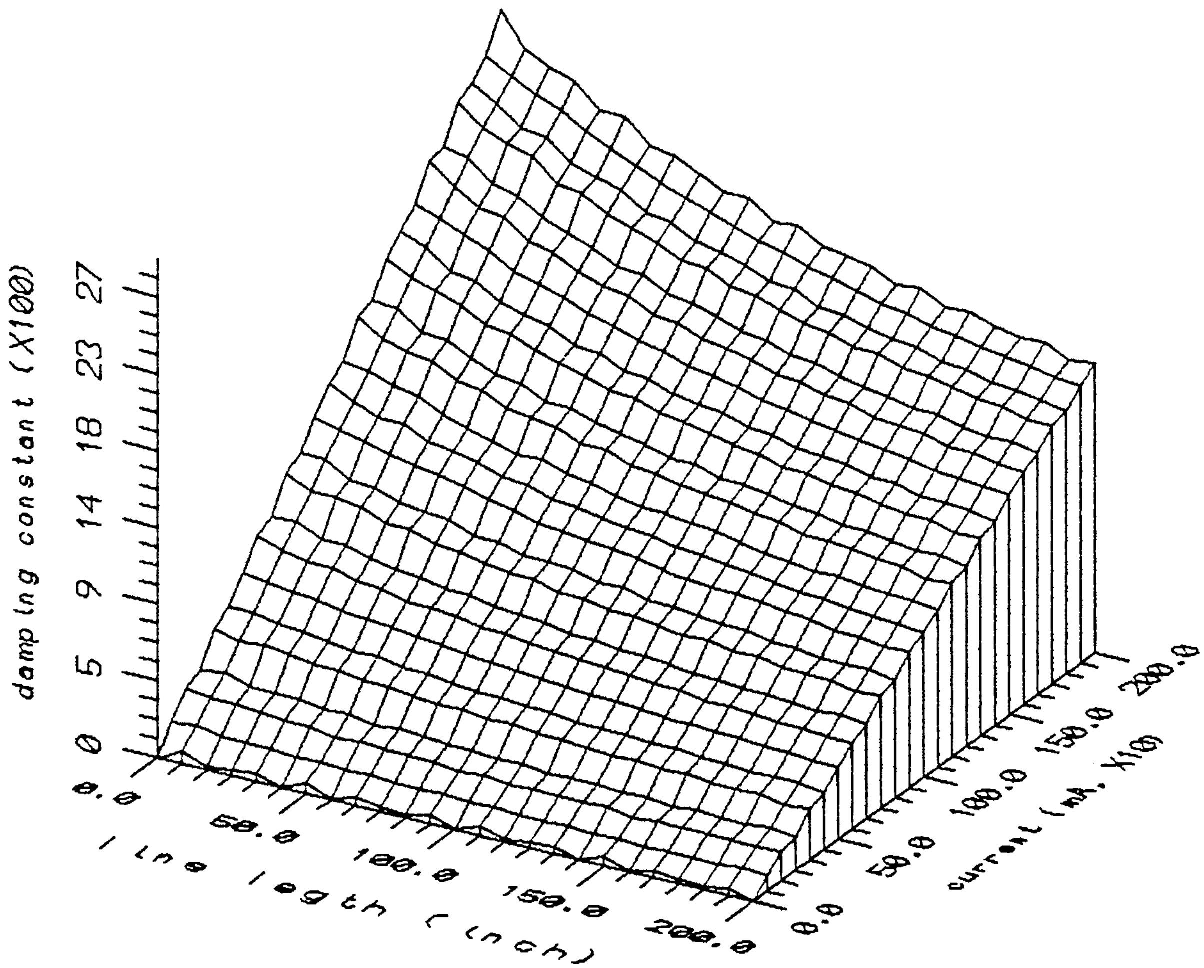

Figure 16. Variation of Damping Constants

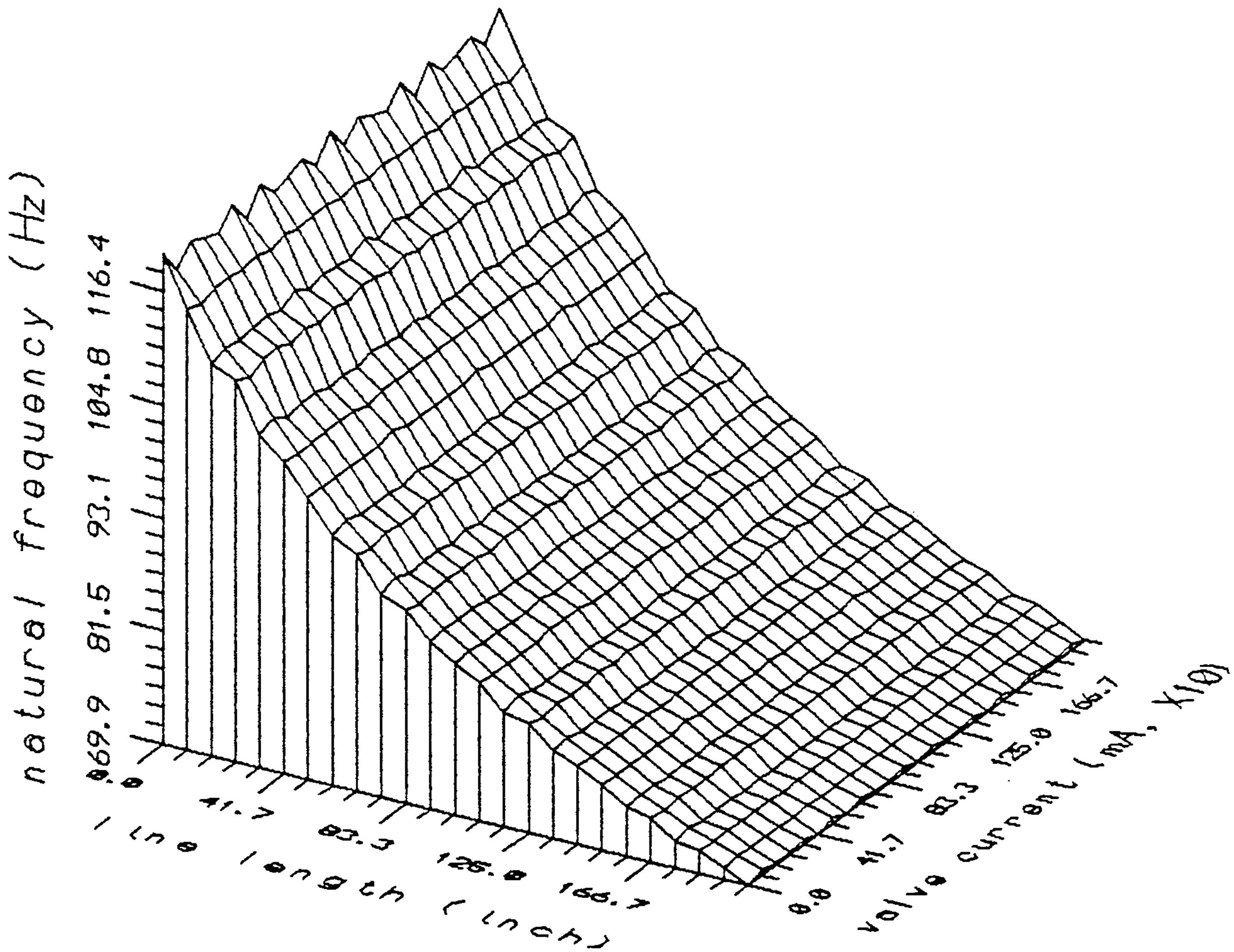

Figure 17. Variation of Natural Frequencies

increases. However, the undamped natural frequency is not affected by valve current.

Distributed Components Modelling

Distributed Parameters Line Modelling - Quasi Explicit Method

Equations (20) to (25) are interpreted as a computer program. See Appendix B. Specific boundary conditions used in the program are: the inlet flow rate is 50 times multiplied the inlet area of the tube and the outlet pressure is equal to the atmospheric pressure. The line is divided into 4 sections (ΔX = 50", Δt = 0.1 msec).

Figures 18 and 19 illustrate the dynamic responses of pressure and flow rate at the following line points: inlet, middle and outlet. Figure 20 reveal the dynamic responses of pressure at the middle point when the section length is changed; 100 inches, 50 inches, and 10 inches. Figure 20 showes that the principal natural frequency is almost same regardless of section number. Figures 21 and 22 reveal differences in the responses when the forward or backward difference method for differential equations (18) and (19) are alternatively changed. This data revealed that changes in the difference order yielded little affect. The absolute discrepancy is within 0.01. In other words, the solution of the partial differential equations of continuity and momentum using this quasi-explicit difference method, is considered to be the converged value with minimal tolerable

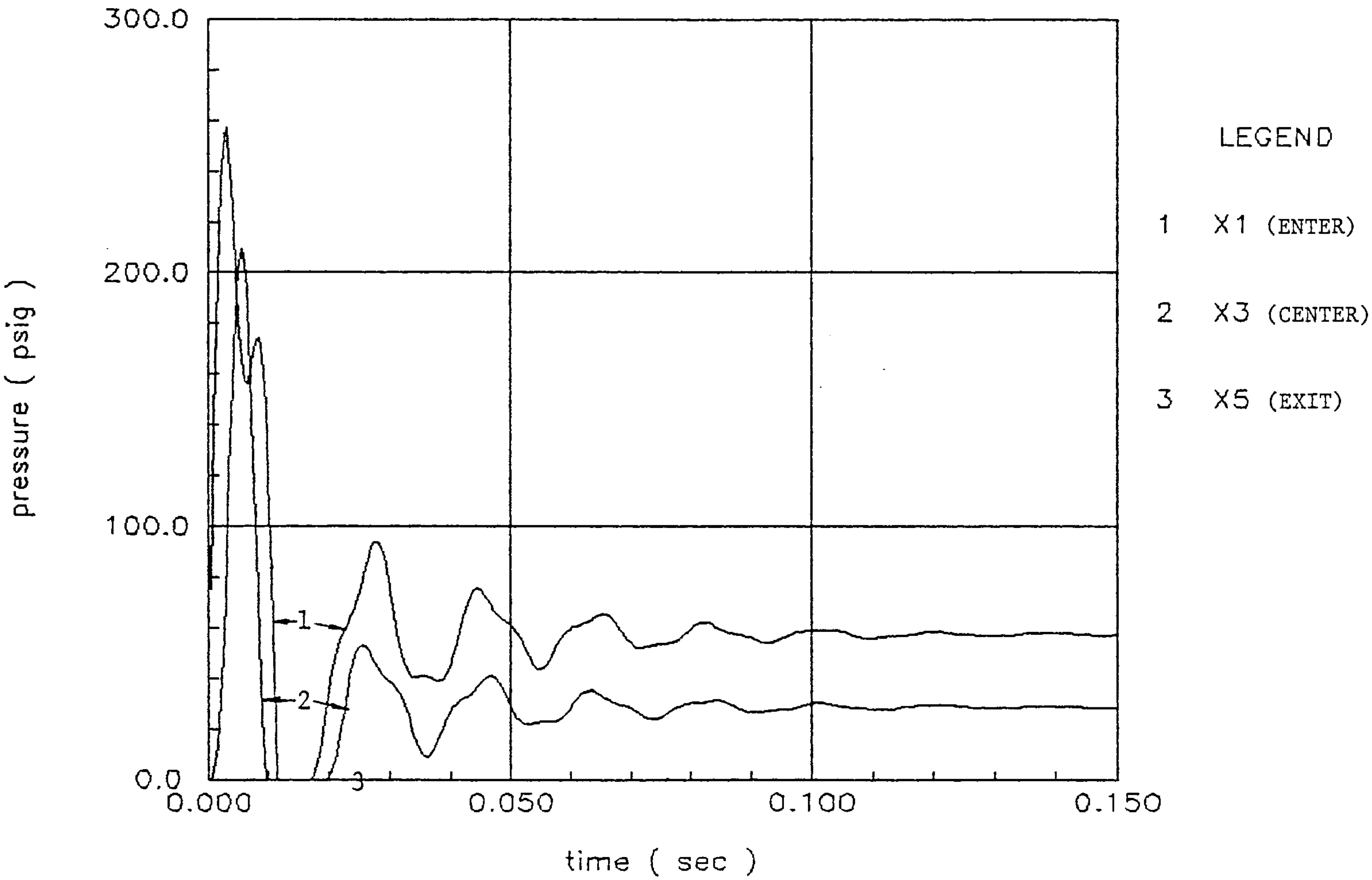

Figure 18. Dynamic Responses - Finite Difference Method

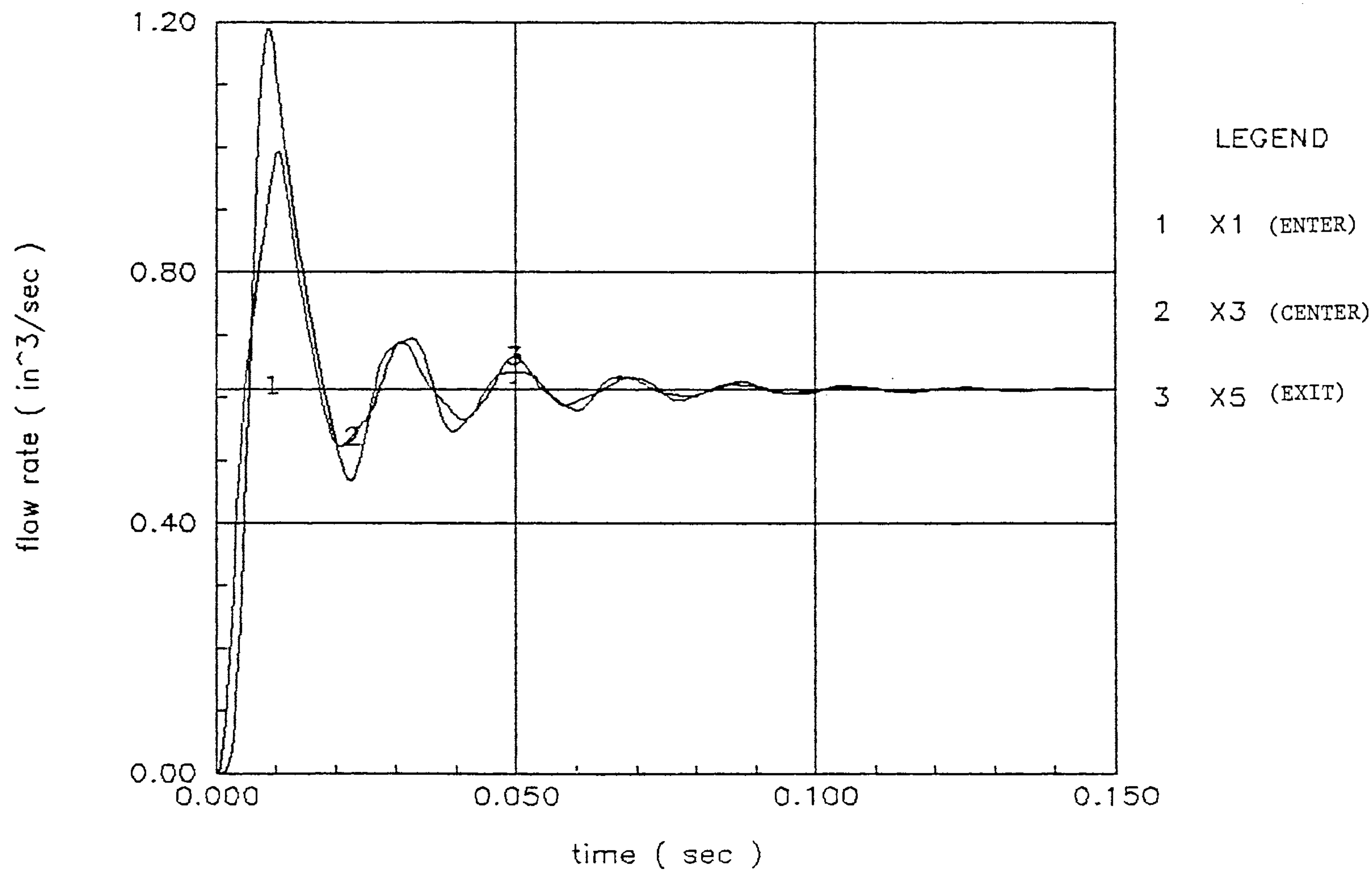

Figure 19. Dynamic Responses - Finite Difference Method

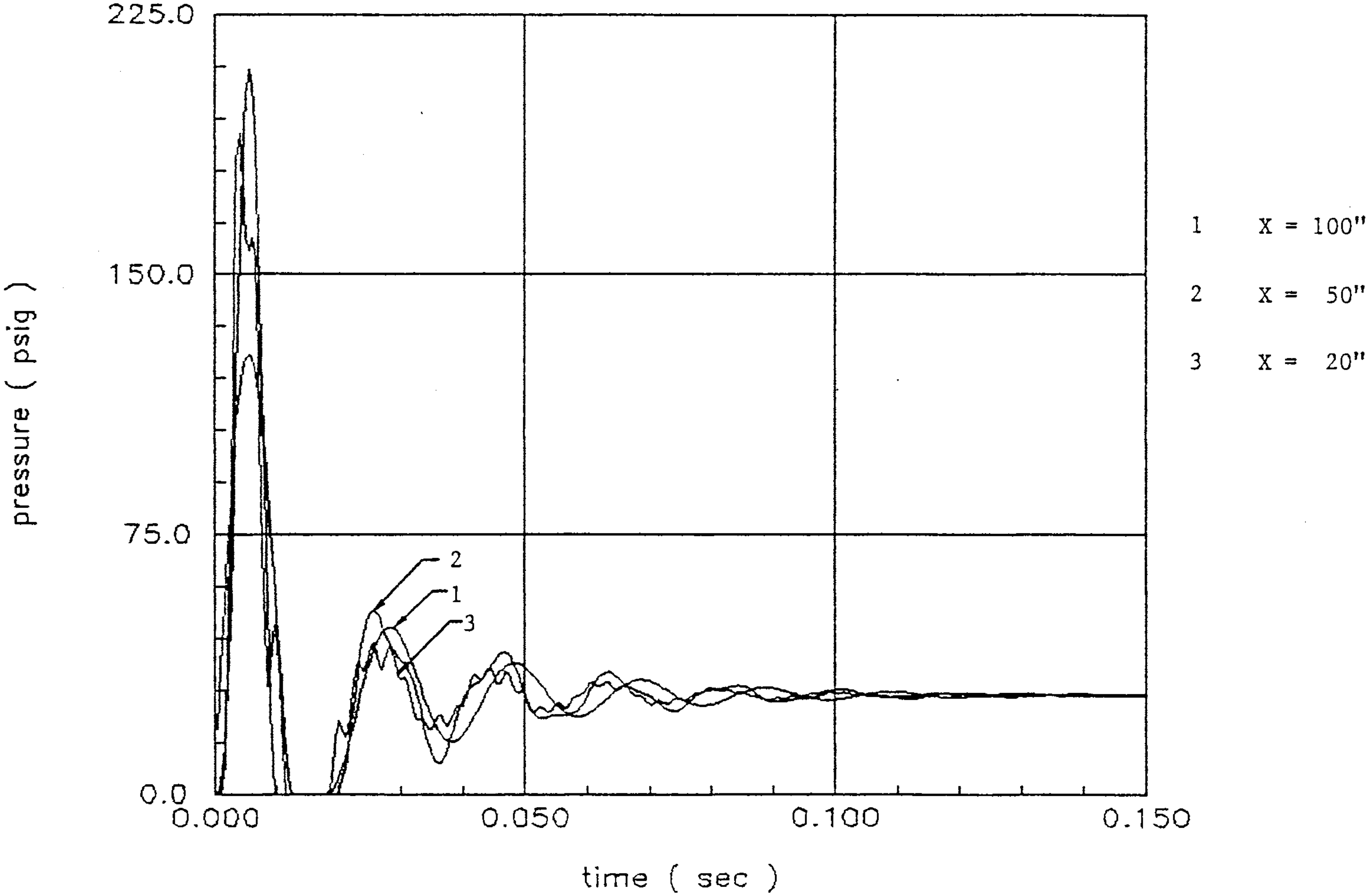

Figure 20. Pressure Transients Depending on the Section Length Using FDM

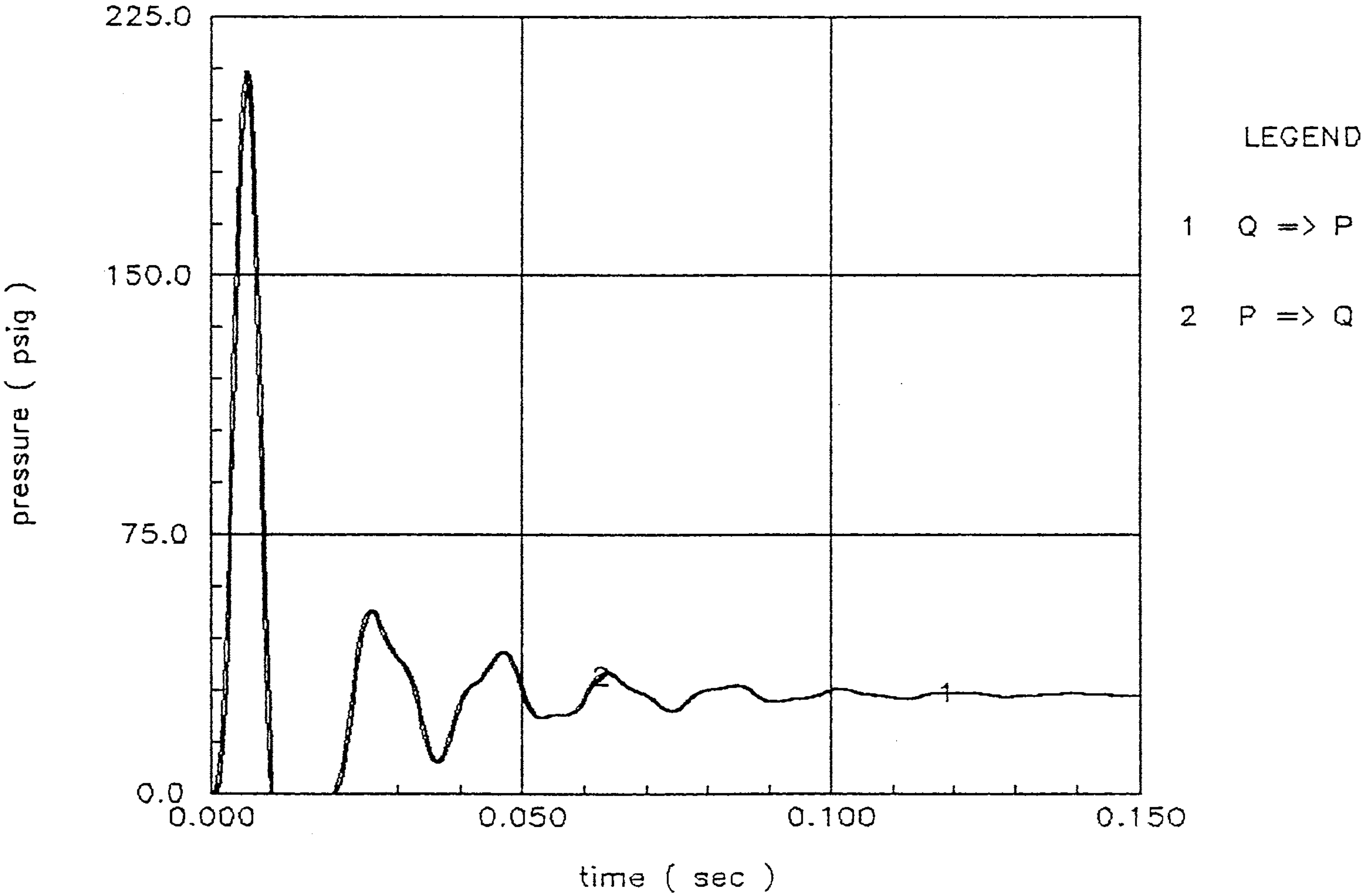

Figure 21. Dynamic Responses - FDM, Difference Order Change

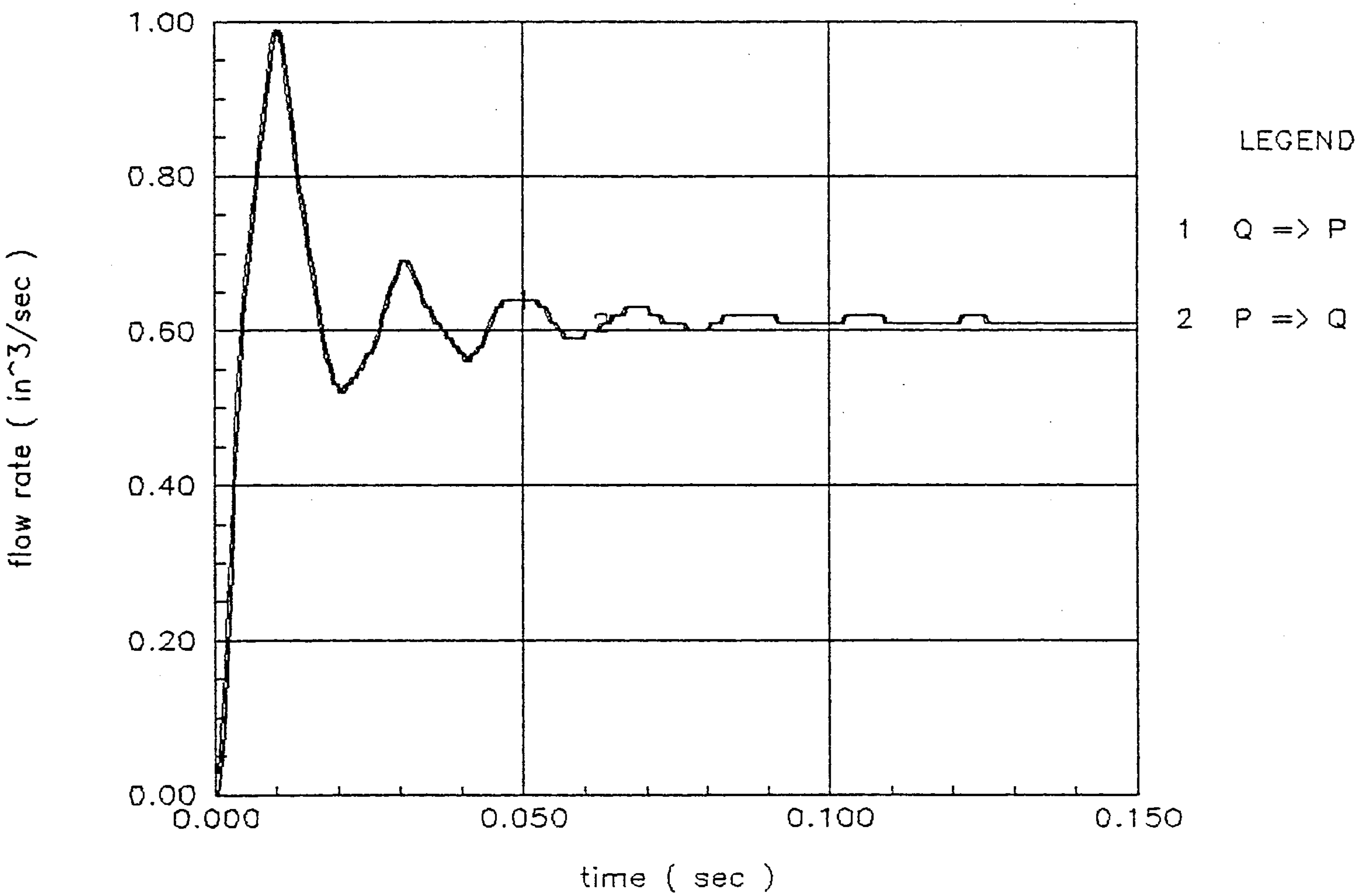

Figure 22. Dynamic Responses - FDM, Difference Order Change

error. Internal iteration for convergence which has been done in the implicit method is not necessary.

Distributed Parameters Line Modelling
- Explicit Method

Two statements for the positive and negative characteristics (C^+ and C^-), and boundary equations, (29) and (30), are solved. See Appendix C. The boundary conditions are the same as those previously cited in the quasi-explicit method. The responses in Figures 23 and 24 revealed the dynamic responses of pressure and flow rate within the lines. Compared to the responses determined using the quasi-explicit method (see Figures 25 and 26), the responses determined using the explicit method are more oscillatory - low damping constant - and show higher natural frequencies. Considering the fact that the characteristics of line dynamics are usually repetitive natural frequencies in higher frequency ranges and the low damping ratio due to its volume, the explicit method is more appropriate than the quasi-explicit method in obtaining the response closest to the real response. The propogation speed of sound in this example transmission line is:

$$C_o = \sqrt{\frac{\beta_e}{\rho}} = 43853 \text{ in/sec}$$

Therefore, the pressure wave frequency is:

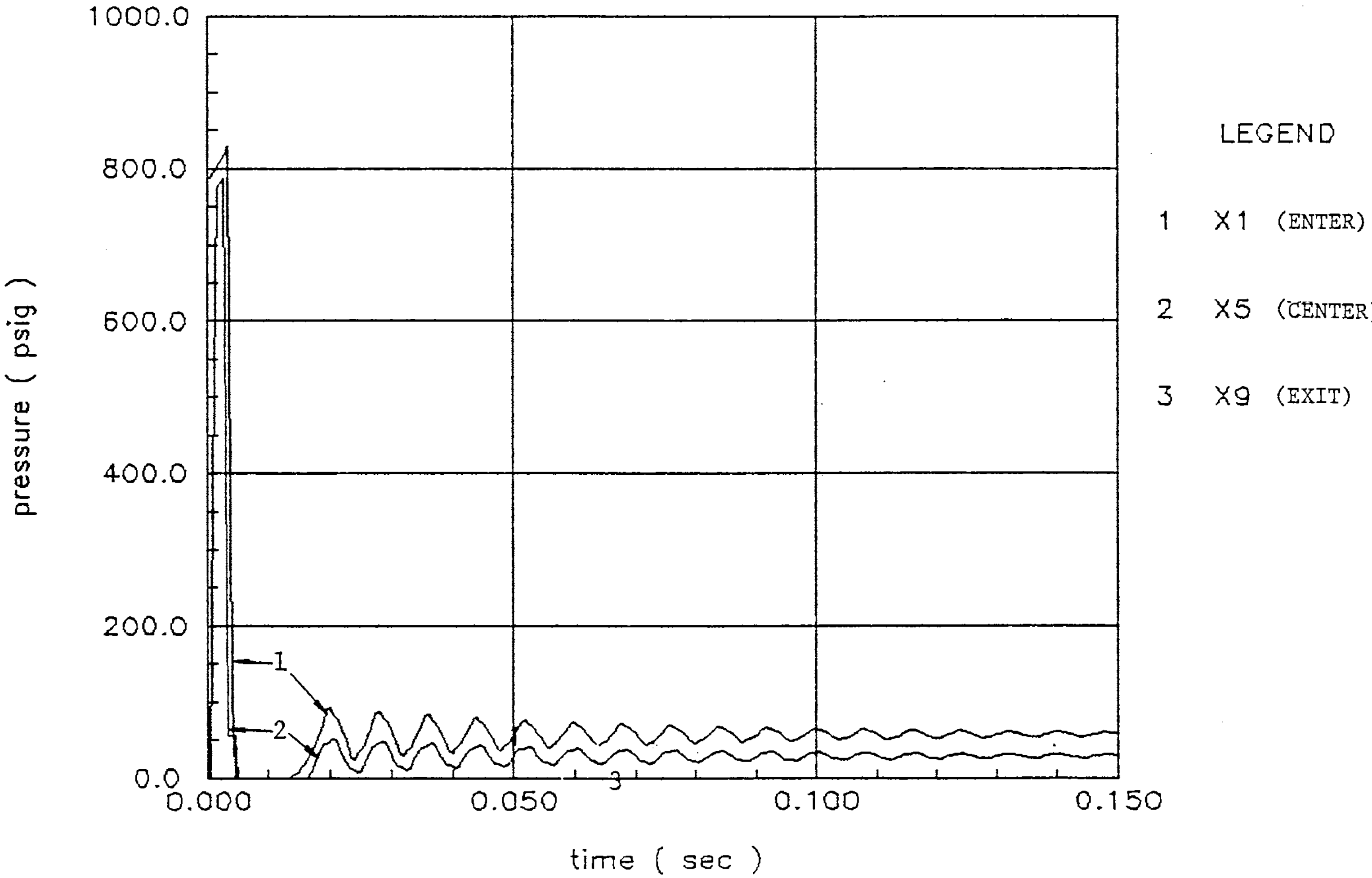

Figure 23. Dynamic Responses - Method of Characteristic

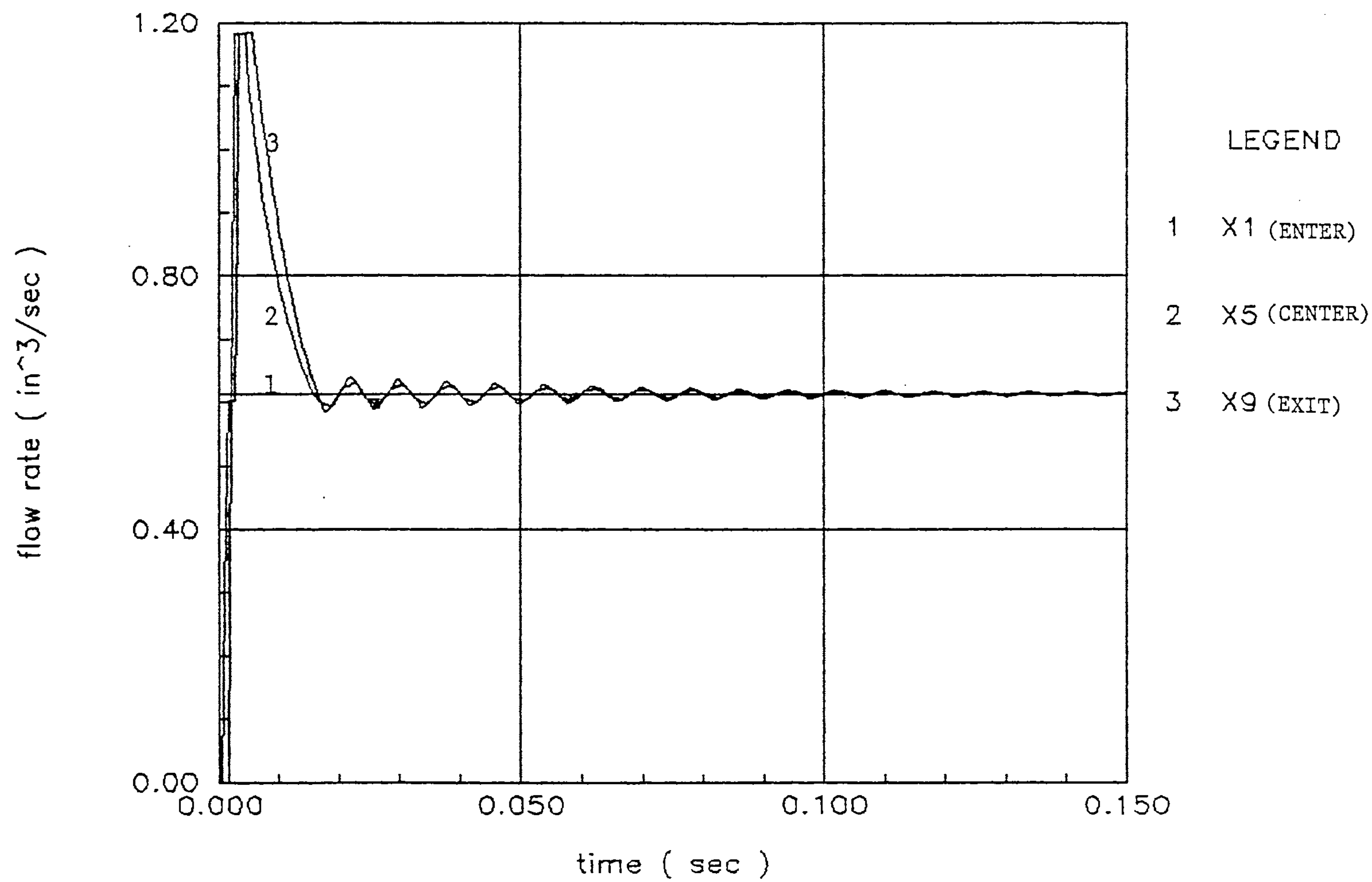

Figure 24. Dynamic Responses - Method of Characteristic

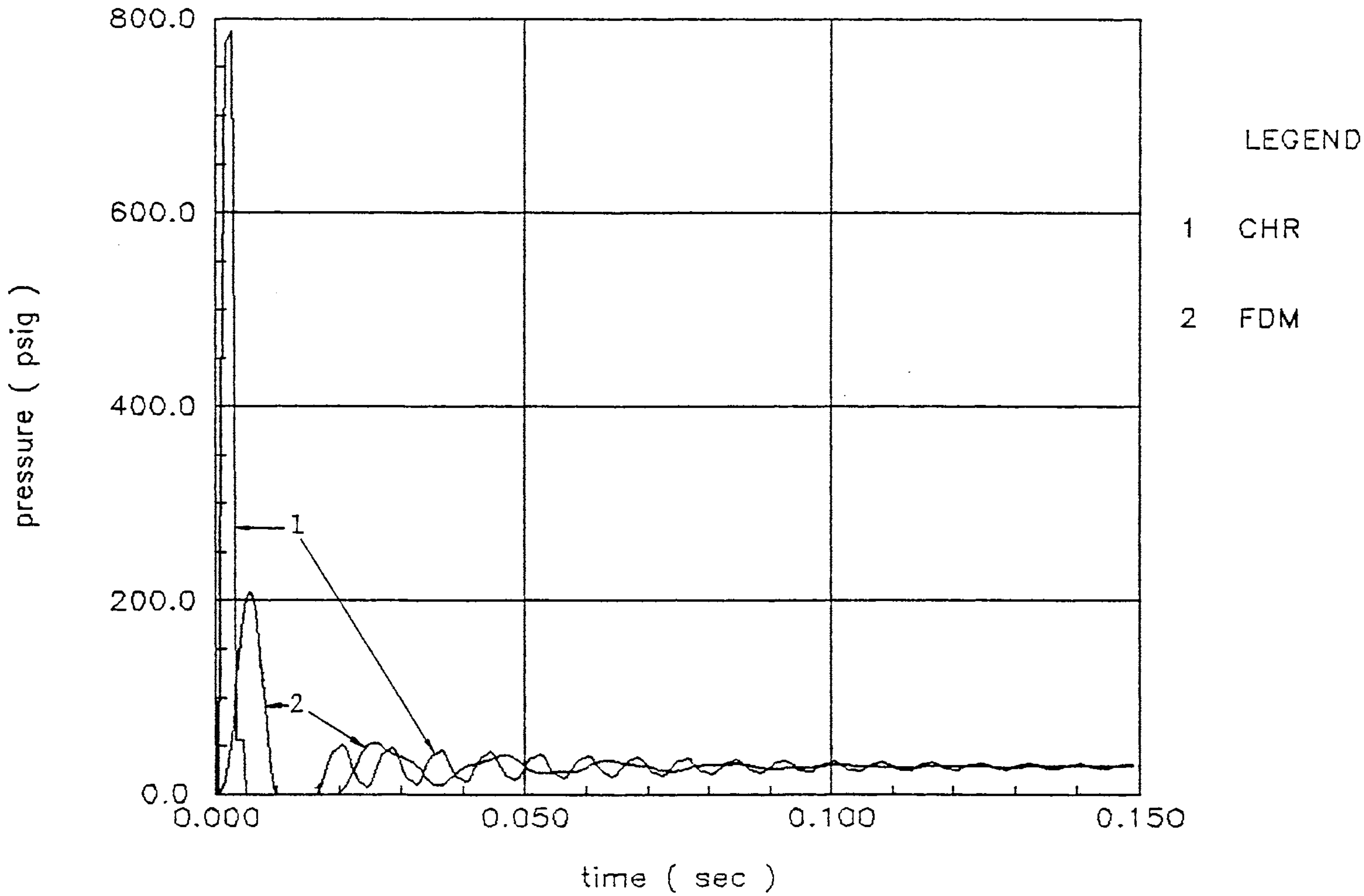

Figure 25. Dynamic Responses - Method of Characteristic and FDM

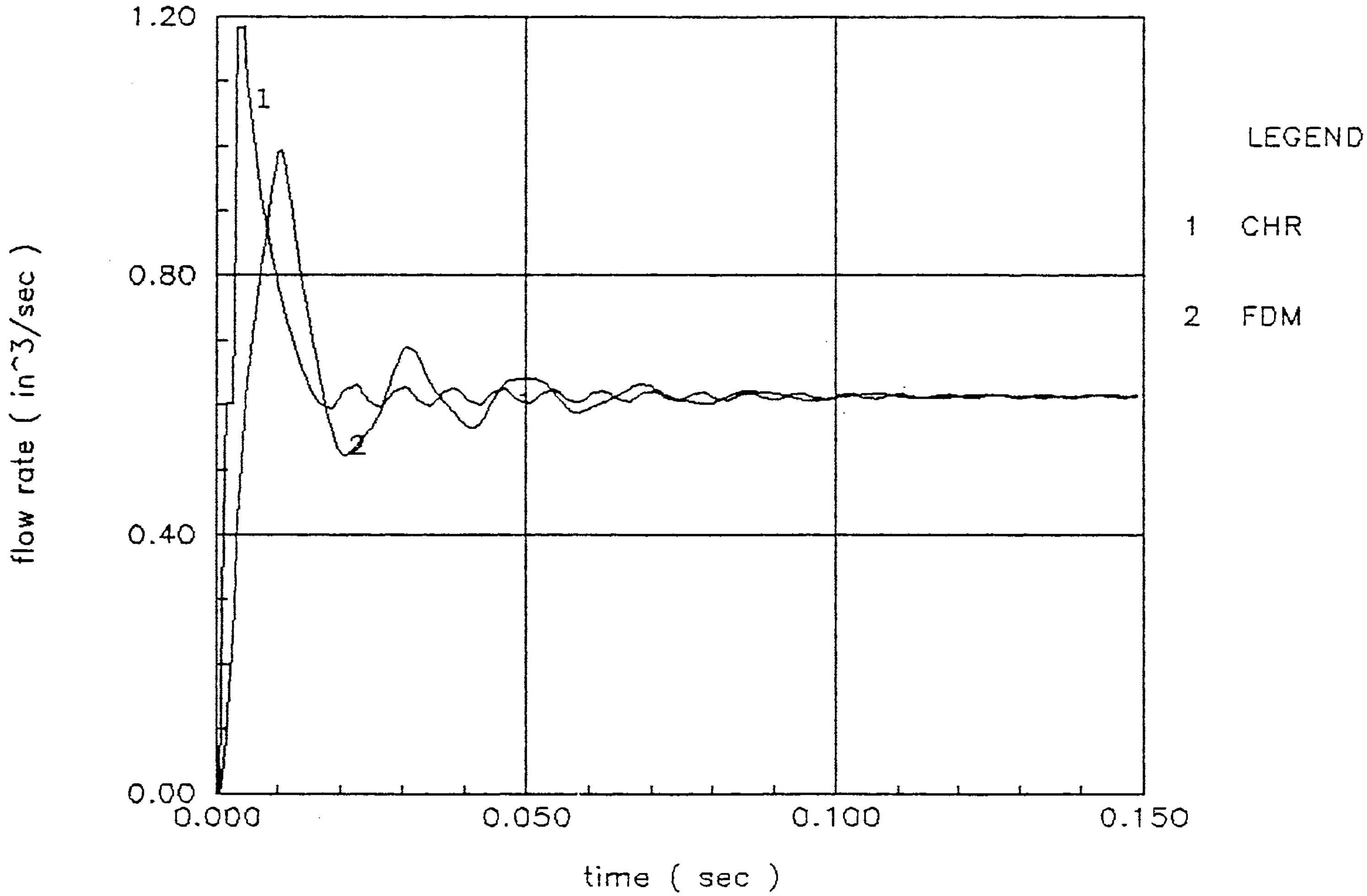

Figure 26. Dynamic Responses - Method of Charcateristic and FDM

$$f = \frac{C_o}{2L_t} = 110 \text{ Hz}$$

The frequency of pressure response obtained by the method of characteristics is about 120 Hz, and the frequency by FDM is about 50 Hz. This results reveal that the method of characteristics is more reliable than the FDM for the distributed parameter transmission line modelling
Both the quasi-explicit method and the implicit method can be quite unstable if the time step size is chosen incorrectly due to the inherent instability of the finite difference method. Therefore, the explicit method of characteristic is used to simulate the distributed parameters modelling of the system.

System Modelling Coupled with Lumped Mass Line

Before etablishment of the complete distributed components modelling using equations (52) through (69) coupled with distributed parameters line dynamics using method of characteristics, the modelling of the system without the line dynamics must be performed in order to see the effect of distributed components modelling. The system is shown in Figure 27. At this stage the line is considered as a fluid mass which has no differential order. Figures 28 and 29 represent the dynamic responses of the piston as the current to the servovalve alternates from 5 mA to 20 mA as the length of each transmission lines increases from 10

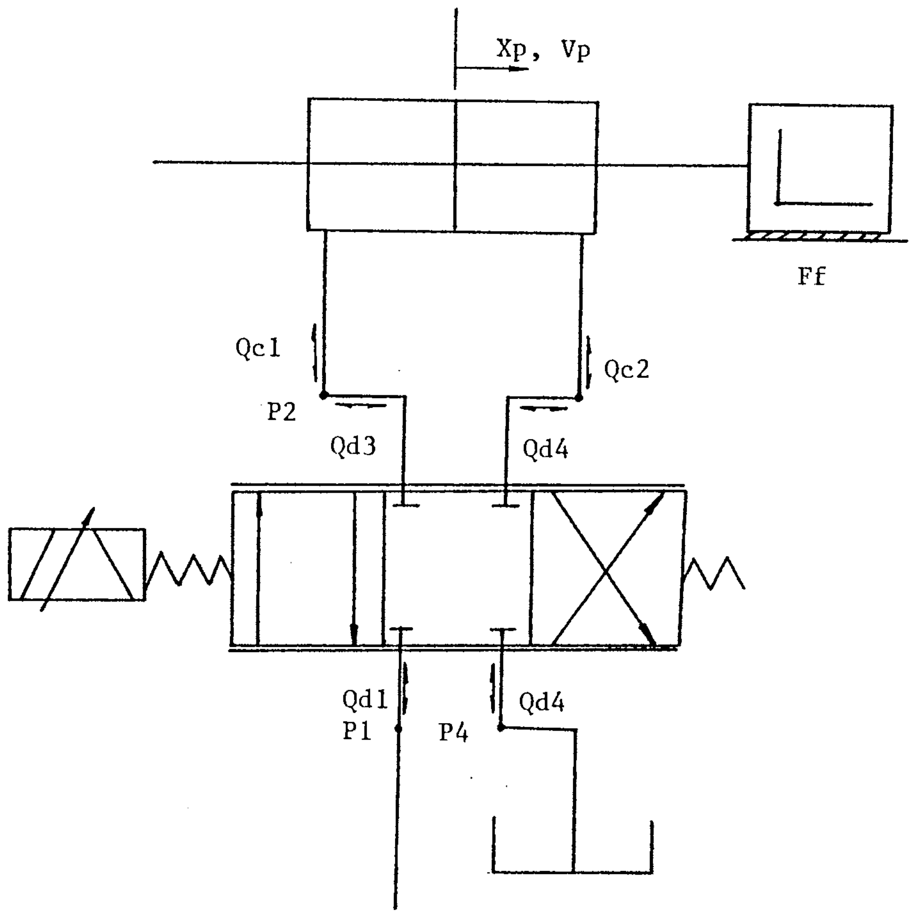

Figure 27. Distributed Components Modeling
(Lumped Mass Line)

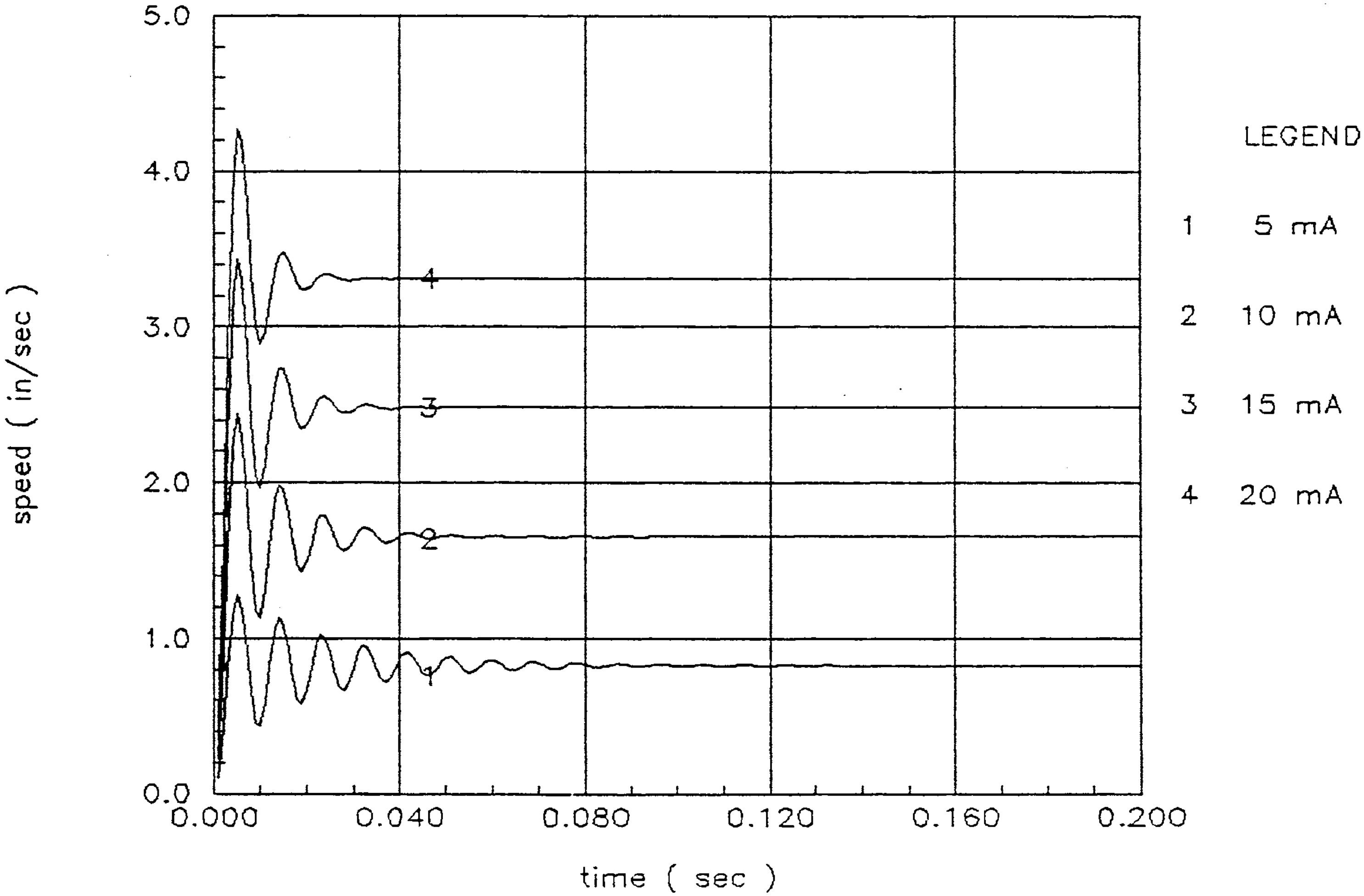

Figure 28. Dynamic Responses - Distributed Components with Lumped Line
(L = 10 inch)

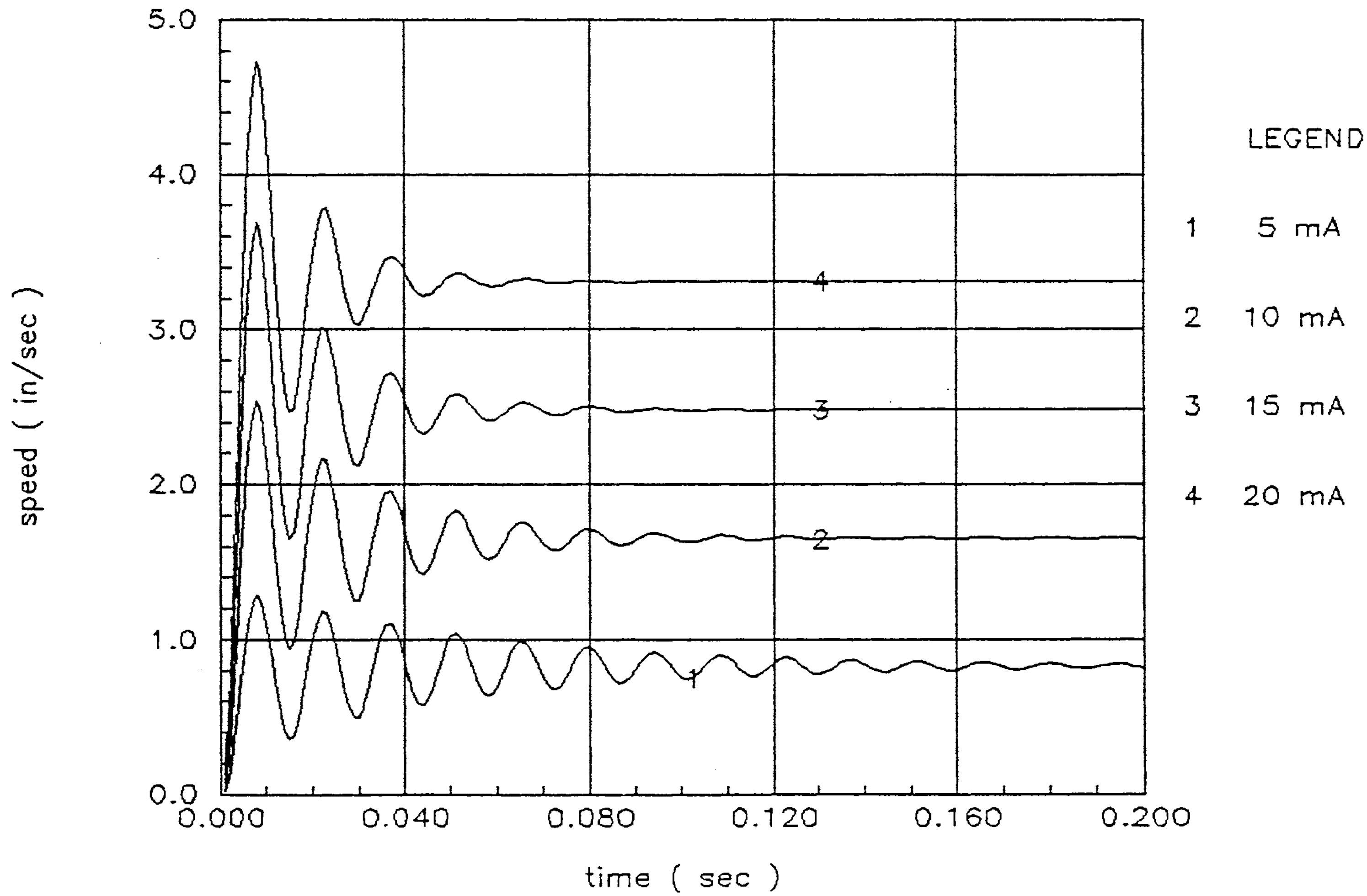

Figure 29. Dynamic Responses - Distributed Components with Lumped Line
(L = 100 inch)

inches to 100 inches. Compared to the responses determined using the linearized transfer function (see Figure 30), the total shape of the transient responses are similar. However, response delay increases as the line become longer. This occurs because more time is required to build up the pressure to overcome the friction force as the line length is longer (see Figure 31). The steady state pressures in the cylinder are : supply pressure = 254.5 psig, return pressure = 245.5 psig, load pressure = 9 psig. See Appendix D.

System Modelling Couple with Explicit Method of Line

The one-order two dimensional partial differential equations expressed by method of characteristics for the distributed parameters line modelling are coupled with the other hydraulic components expressed by ordinary differential equations. The following boundary equations of the line are used to determine the unknown flow rate values from the known pressure values at inlet and outlet port.

$$Q_1 = \frac{\left(C_oL - R\frac{dX}{4} \right) Q_2 + (P_1 - P_2)}{C_oL + R\frac{dX}{4}} \tag{94}$$

$$Q_{2n+1} = \frac{\left(C_oL - R\frac{dX}{4} \right) Q_{2n} - (P_{2n+1} - P_{2n})}{C_oL + R\frac{dX}{4}} \tag{95}$$

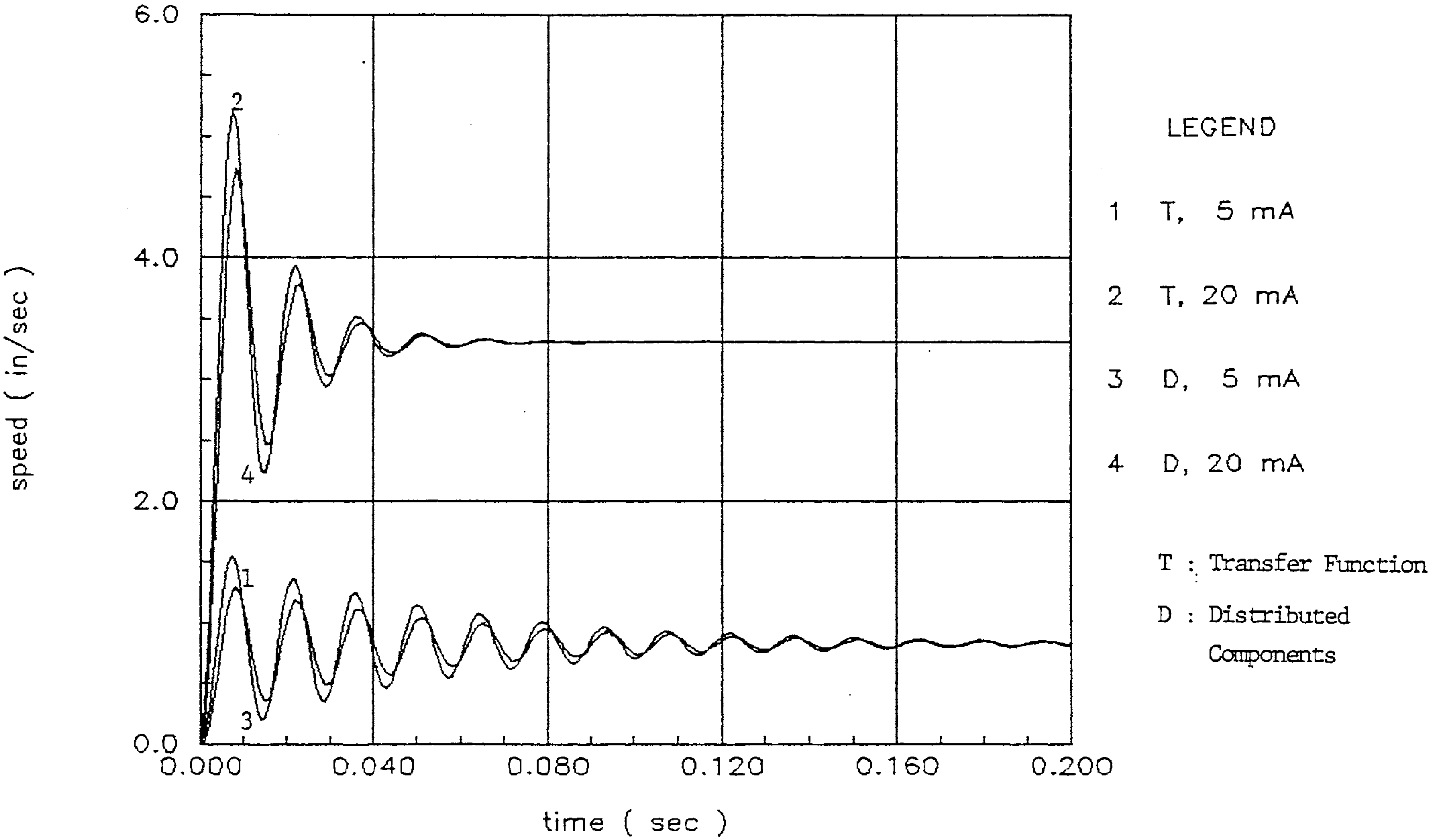

Figure 30. Dynamic Responses (L = 100 inch, I = 5, 20 mA)

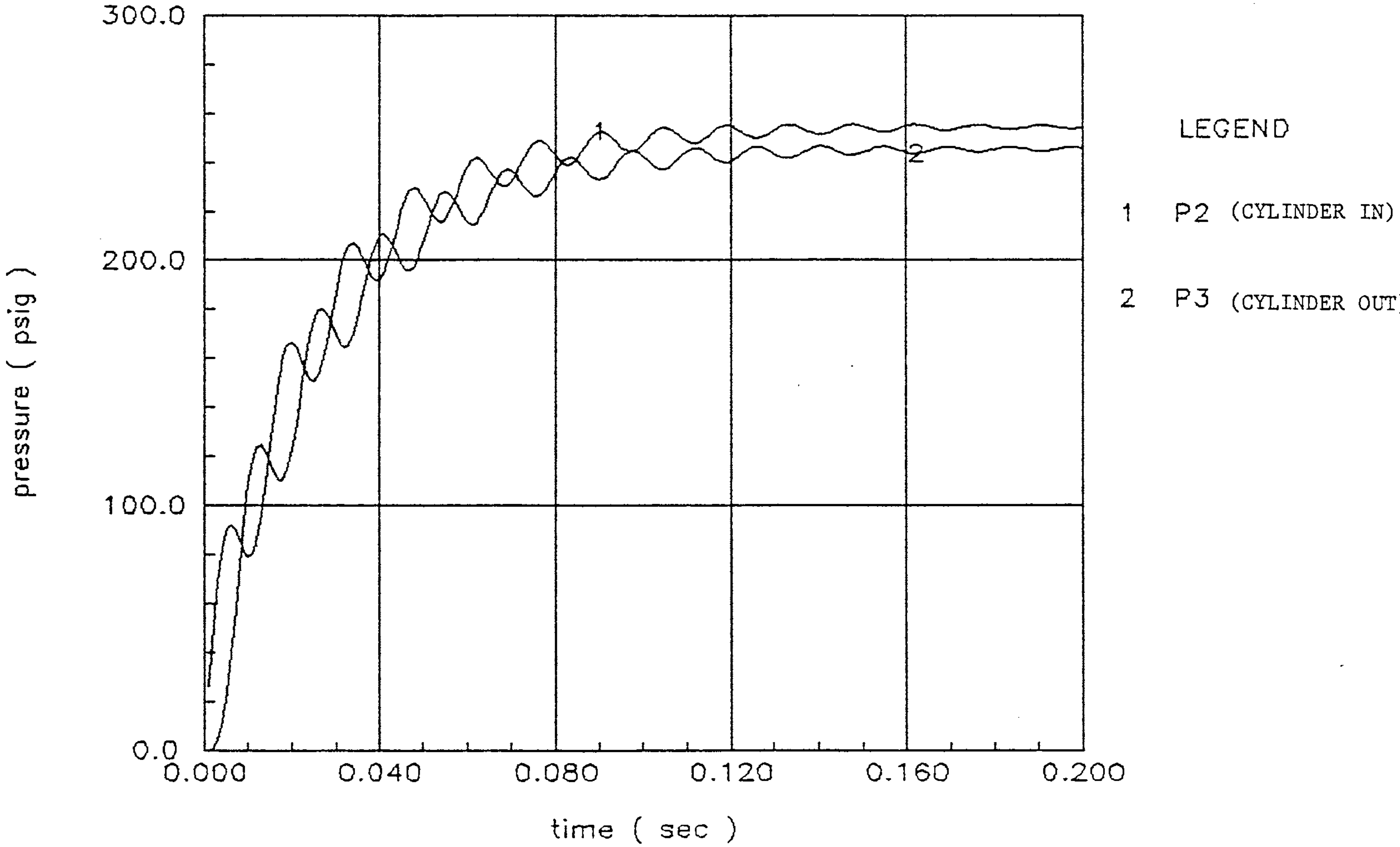

Figure 31. Dynamic Response - Distributed Components with Lumped Line
(L = 100 inch, I = 5 mA)

The schematic circuit for the servovalve controlled cylinder system coupled with the lines by distributed parameters modelling is shown in Figure 32. Figure 33 reveals the computing diagram. In reference to the computing diagram, the solution procedure is :

a. According to the initial conditions, the solution of the ordinary differential equations can be obtained in the time increment Δt, to obtain the state variables, P_2, P_3,P_4, P_5, V_p.

b. According to the initial conditions and the state variables known, the partial differential equation expressed by method of characteristics (see the statements about the positive and the negative characteristic, C^+ and C^-) are solved. The flow rates at inlet and outlet ports of the supply and return lines are determined using equations (94) and (95).

c. Using the flow rates of lines determined at step b, step a is repeated until a certain time limit is obtained.See Appendix E for the computer program.

Figure 34 through 36 reveal the responses of piston velocity when the valve current is 5 mA to 20 mA. The length of both the lines of supply and return lines are 10 inches, 100 inches, and 200 inches. As Watton recommended [15], the line is divided into four sections to expedite numerical analysis. Figures 37 and 38 reveal that the response changes are dependent upon the section length of line. As shown in Figures 39 and 40, the response delay is

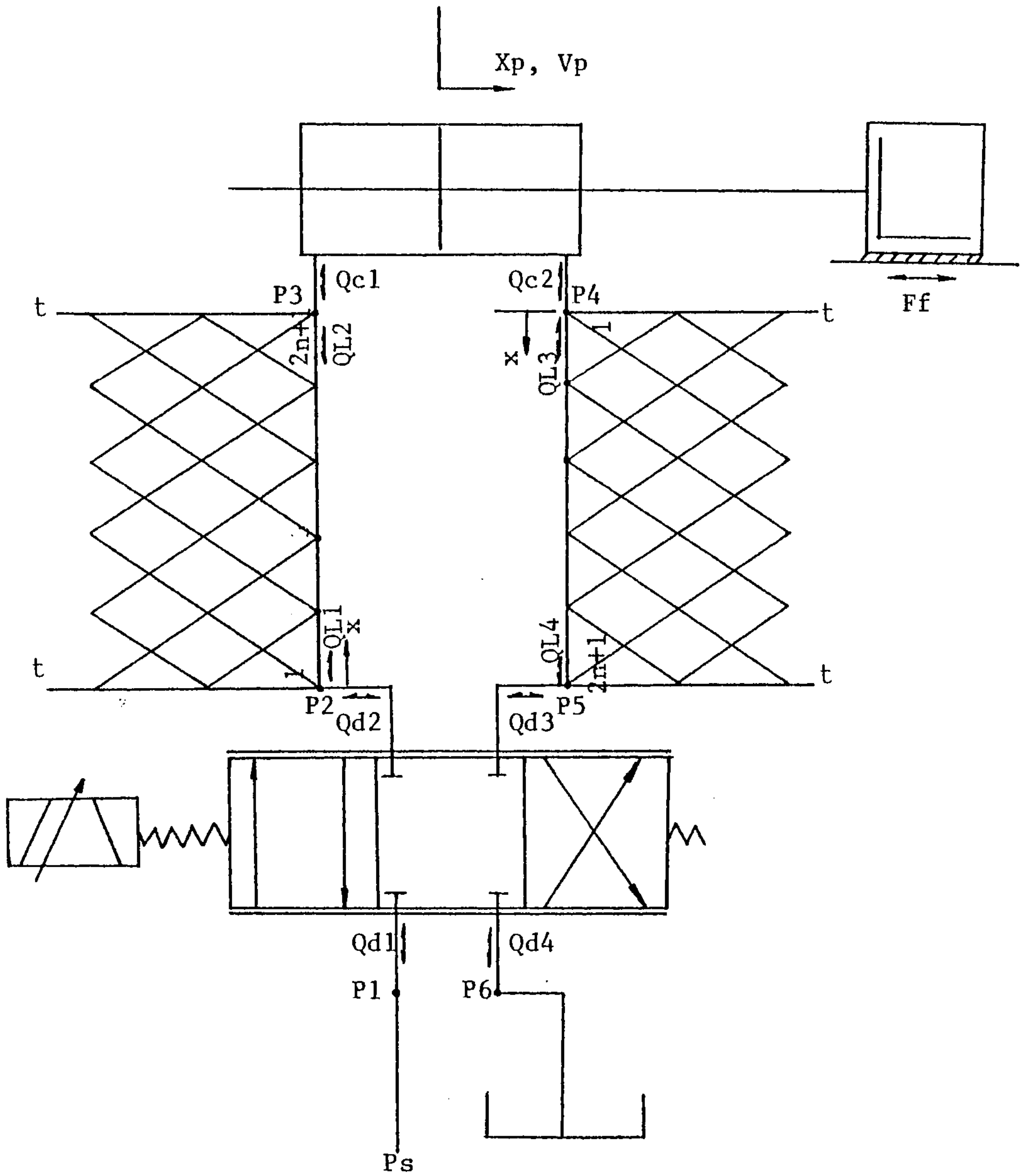

Figure 32. Distributed Components Modeling (Distributed Parameter Line)

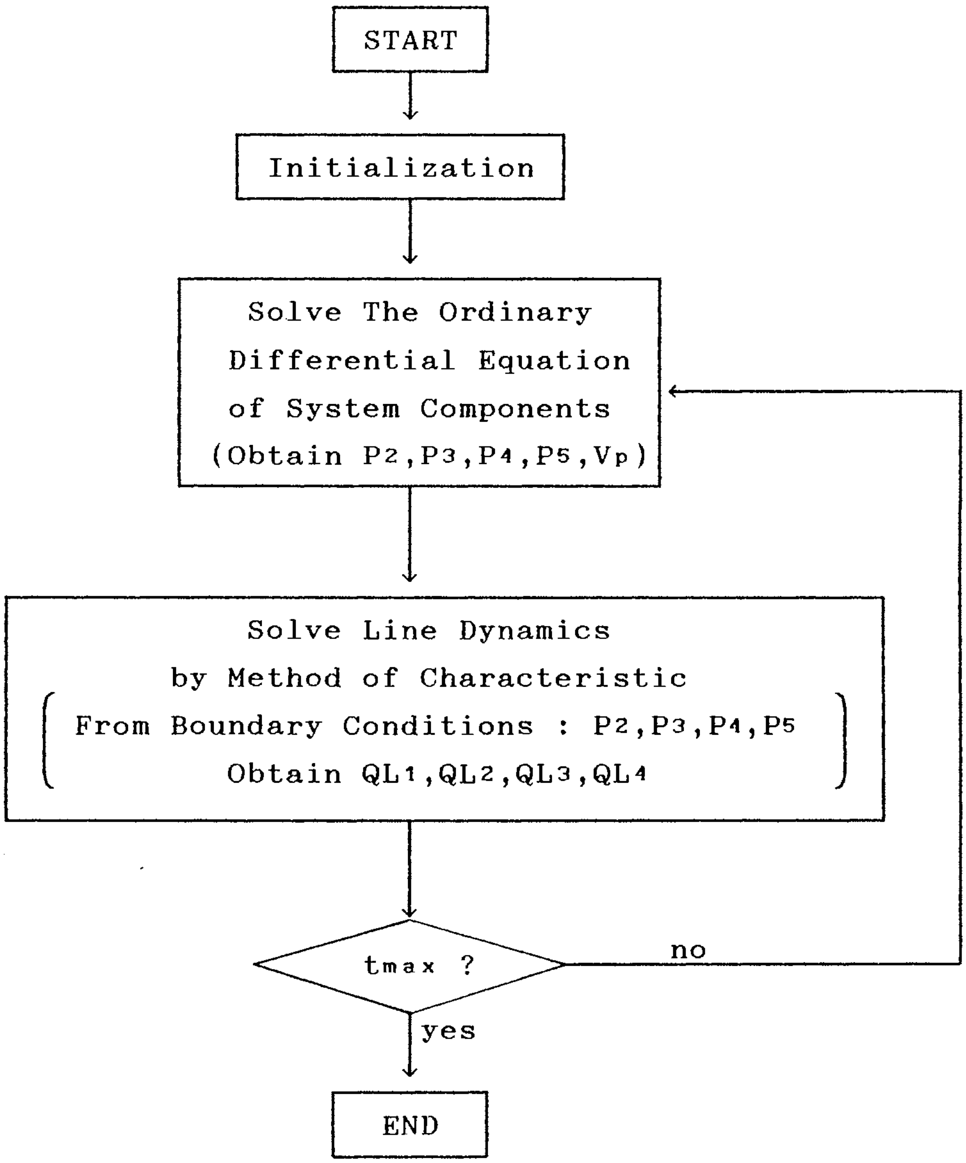

Figure 33. The Computing Diagram for Distributed Components System

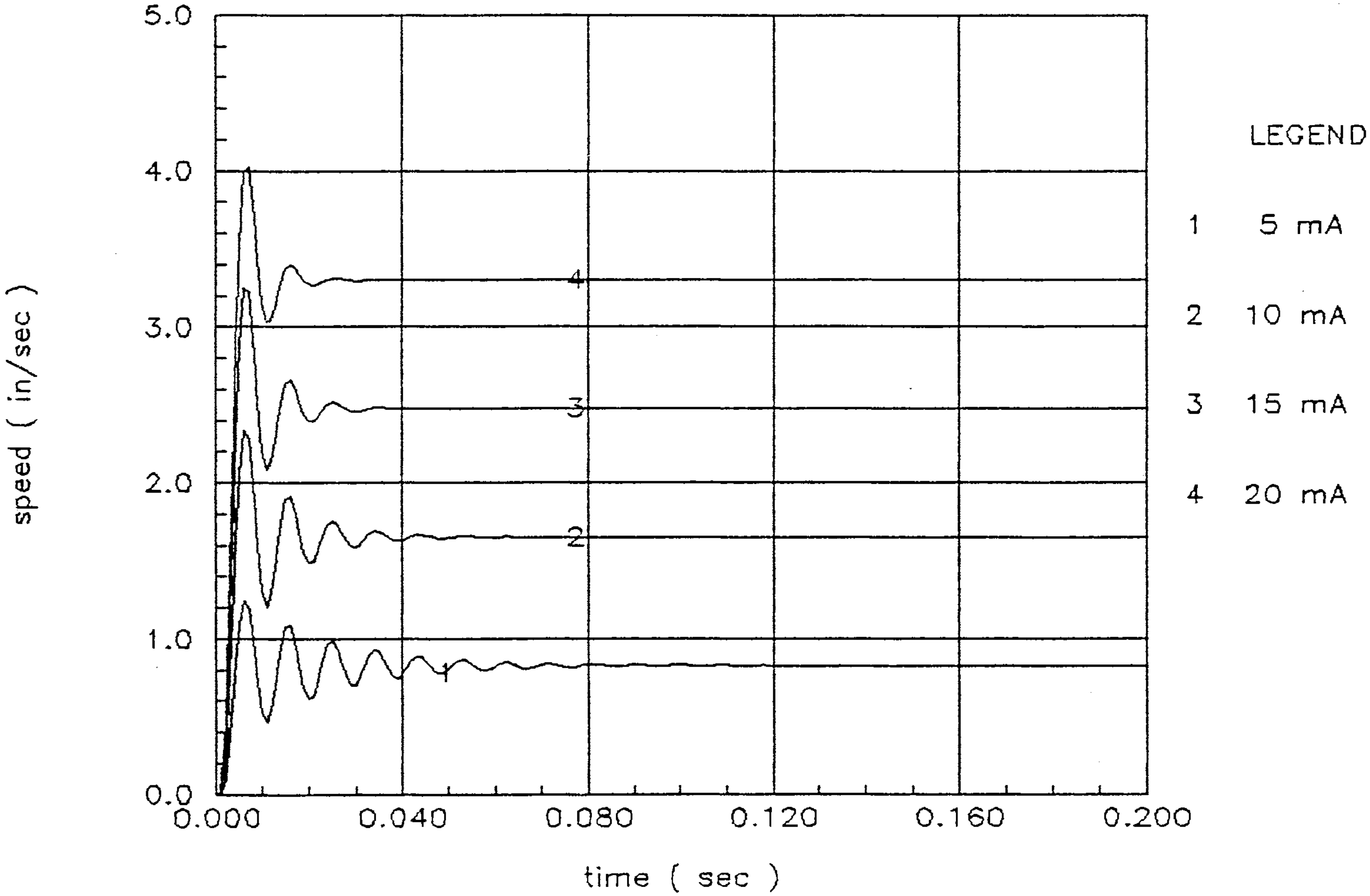

Figure 34. Dynamic Responses - Distributed Components with Distributed Parameter Line (L = 10 inch)

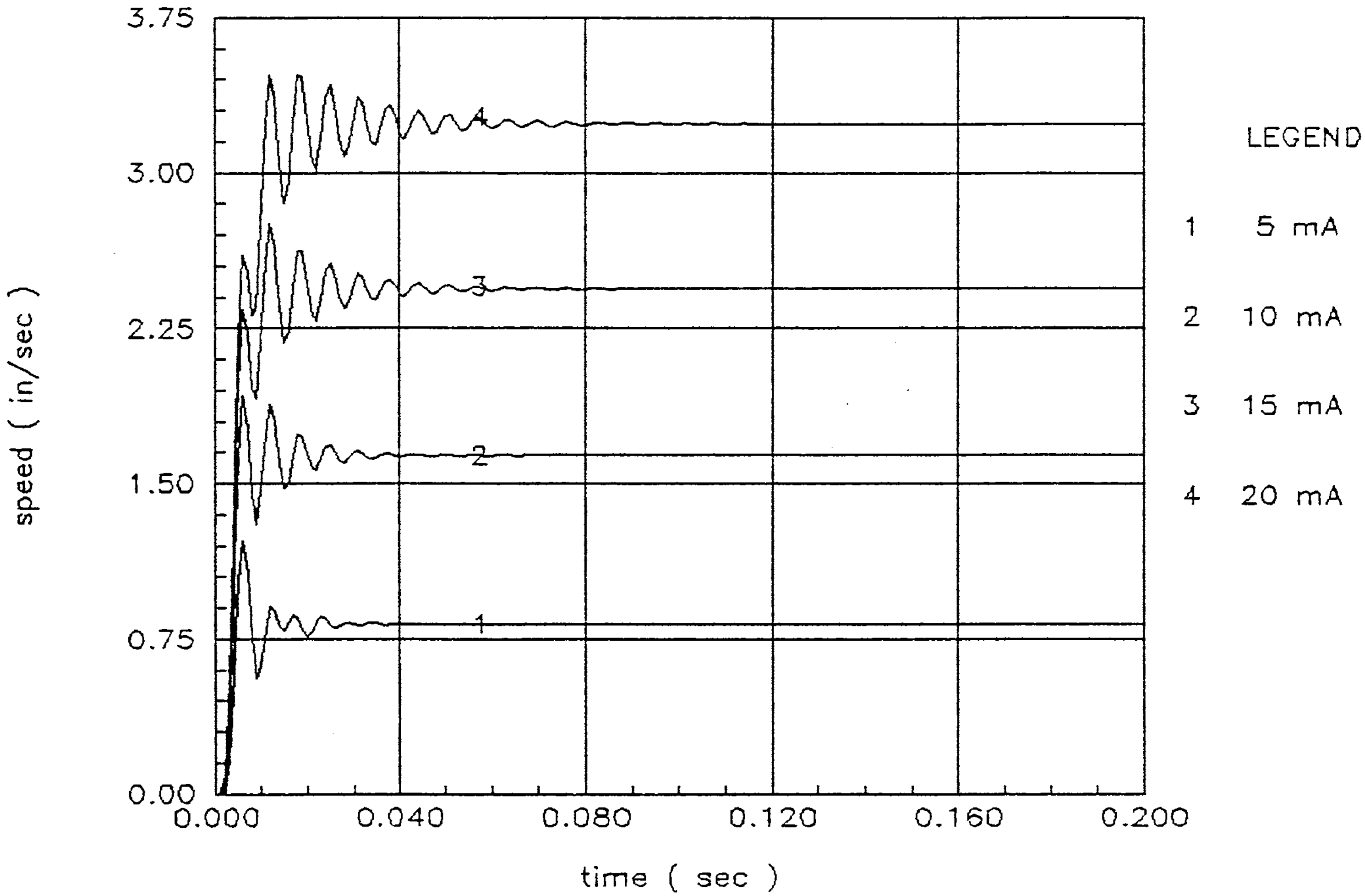

Figure 35. Dynamic Responses - Distributed Components with Distributed Parameter Line (L = 100 inch)

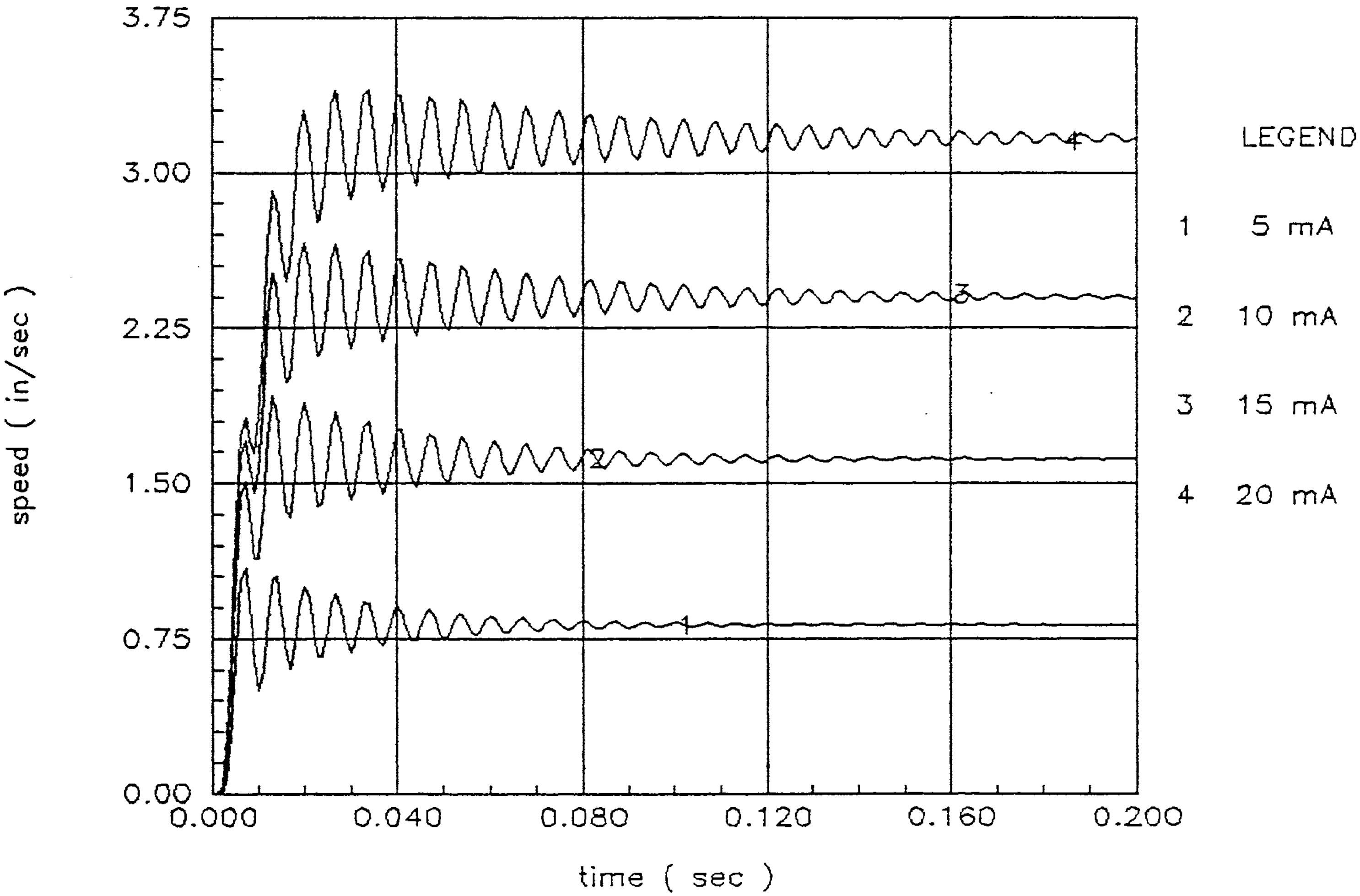

Figure 36. Dynamic Responses - Distributed Components with Distributed Parameter Line (L = 200 inch)

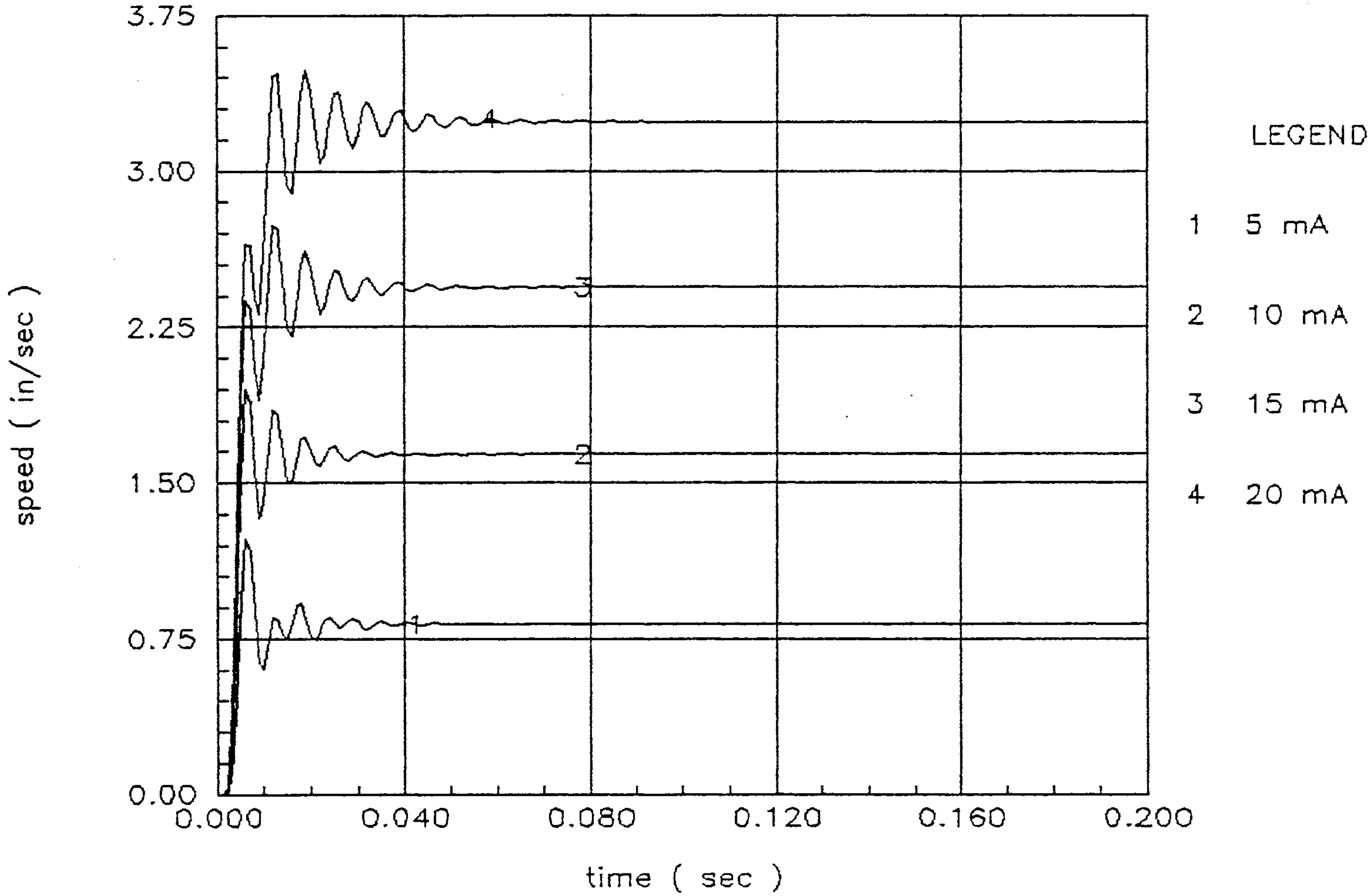

Figure 37. Dynamic Responses - Distributed Components with Distributed Parameter Line (L = 100 inch)

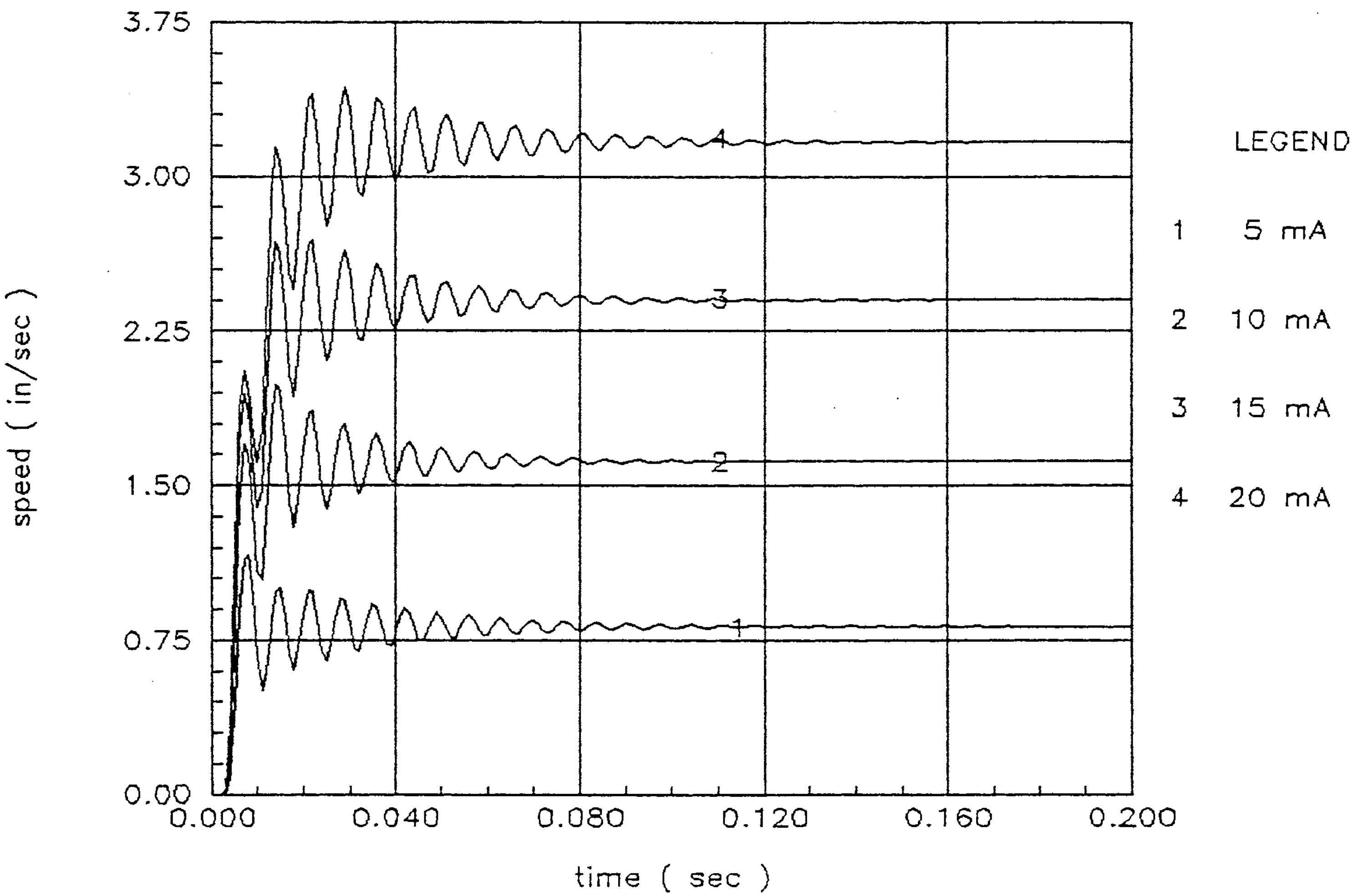

Figure 38. Dynamic Responses - Distributed Components with Distributed Parameter Line (L = 200 inch)

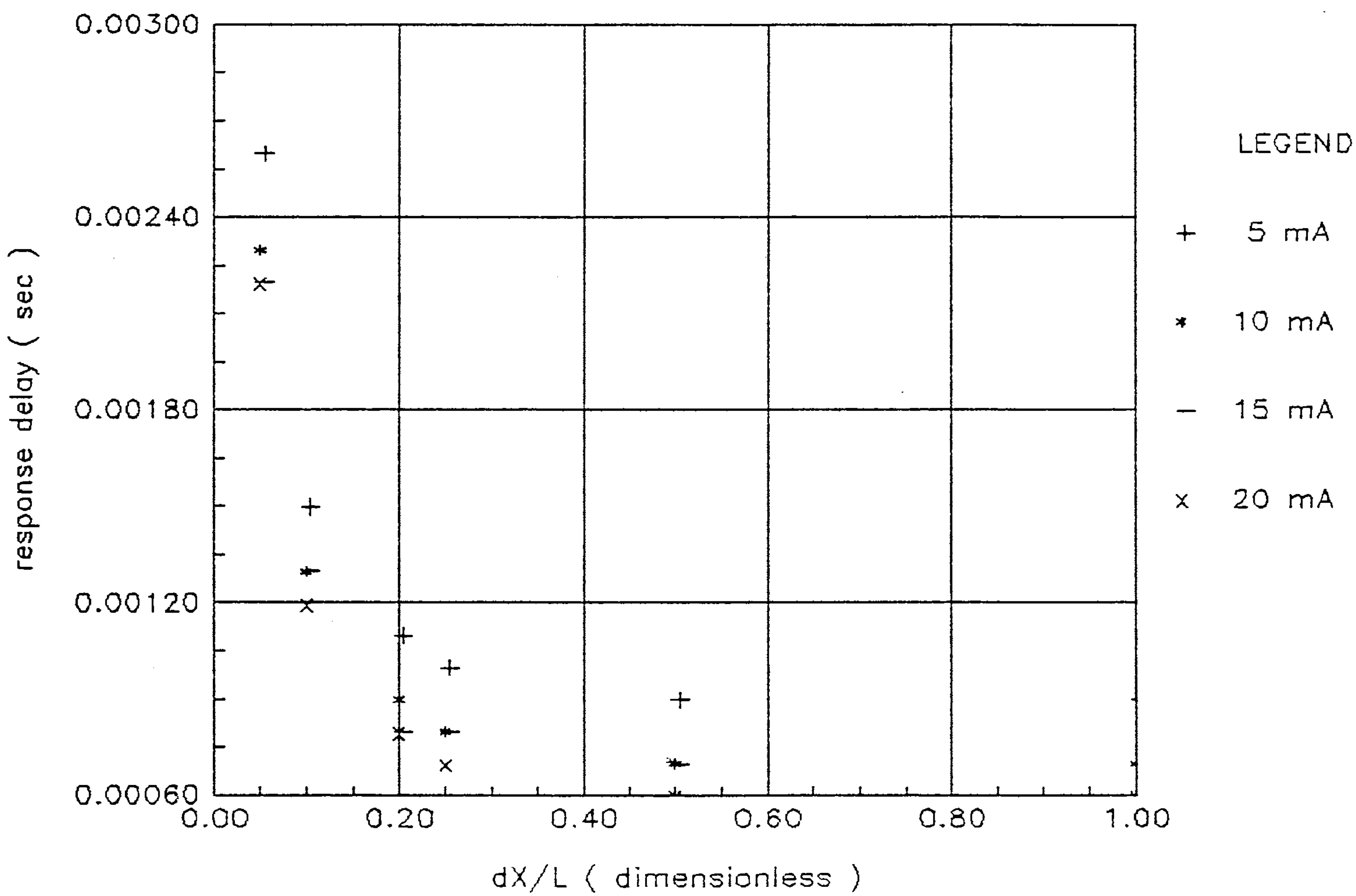

Figure 39. Response Delay (L = 10 inch)

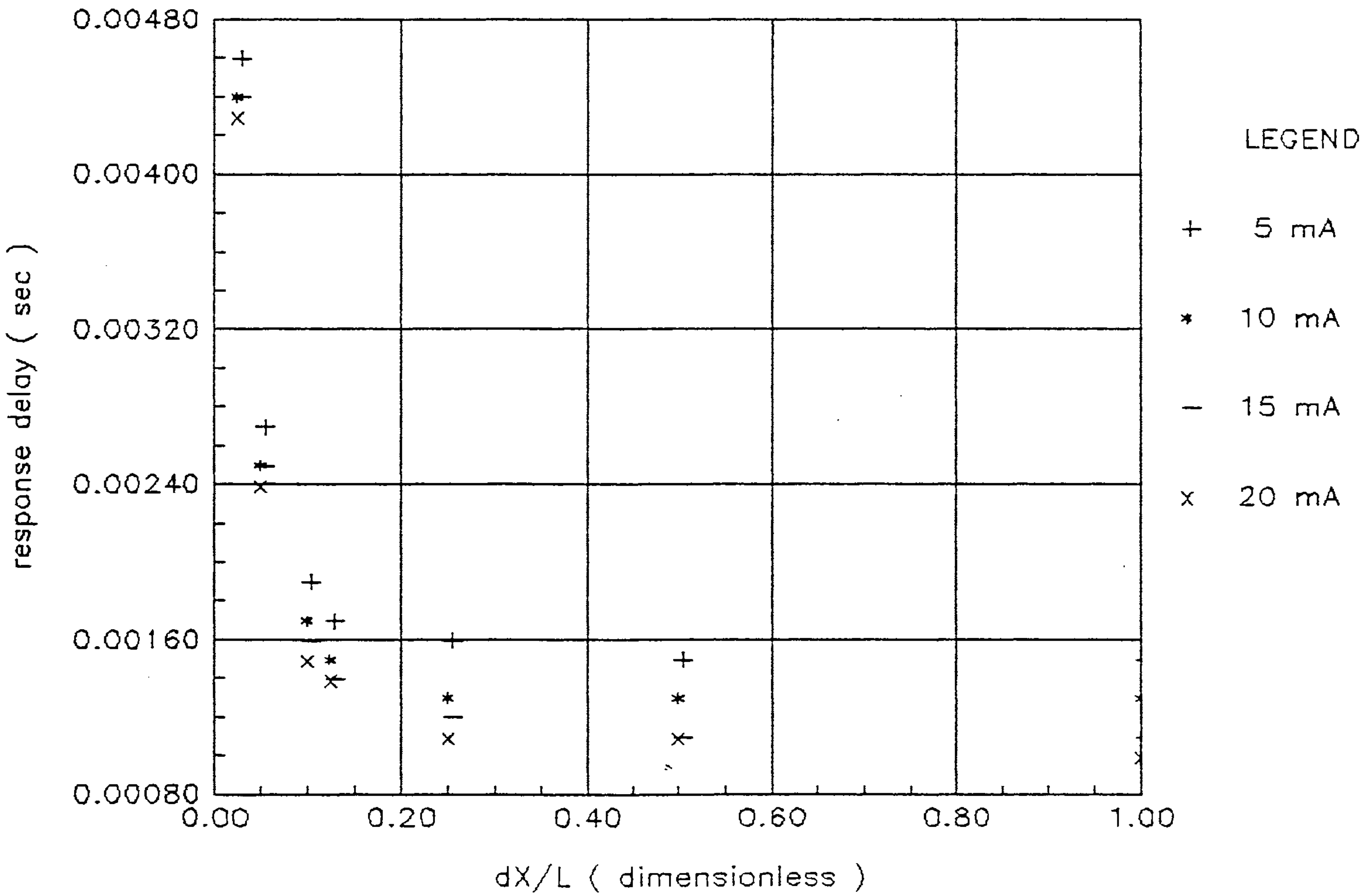

Figure 40. Response Dealy (L = 100 inch)

also dependent upon the section length of line for numerical analysis. The method of characteristic reveals that shorter section length results in long delay time. Following Watton's recommendation, the point of division of four sections ($\Delta X/L_t = 0.25$) is closely located at the point where the shorter length results in sudden increase of response dealy. The exhorbitant increase in time delay seems to be manifestation of inherent in computer simulation time. In the remainder of this study, Watton's recommendation of four sections has been utilized.

Compared to the responses determined using the linearized transfer function (see Figures 41 and 42), there is little difference in transient responses when a short line of 10 inches is used. However, when the line length increases the responses are quite different. In the system with 100 inch lines length (see Figure 35), the response at 20 mA of valve current is highly oscillatory. This results from higher natural frequencies of long line. The effect of higher natural frequencies is more obvious in the system responses of 200 inch lines as (see Figure 36). The response at 20 mA of valve current reveals the system is longer second order but higher than second order. Since the propogation speed of sound in the oil, C_o is

$$C_o = \sqrt{\frac{\beta_e}{\rho}} = 43{,}853 \text{ in/sec} \qquad (96)$$

and the pressure wave frequency in the line of 200 inches is

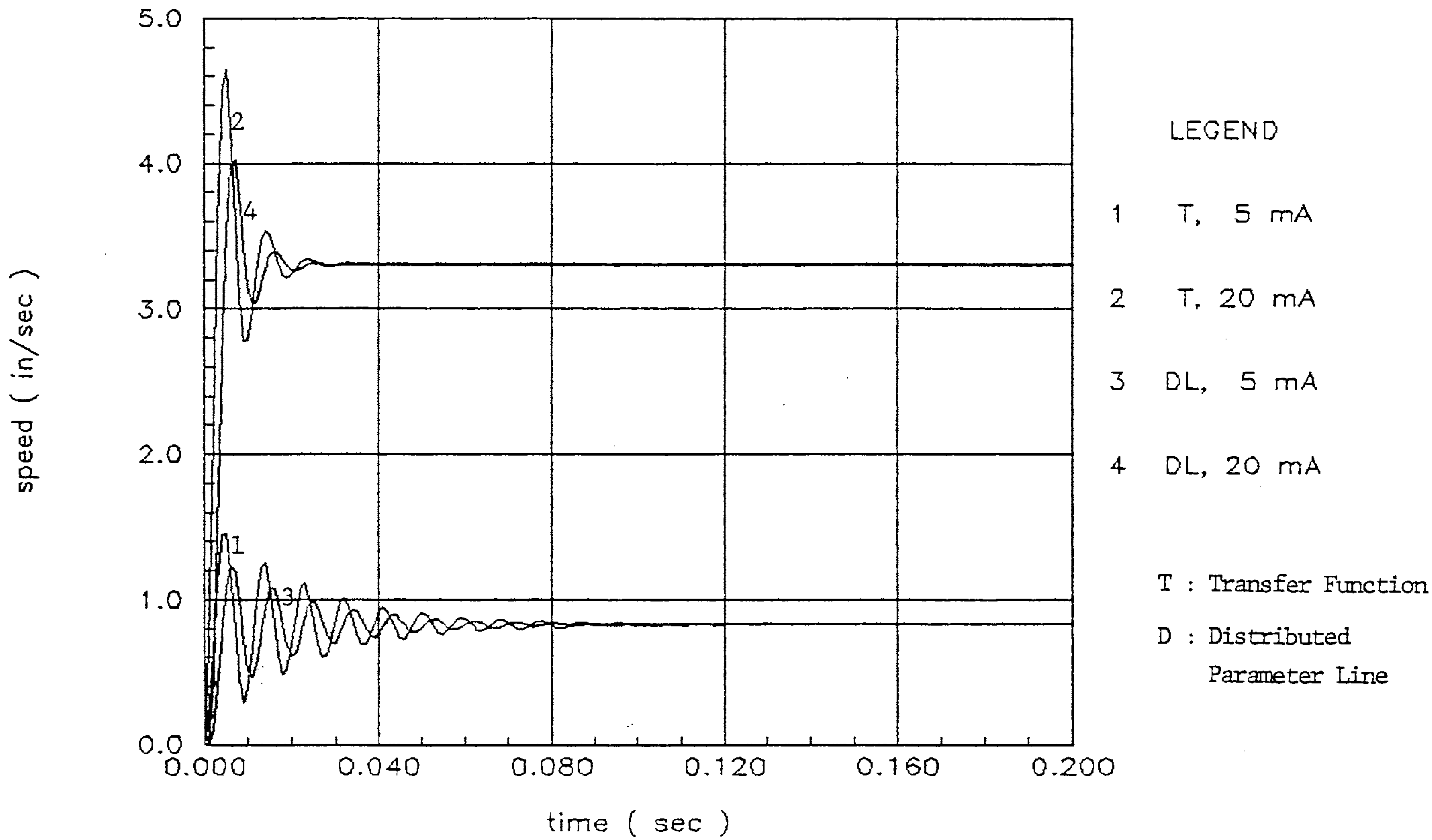

Figure 41. Dynamic Responses - Transfer Function and Distributed Components with Dist. Parameter Line (L = 10 inch)

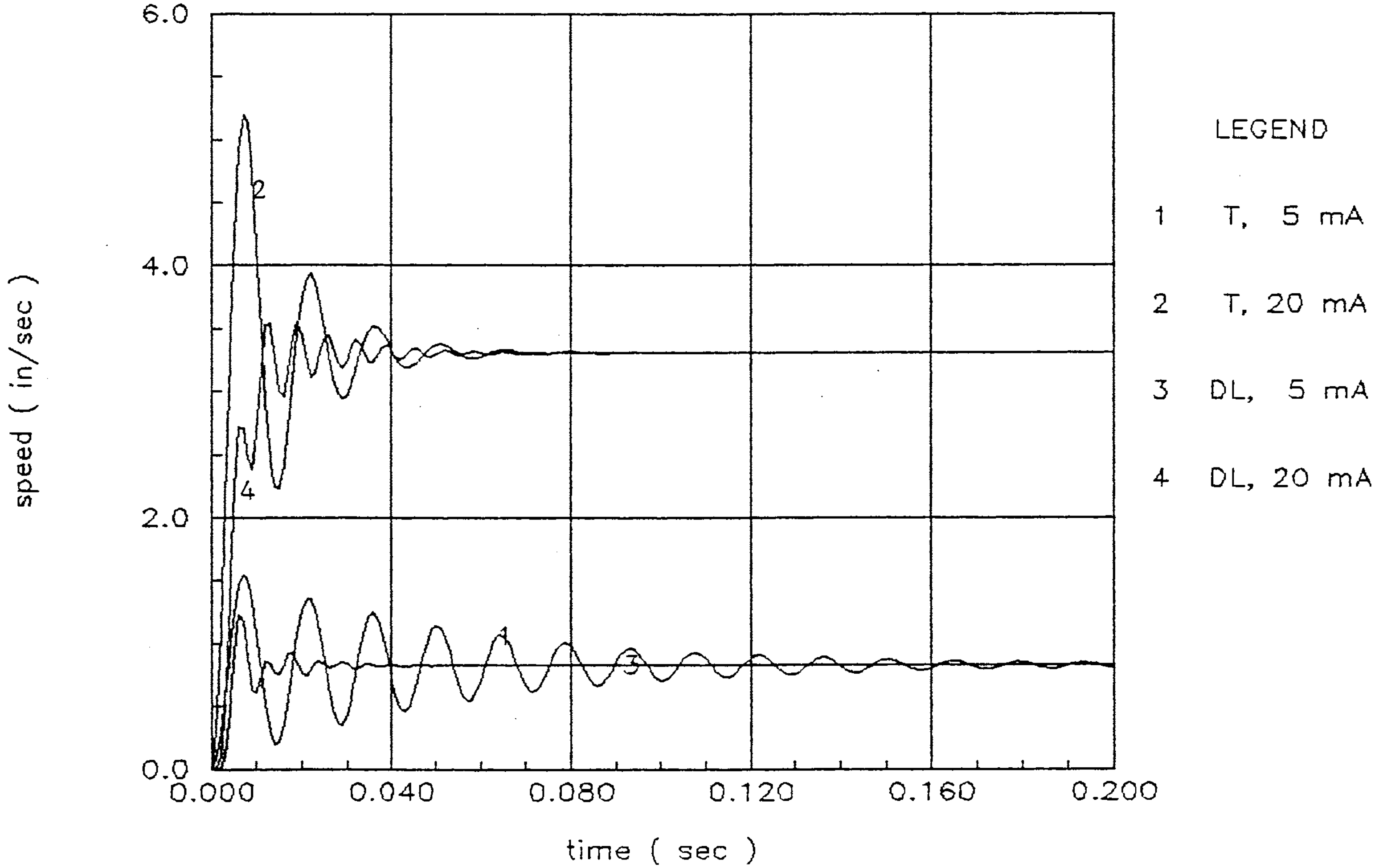

Figure 42. Dynamic Responses - Transfer Function and Distributed Components with Dist. Parameter Line (L = 100 inch)

$$f = \frac{C_o}{2L_t} = 108 \text{ Hz}, \tag{97}$$

the line dynamics greatly affects the total system response. The hydraulic natural frequency is approximately 100 Hz.

It is certain that the linearized transfer function method can be used to obtain the response in time domain when the line connecting the valve and the actuator is short. But, as the line become longer distributed parameters line modelling is necessary for accurate analysis.

Frequency Domain Analysis

Linearized Transfer Function with Lumped Mass Line

Equation (31) results in a transfer function between the servovalve current and the piston speed which can be written in the following manner:

$$G(s) = \frac{\frac{K_i}{A_p}}{\frac{s^2}{\omega_h^2} + \frac{2\delta_h}{\omega_h} s + 1} \tag{98}$$

By substituting $j\omega$ for s in the preceding equation and by incorporating the definition of logarithmic magnitude of $G(j\omega)$ the Bode diagam can be drawn.

$$\text{magnitude ratio (dB)} = 20 \log \left| G(j\omega) \right| \tag{99}$$

$$= 20 \log \left| \frac{Ki}{Ap} \right| \tag{100}$$

$$= - 20 \log \sqrt{\left(1 - \frac{\omega^2}{\omega h^2} \right)^2 + \left(2\delta h \frac{\omega}{\omega h} \right)^2}$$

$$\text{phase angle (degree)} = \underline{/ G(j\omega)} \tag{101}$$

$$= -90 - \tan^{-1} \left[\frac{2\delta h \frac{\omega}{\omega h}}{1 - \left(\frac{\omega}{\omega h} \right)^2} \right] \tag{102}$$

Figures 43 through 48 are the Bode Diagrams which reveal the frequency responses of the system. As the line increases in length, significant phase lag occurs. Also, the damping ratio decreases as in time domain analysis. See Appendix F for the computer program.

Linearized Transfer Function with Lumped Parameter Line Modelling

If lumped parameter line modelling with lossless line theory is used [15], the linearized transfer function of the system can be derived in the following manner. (See Appendix G)

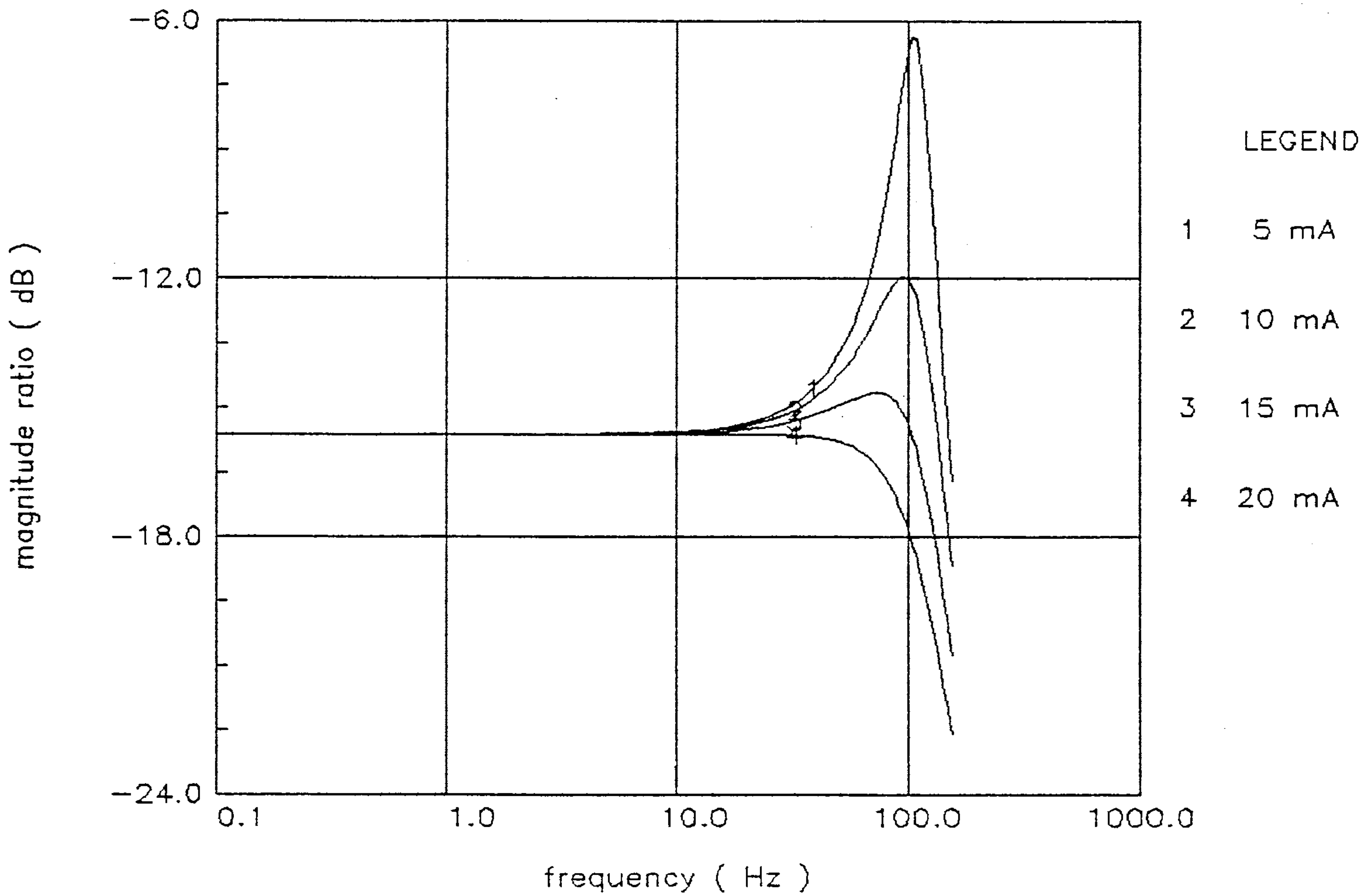

Figure 43. Frequency Response - Transfer Function (L = 10 inch)

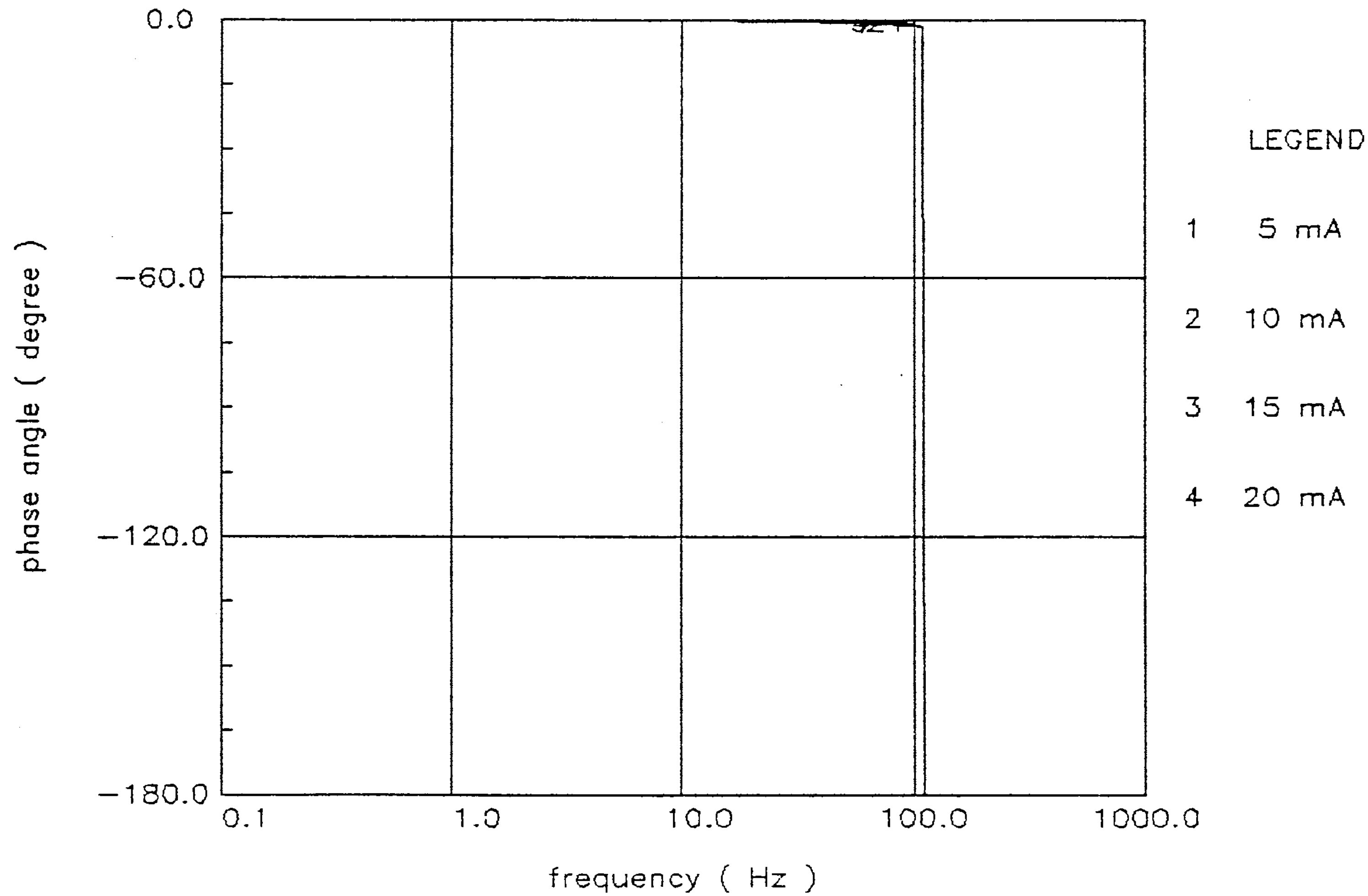

Figure 44. Frequency Response - Transfer Function (L = 10 inch)

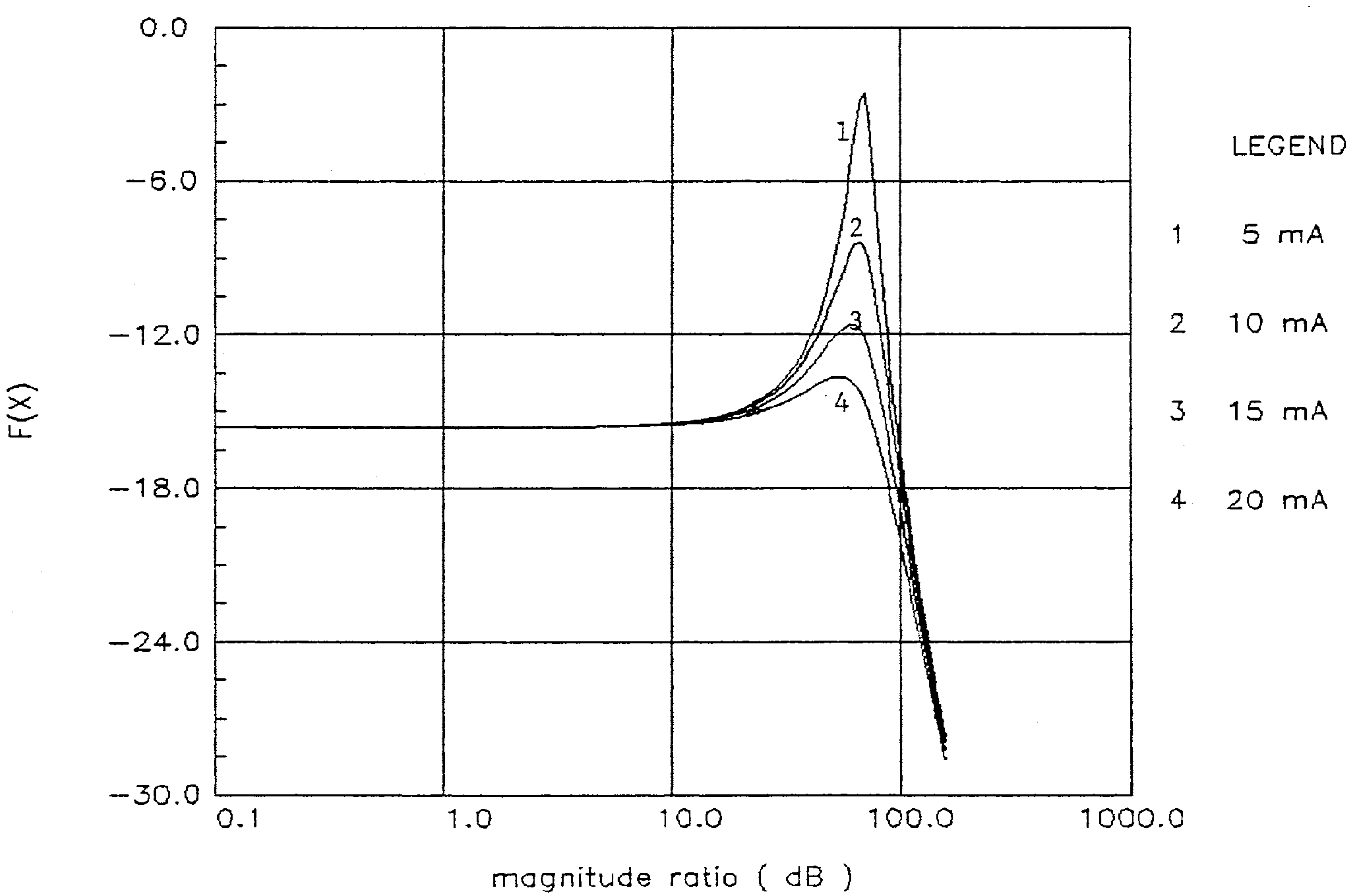

Figure 45. Frequency Responses - Transfer Function (L = 100 inch)

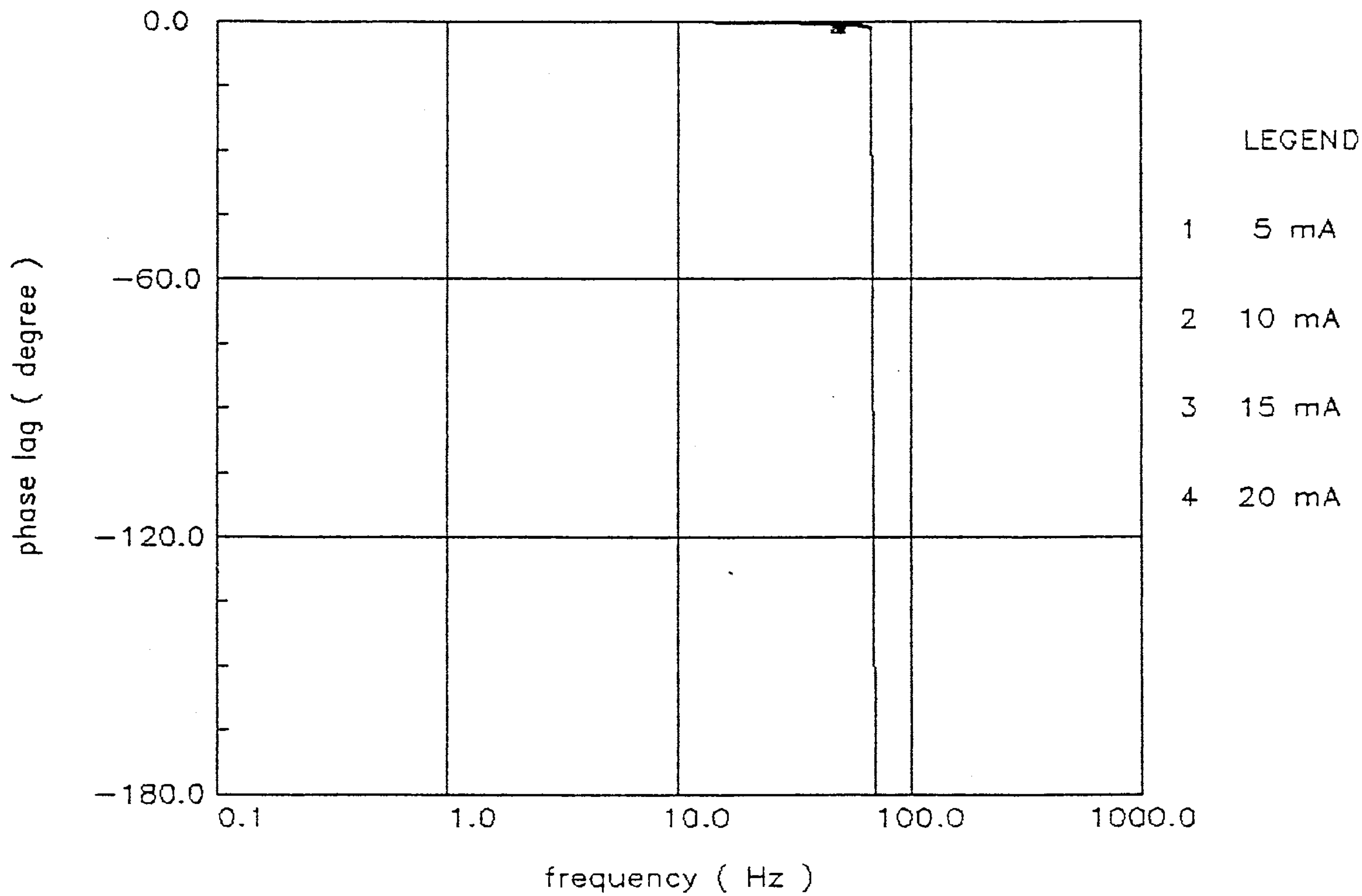

Figure 46. Frequency Responses - Transfer Function (L = 100 inch)

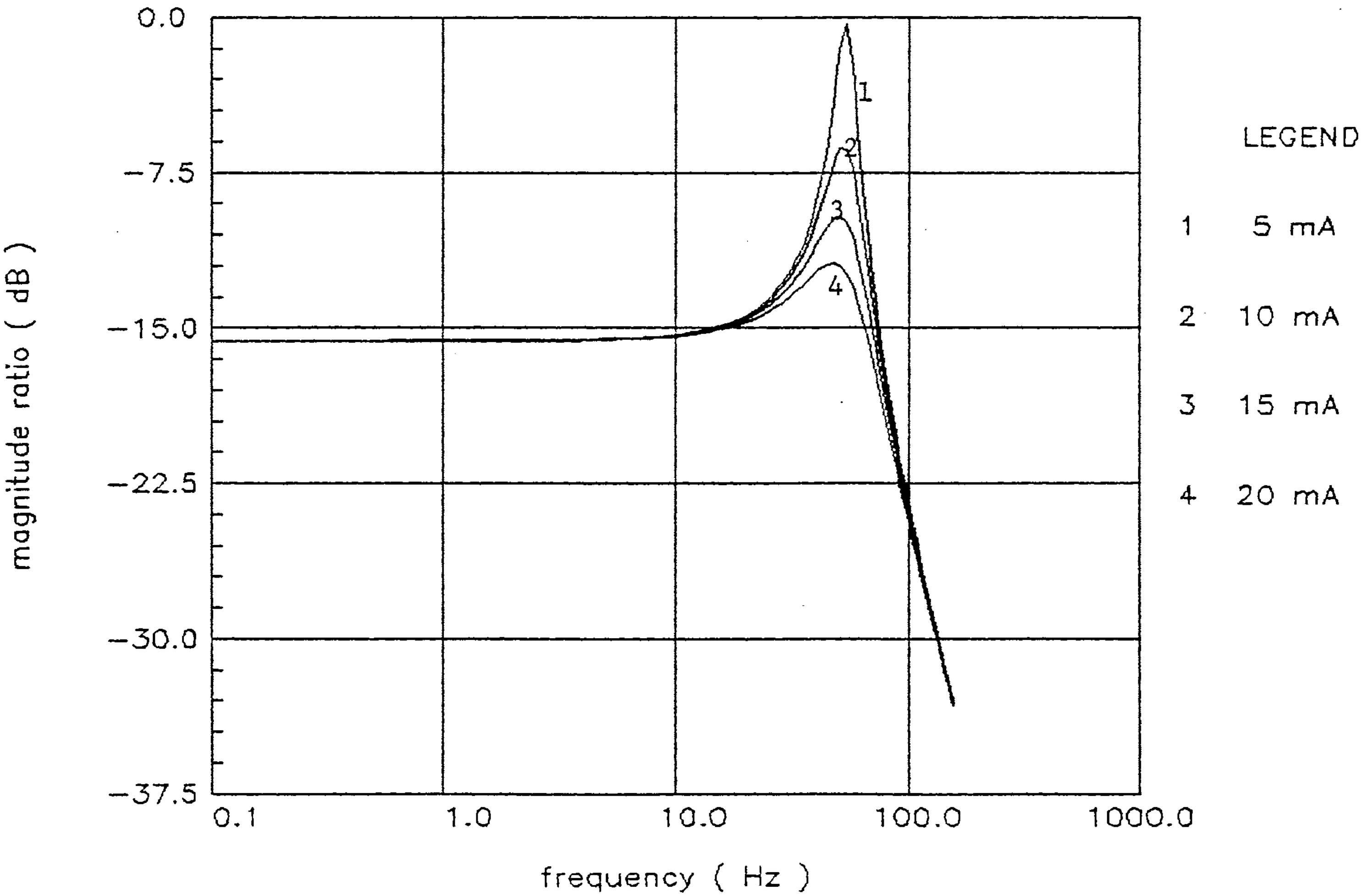

Figure 47. Frequency Responses - Transfer Function (L = 200 inch)

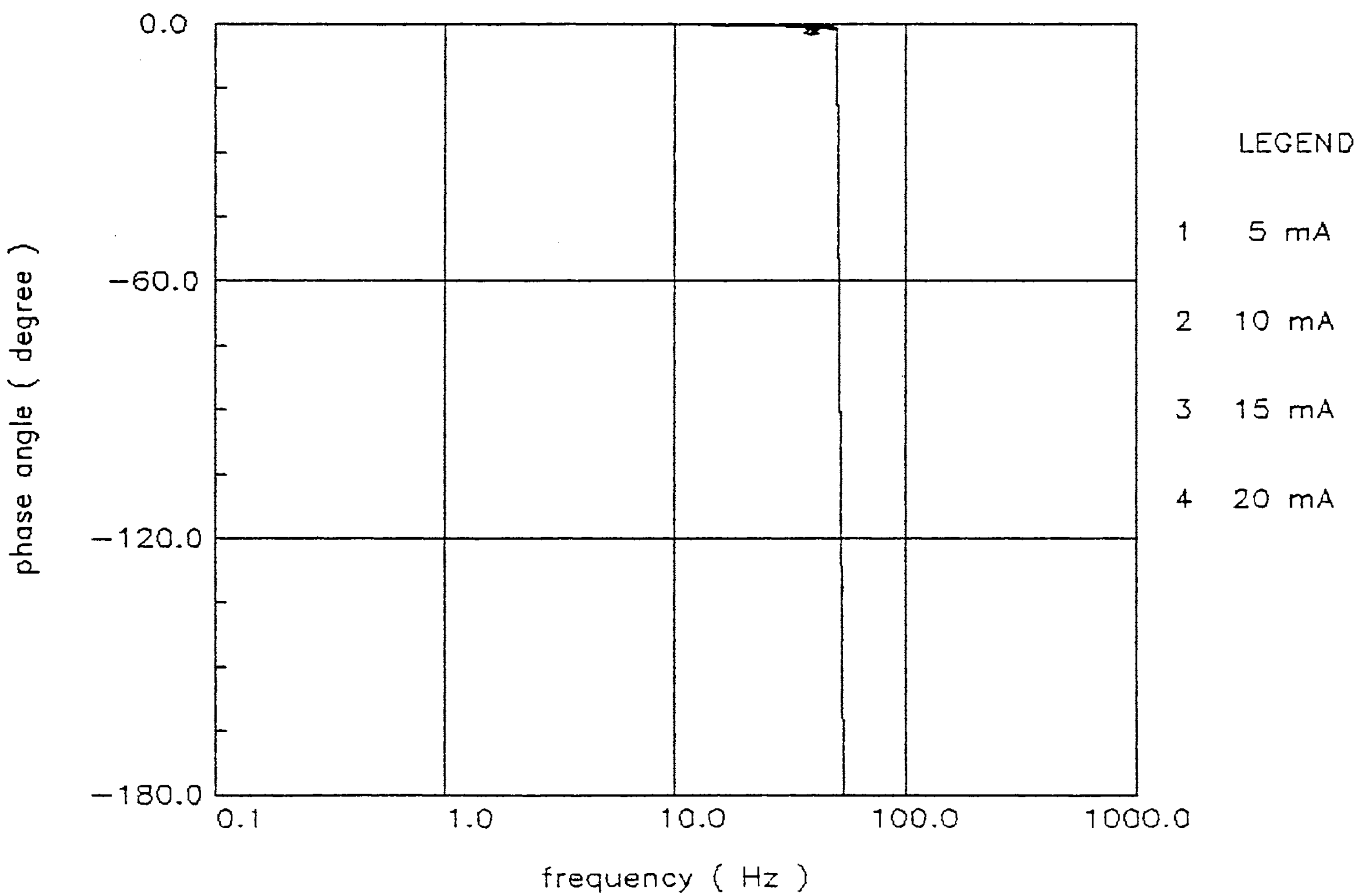

Figure 48. Frequency Responses - Transfer Function (L = 200 inch)

$$\frac{\delta V_p}{\delta I} = \frac{\dfrac{K_i}{A_p}}{\cosh(\Gamma l) + \dfrac{Z_c}{R_v}\sinh(\Gamma l) + \dfrac{sL_m}{2R_v}\left(1 + \dfrac{R_v}{Z_c}\right)\cosh(\Gamma l)} \qquad (103)$$

where, Γ : propogation factor, $\Gamma = s\sqrt{LC}$ (104)

Z_c : characteristic impedence,

$$Z_c = \sqrt{\frac{L}{C}} = Z_{ca} = LC_o \qquad (105)$$

C_o : velocity of sound propogation in the fluid,

$$C_o = \sqrt{\frac{\beta_e}{\rho}} \qquad (106)$$

R_v : valve resistance, $R_v = \dfrac{1}{K_c}$ (107)

L_m : load inductance, $L_m = \dfrac{M_t}{A_p^{\,2}}$ (108)

Therefore, if the inductance of one line is expressed as a ratio of the load inductance, i.e.

$$\alpha = \frac{L_l}{L_m} \qquad (109)$$

then assuming lossless-line theory means that the transfer function (103) may be written in the following manner:

$$\frac{\delta V_p(j\omega)}{\delta I(j\omega)} = \frac{\dfrac{K_i}{A_p}}{\cos(\bar{\omega}) + j\left[\dfrac{Z_{ca}}{R_v}\sin(\bar{\omega}) + \dfrac{\bar{\omega}}{2\alpha}\left(1 + \dfrac{Z_{ca}}{R_v}\cos(\bar{\omega})\right)\right]} \qquad (110)$$

Figures 49 through Figure 54 reveal the frequency responses of the system when the valve current and the line length change. They reveal that significant phase lag occurs as the line increases in length and that damping ratio decreases at a same valve current. In a system using 10 inches line, the response reveals far less damping ratio than that of time domain response. Therefore, this linearized transfer function method can not be applied to a system with short lines. Compared to the results from lumped mass of lines (see Figure 55 through Figure 58), the natural frequency is predicted at approximately the same value as in the 200 inche lines. However it cannot be assumed that both the linearized transfer function with lumped mass of line and lumped parameter line modelling can predict the same natural frequency in the long line.

System Identification

Pulse Testing Method

Time Domain Response

In order to identify the system using pulse testing method, the input pulse of valve current must be used. The waterhammer effect is anticipated in the servovalve controlled cylinder system if a pulse current is given. When fluid flowing in a tube is suddenly stopped due to a rapid valve closure, a very large pressure transient, commonly reffered to as a pressure surge, may result. This

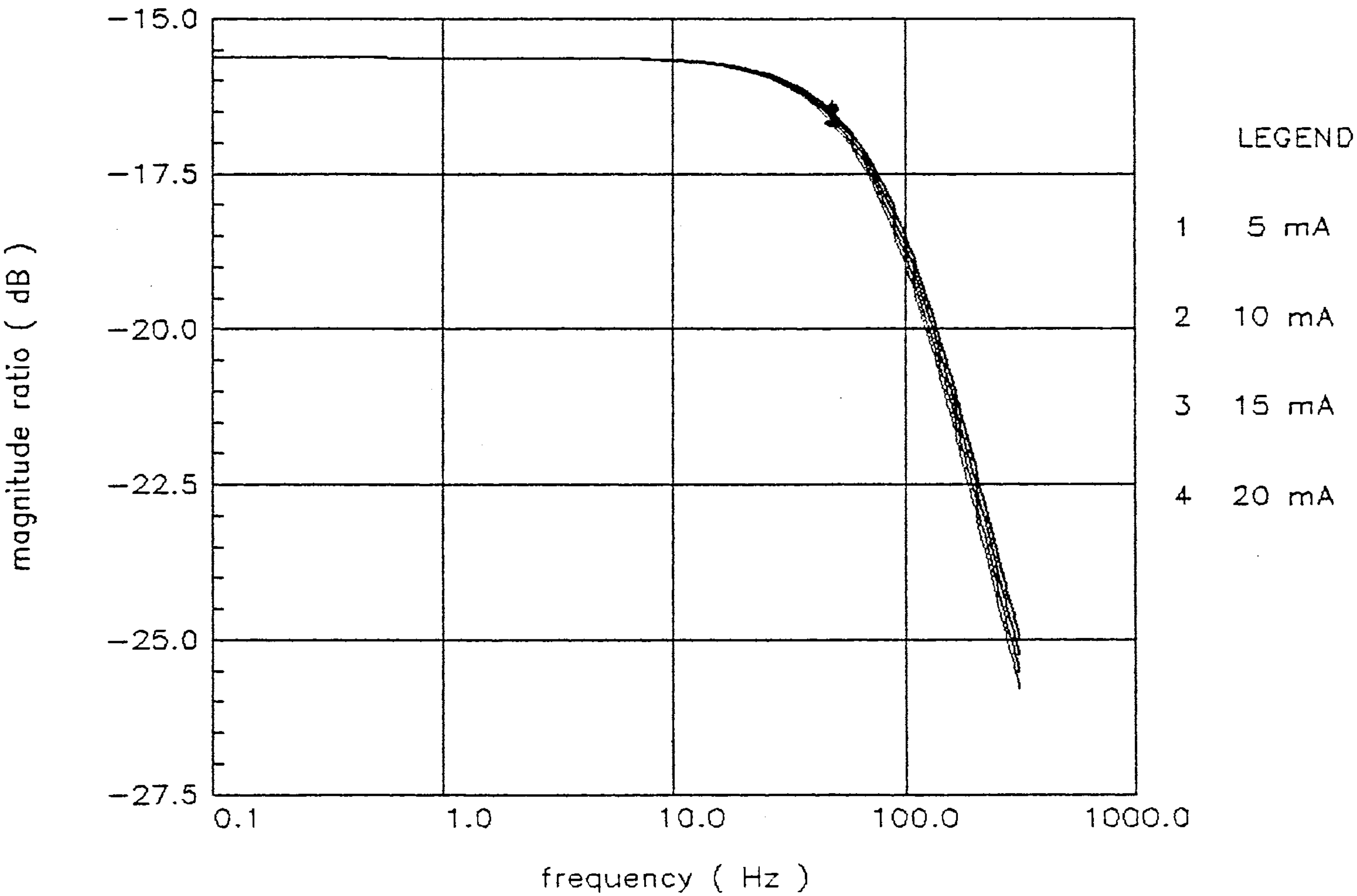

Figure 49. Frequency Responses - Lumped Parameter Line (L = 10 inch)

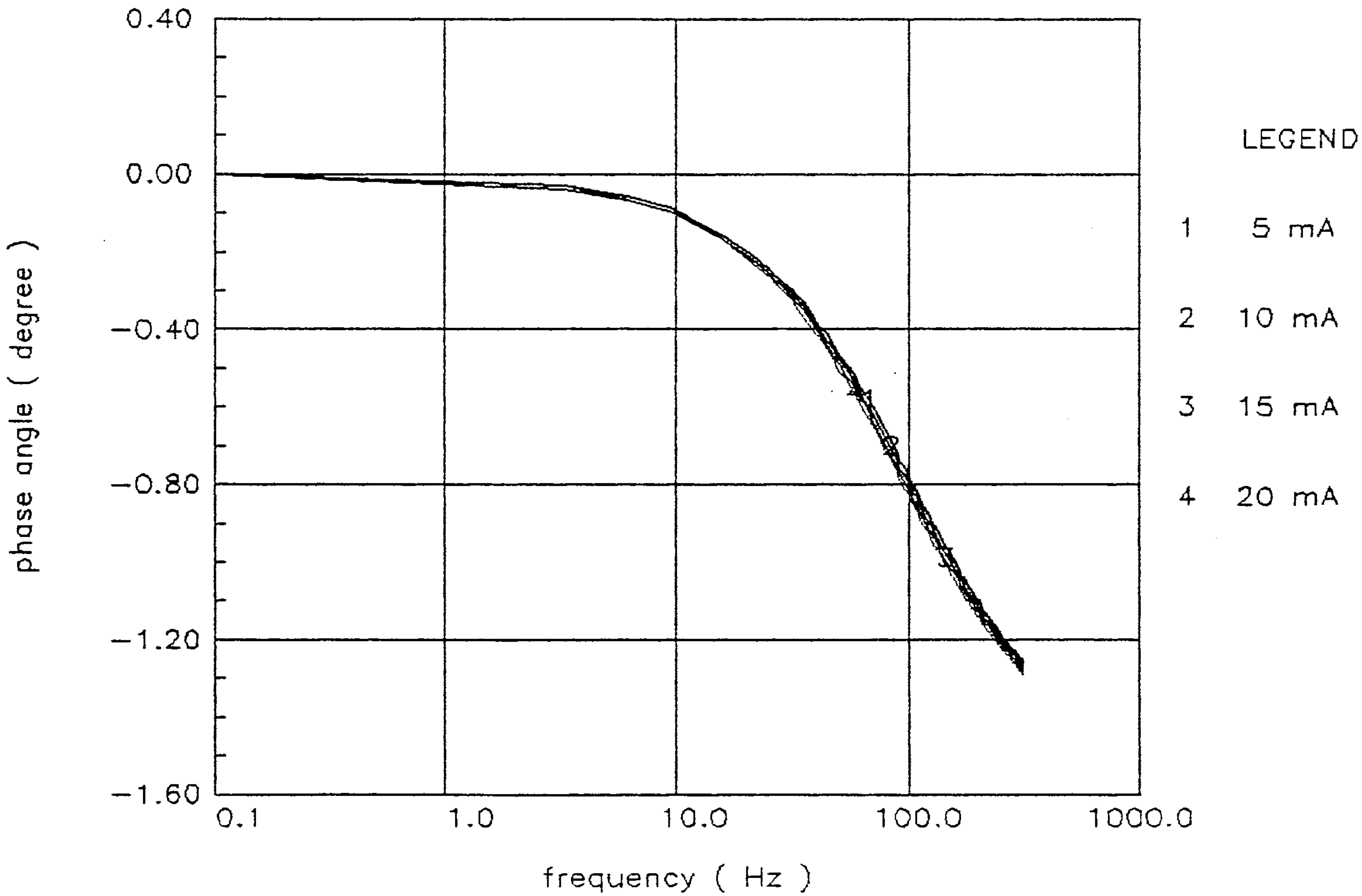

Figure 50. Frequency Responses - Lumped Parameter Line (L = 10 inch)

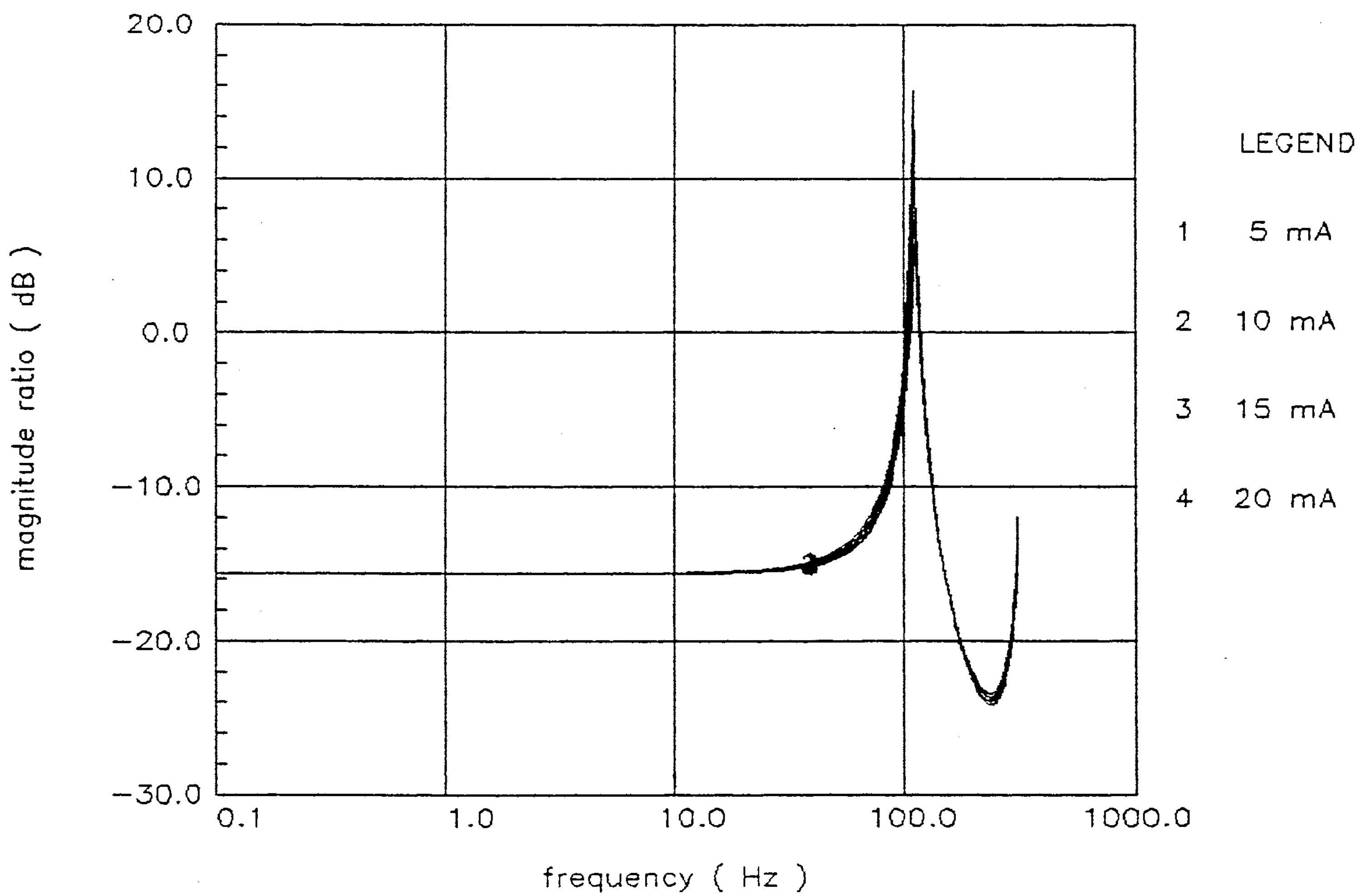

Figure 51. Frequency Responses - Lumped Parameter Line (L = 100 inch)

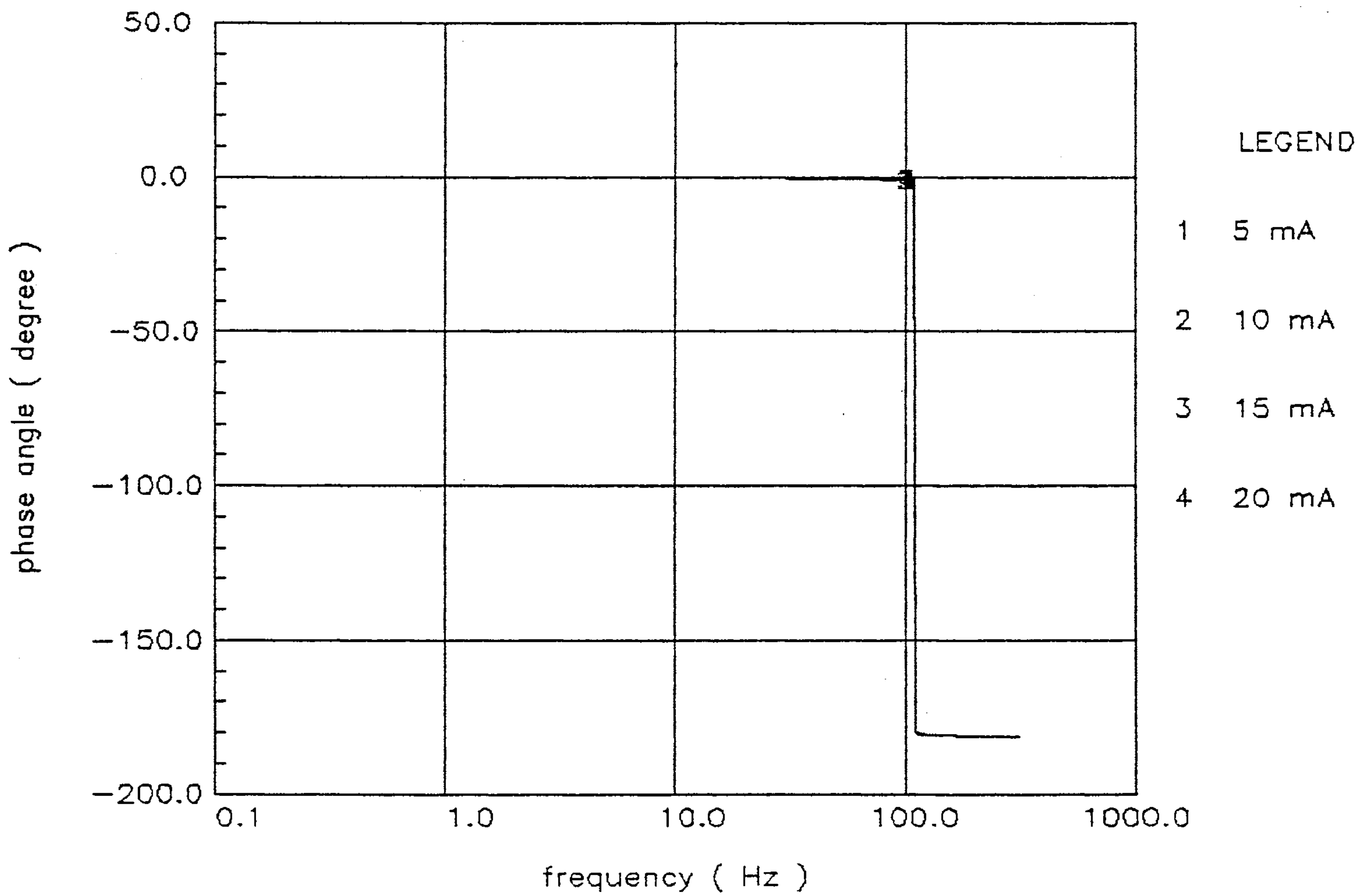

Figure 52. Frequency Responses - Lumped Parameter Line (L = 100 inch)

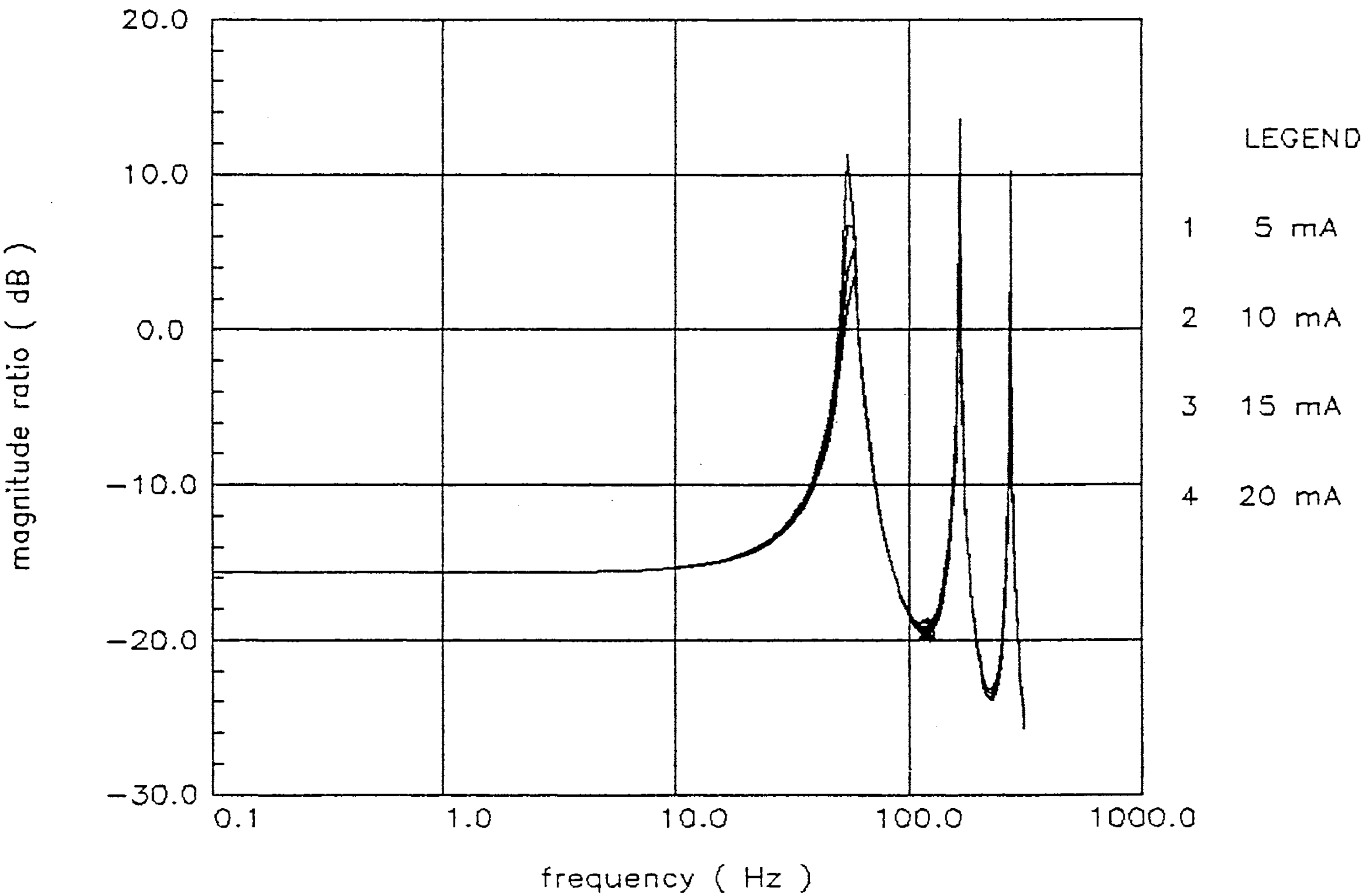

Figure 53. Frequency Responses - Lumped Parameter Line (L = 200 inch)

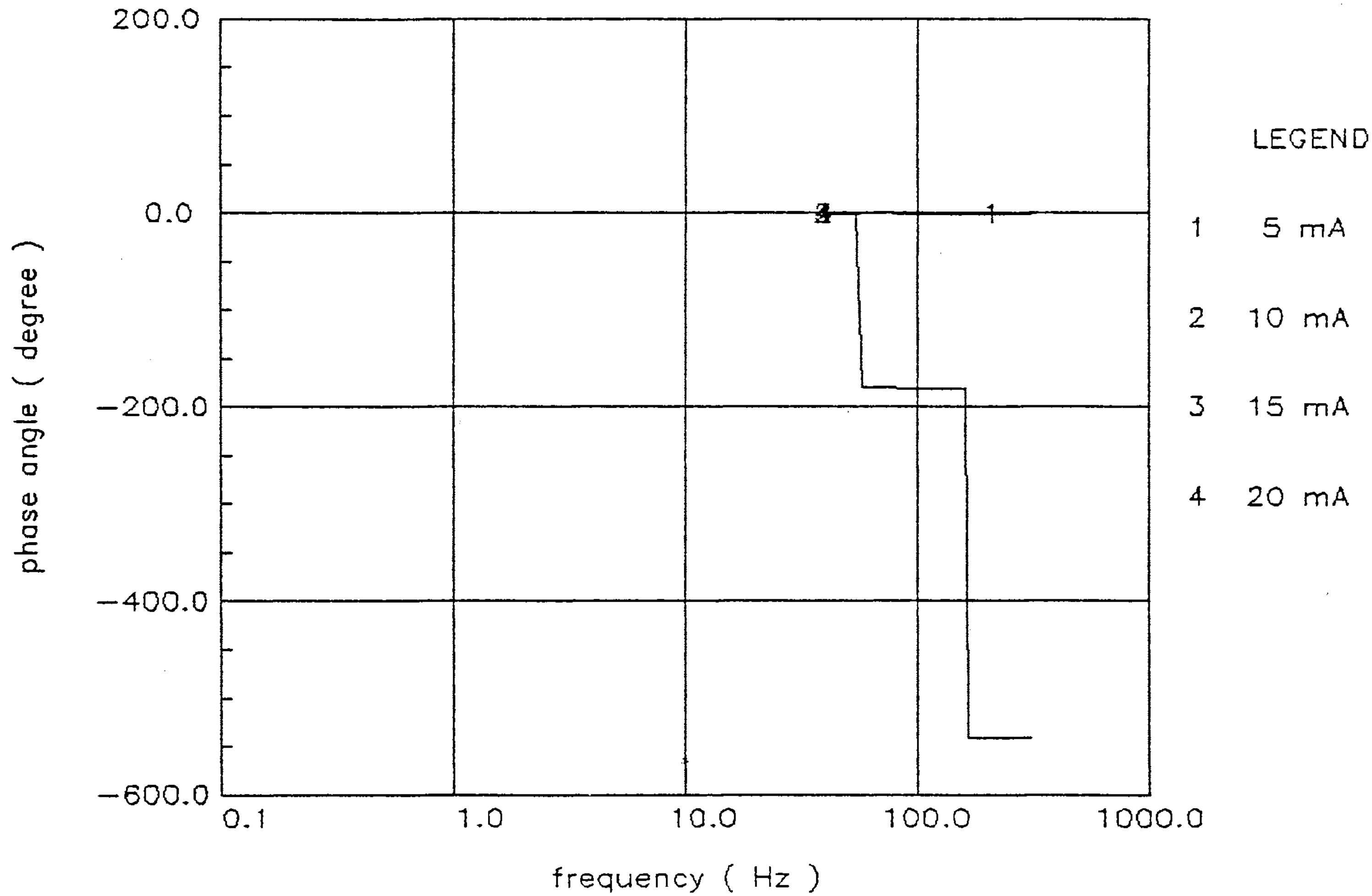

Figure 54. Frequency Responses - Lumped Parameter Line (L = 200 inch)

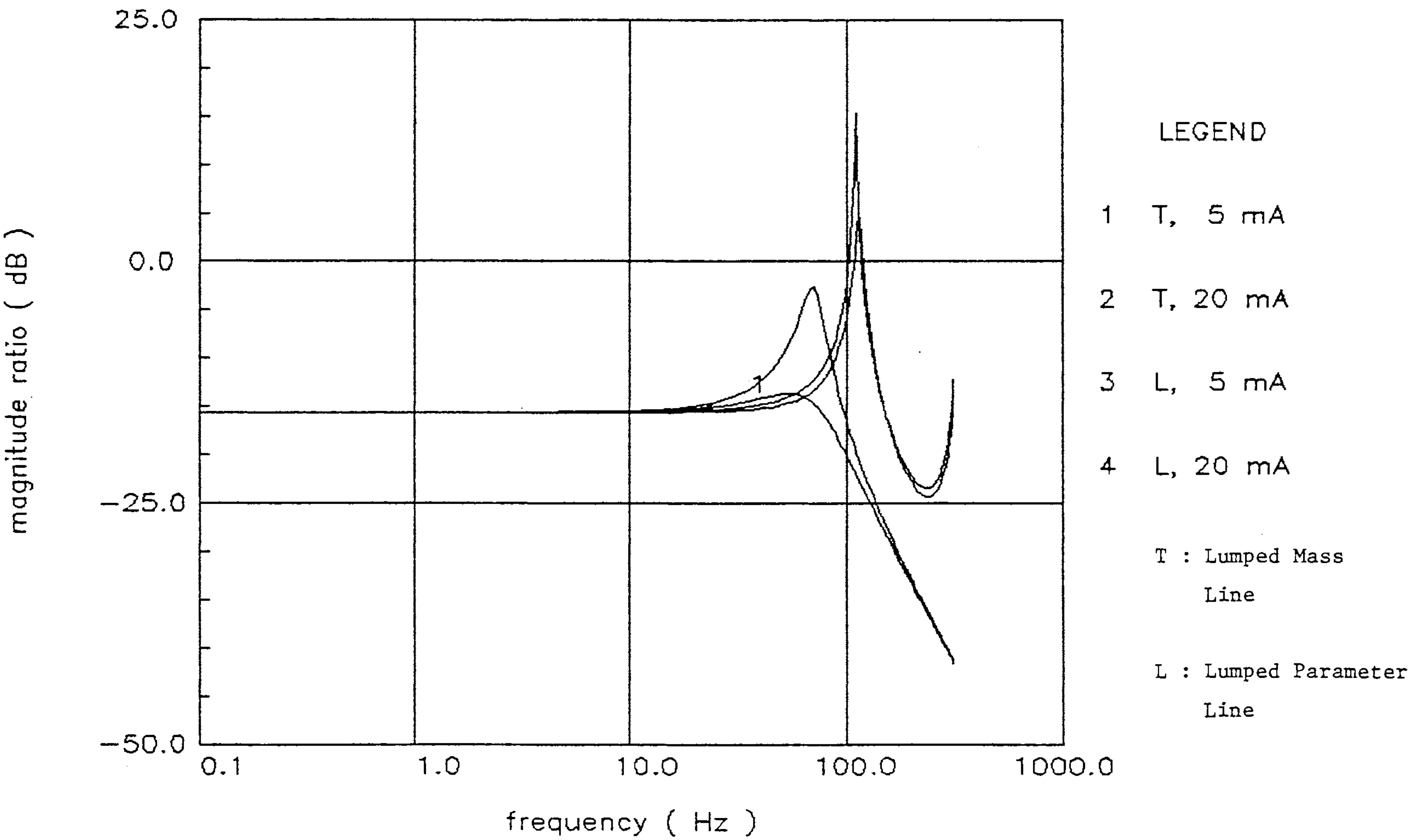

Figure 55. Frequency Responses - Transfer Function with Lumped Mass Line and Lumped Parameter Line (L = 100 inch)

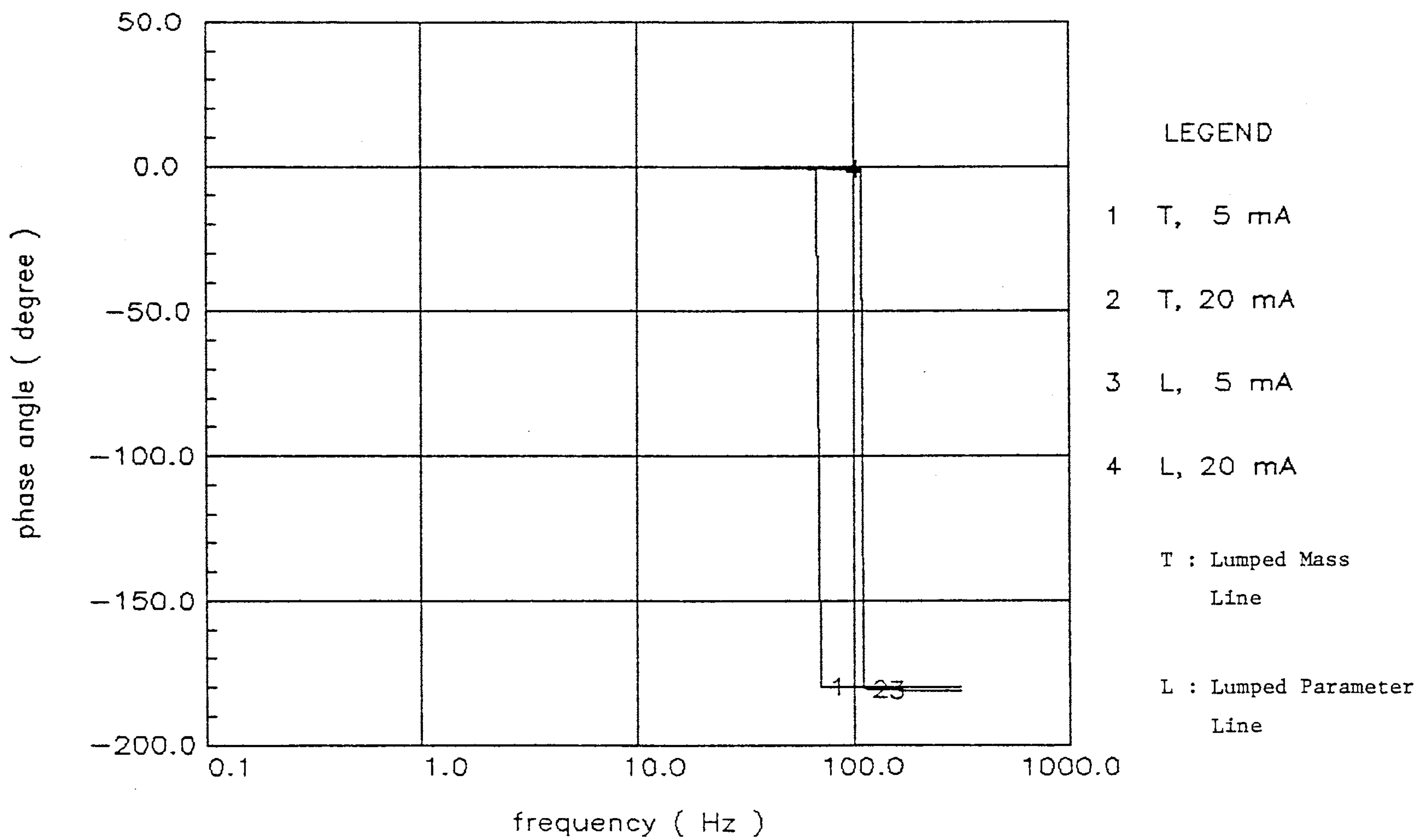

Figure 56. Frequency Responses - Transfer Function with Lumped Mass Line and Lumped Parameter Line (L = 100 inch)

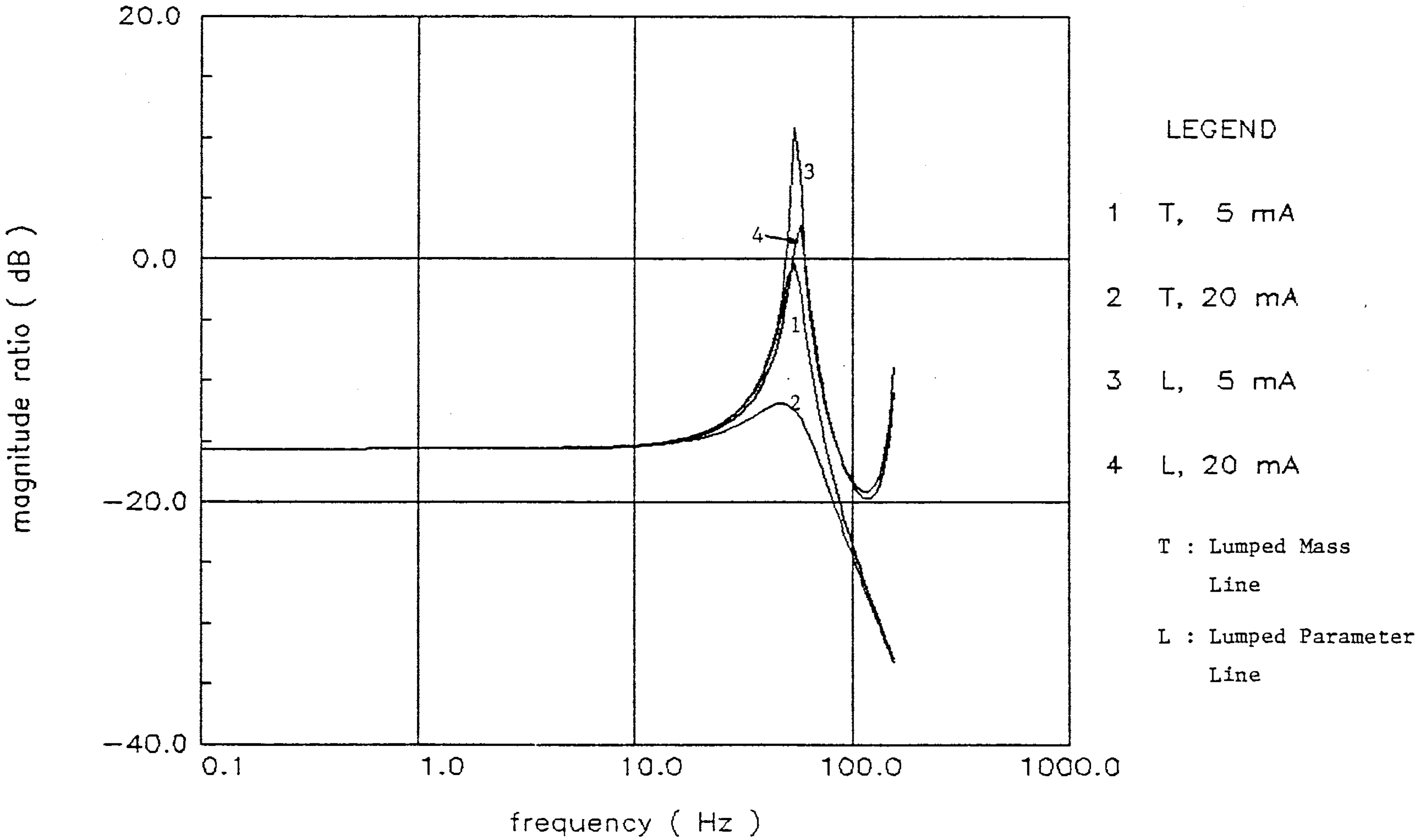

Figure 57. Frequency Responses - Transfer Function with Lumped Mass Line and Lumped Parameter Line (L = 200 inch)

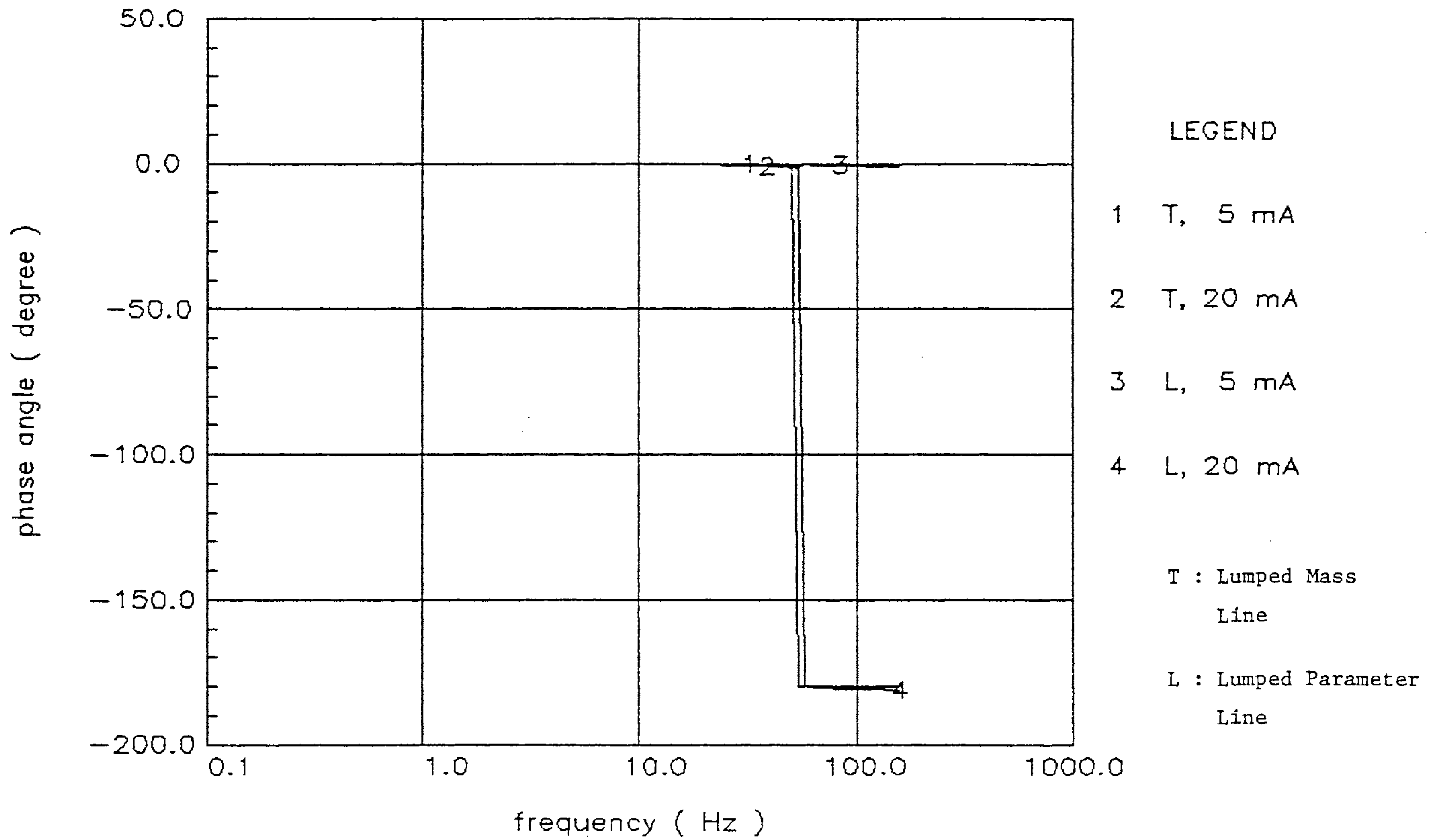

Figure 58. Frequency Responses - Transfer Function with Lumped Mass Line and Lumped Parameter Line (L = 200 inch)

phenomenon is called waterhammer. Because of inertia the piston continues to move and compresses the fluid in the return chamber, which causes the pressure to increase rapidly above the steady-state level. Simultaneously, the pressure in the forward chamber is decreased below the steady-state level. The piston will come to a stop and reverse direction when the kinetic energy of the moving mass is stored in the two fluid springs as potential energy. The piston continue to travel back and forth with the associated interchanges of kinetic and potential energies until friction and leakage losses dissipate the energy involved. The simulation results shown in Figures 59 and 60 reveal that waterhammer occured in the system (see Appendix H).

Frequency Domain Response

Figures 61 through 66 reveal the frequency response results using pulse testing. The line lengths were 10 inches, 100 inches, and 200 inches at 10 mA of valve current. The results reveal that pulse testing can predict the natural frequencies as well as the real system. However, in the high frequency range it is susceptible to noise. When compared with the results determined utilizing the linearized transfer function (see Figures 67 through 69), the similarity of the conclusions reveal that pulse testing can be used to identify the system. See Appendix I for the computer program to draw the Bode Plot using pulse

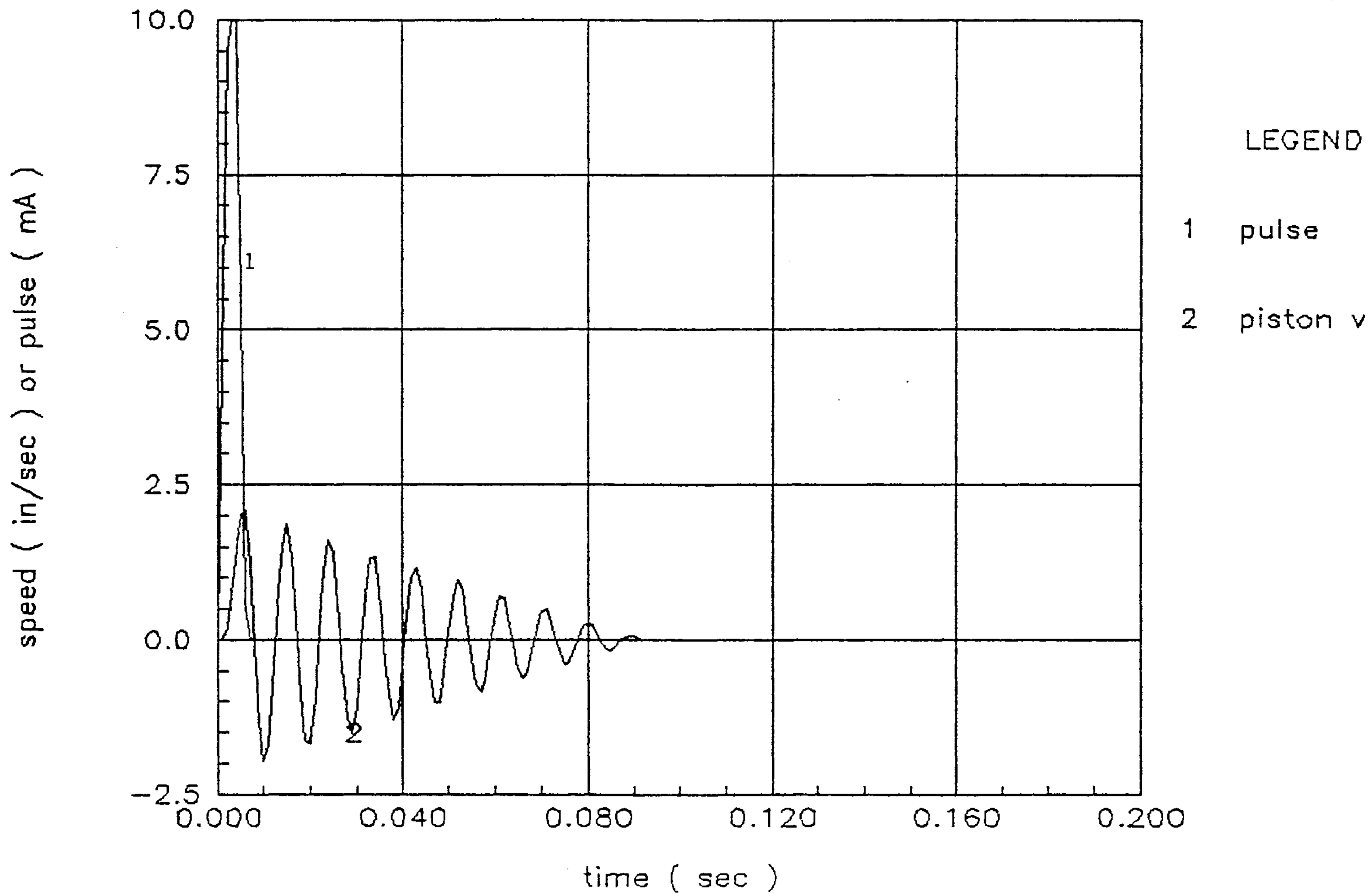

Figure 59. Waterhammer Effect

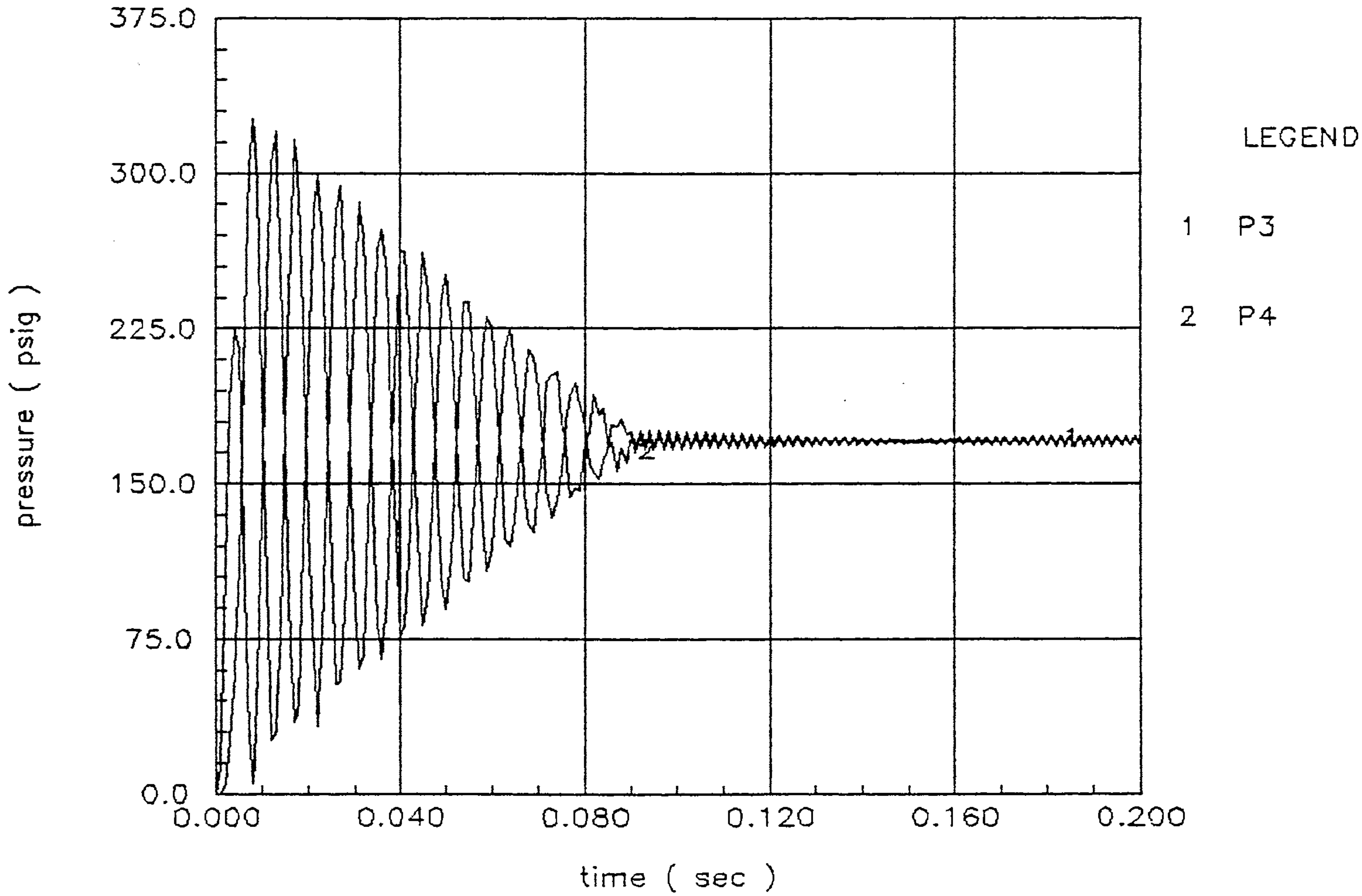

Figure 60. Waterhammer Effect

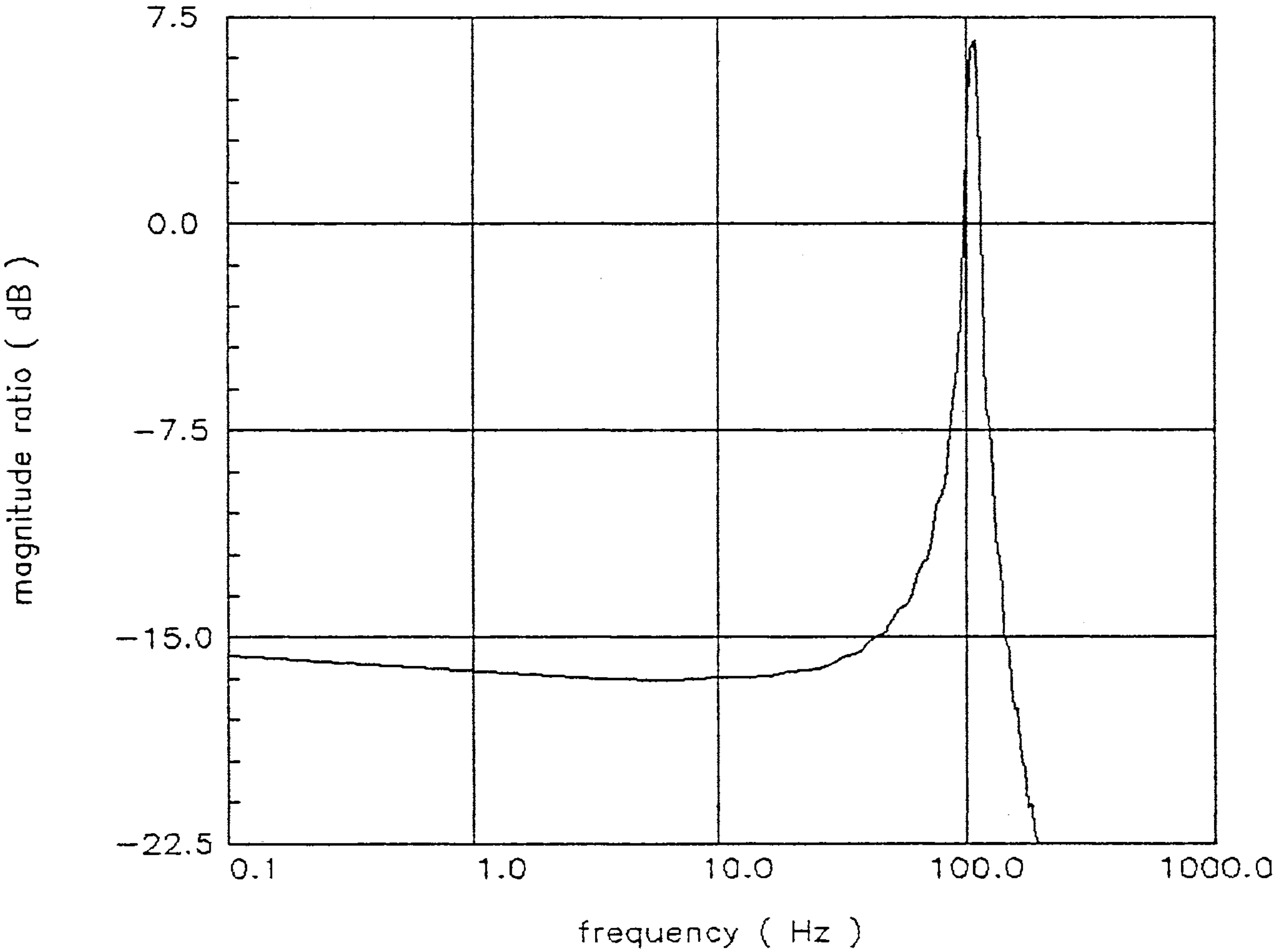

Figure 61. Frequency Response - Pulse Testing (L = 10 inch, I = 10 mA)

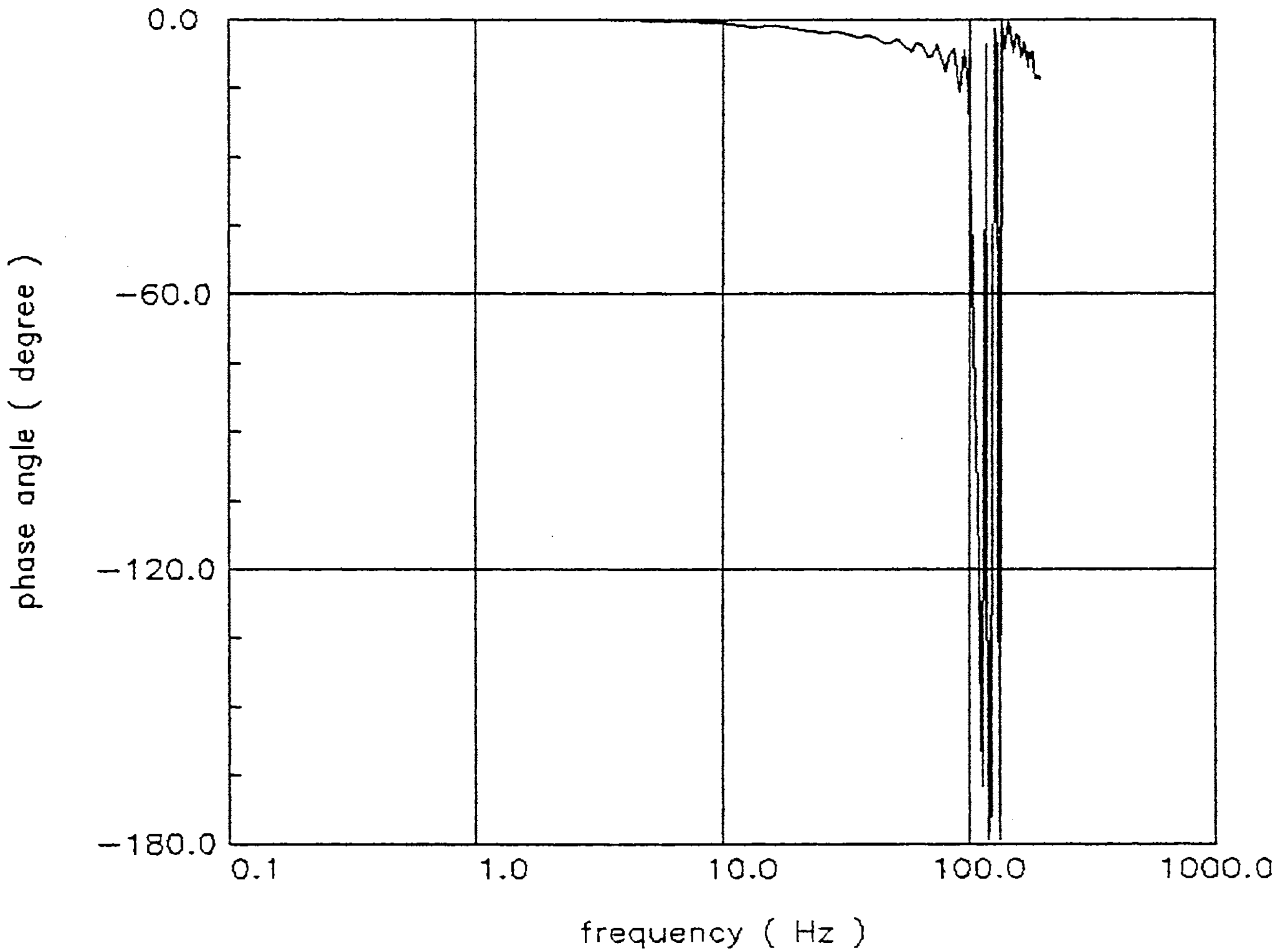

Figure 62. Frequency Response - Pulse Testing (L = 10 inch, I = 10 mA)

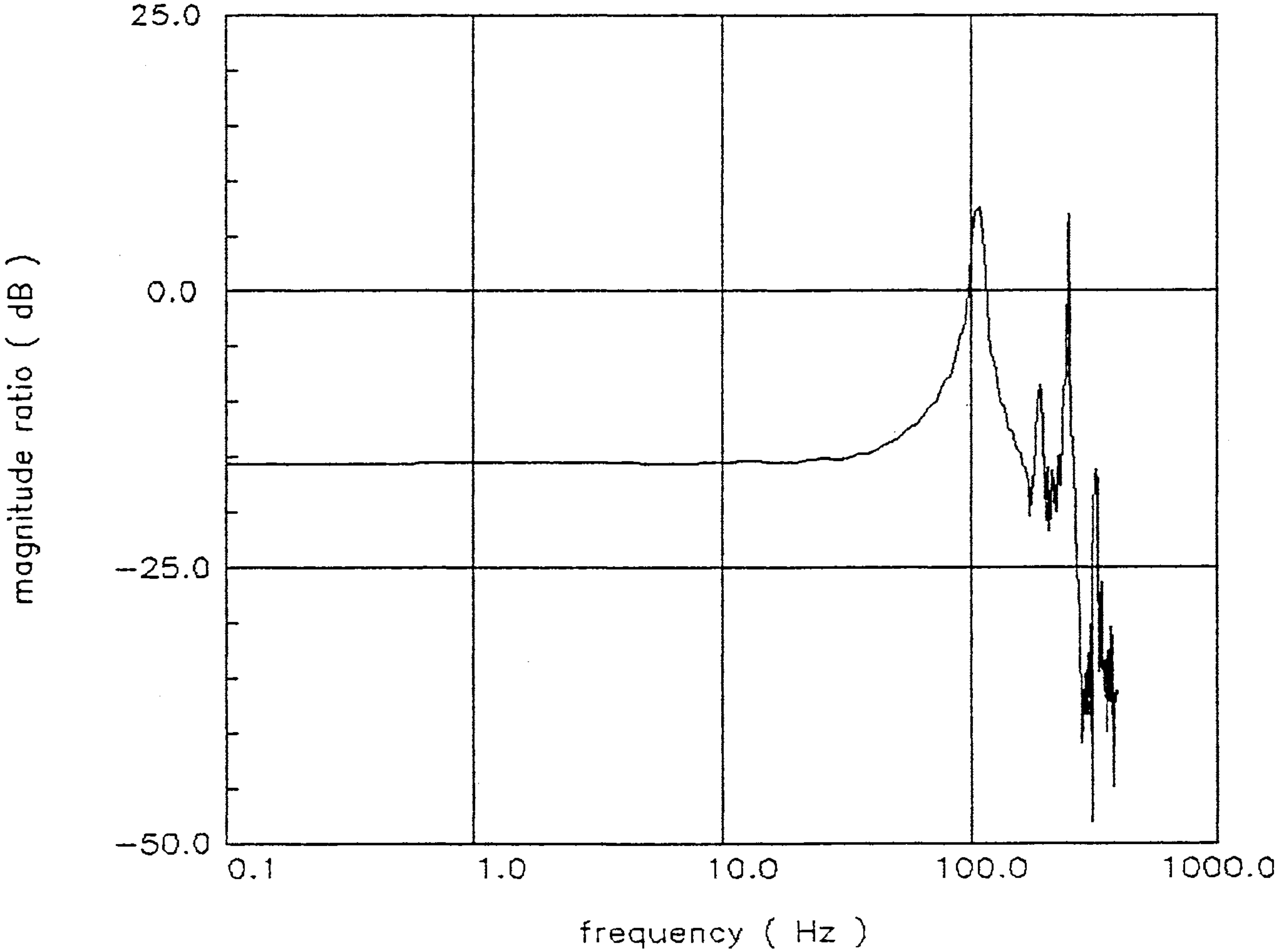

Figure 63.Frequency Response - Pulse Testing (L = 100 inch, I = 10 mA)

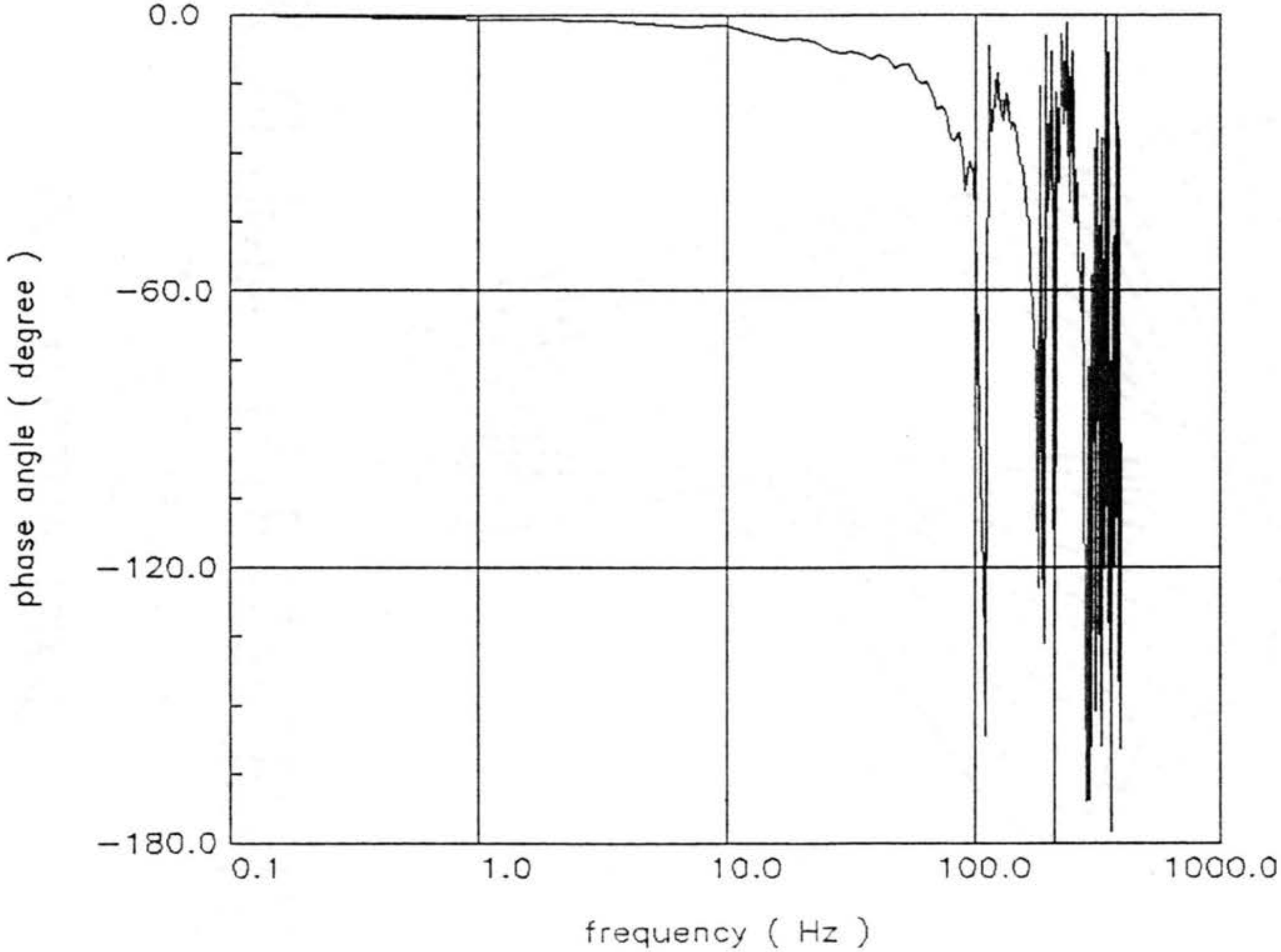

Figure 64. Frequency Response - Pulse Testing (L = 100 inch, I = 10 mA)

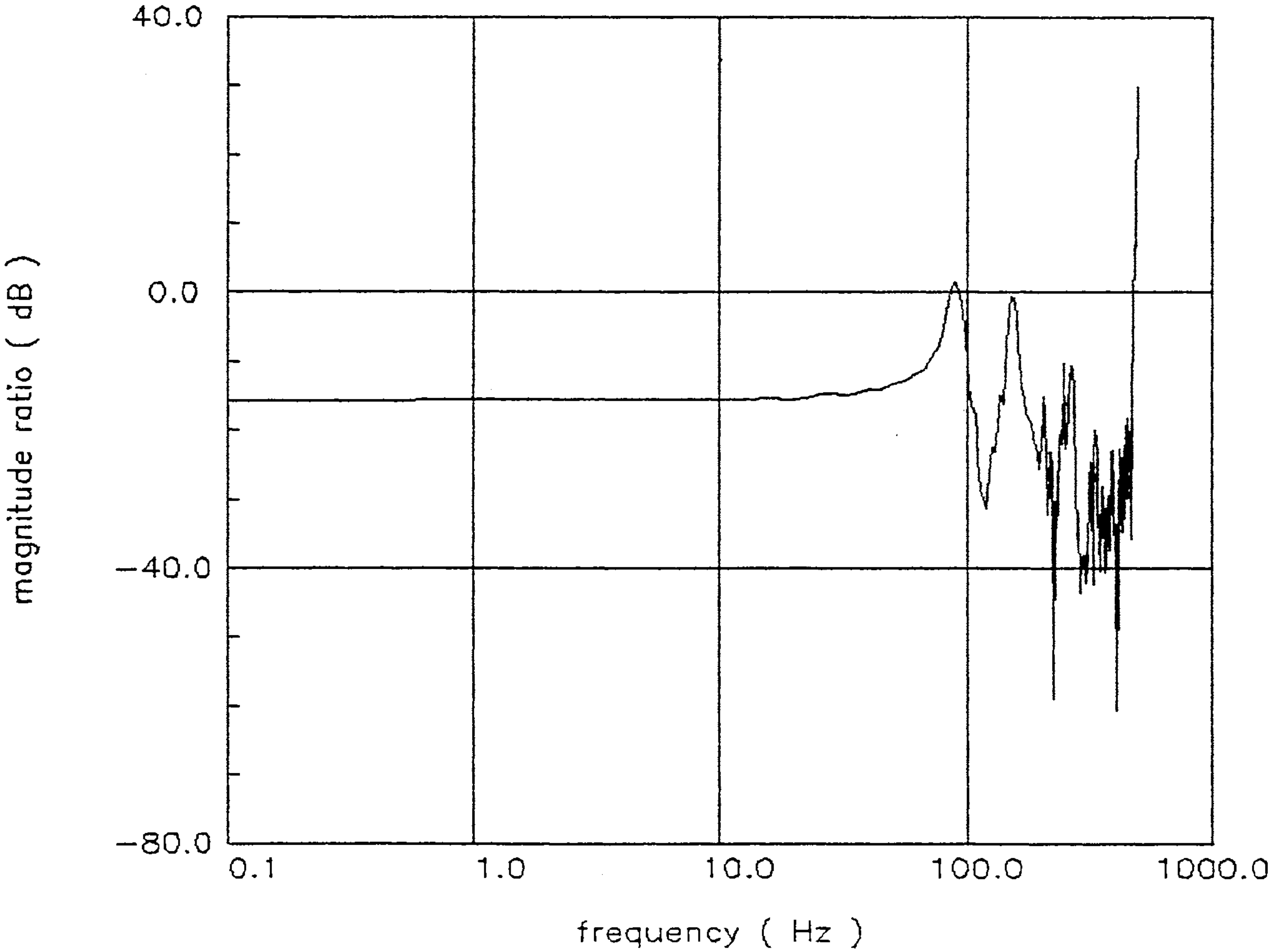

Figure 65. Frequency Response - Pulse Testing (L = 200 inch, I = 20 mA)

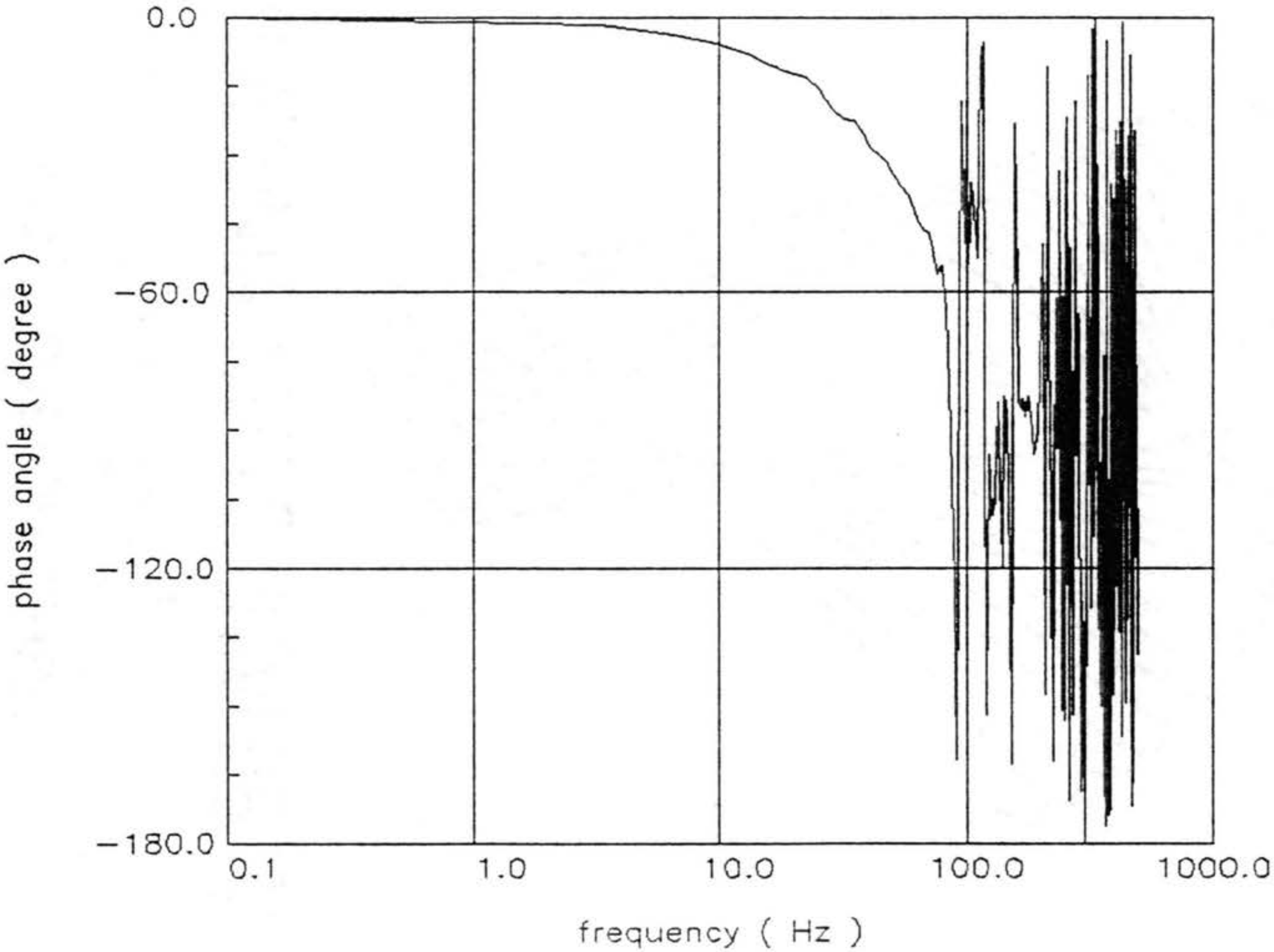

Figure 66. Frequency Response - Pulse Testing (L = 200 inch, I = 20 mA)

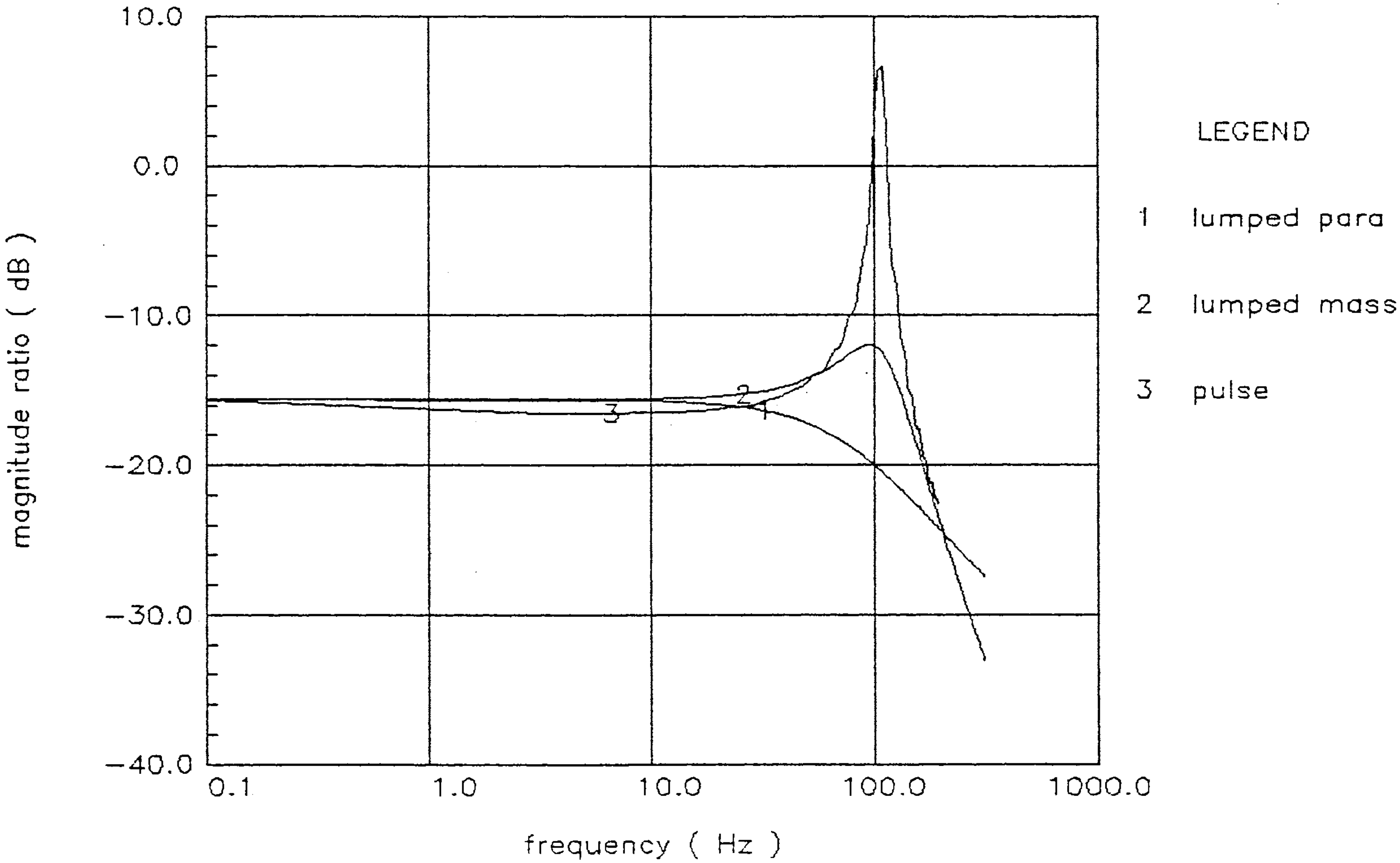

Figure 67. Comparision of Frequency Responses
(10 mA, 10 inches Line)

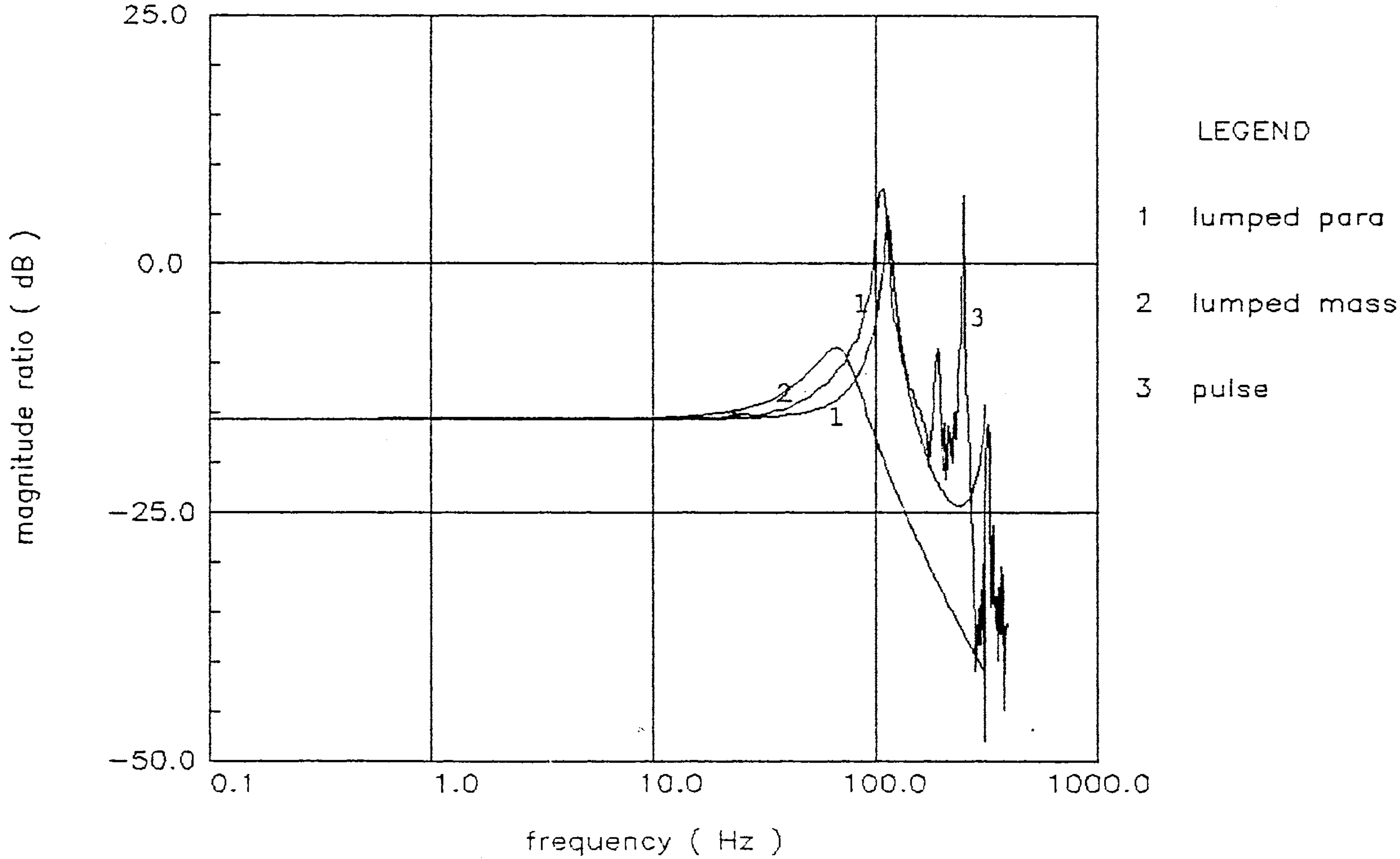

Figure 68. Comparision of Frequency Responses
(10 mA, 100 inches Line)

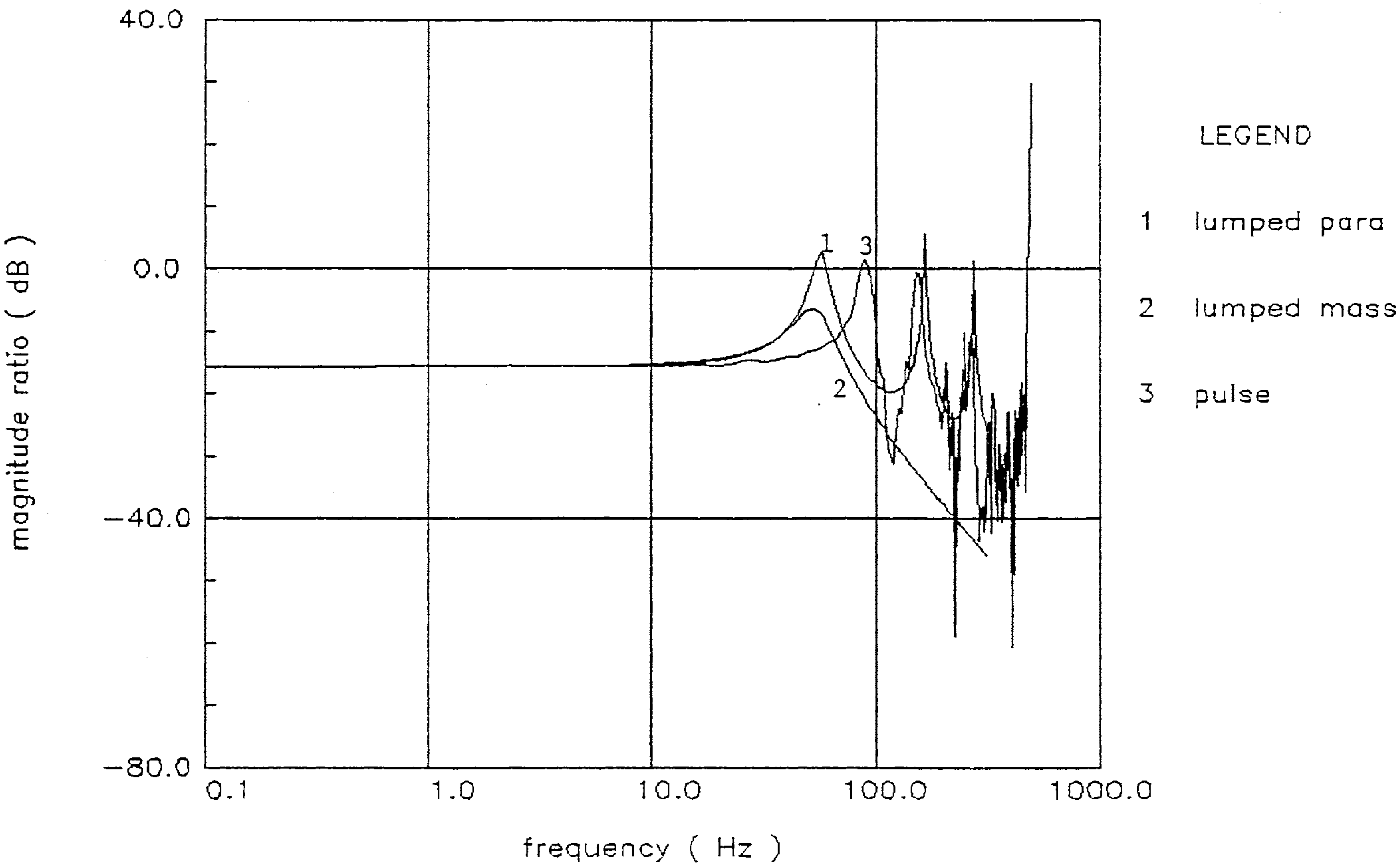

Figure 69. Comparision of Frequency Responses
(20 mA, 200 inches Line)

testing responses.

Parameter Estimation

After the system's order is identified, the parameters are estimated by using Orthogonal Projection Algorithm. For example the system of 10 inches line with 5 mA of valve current is selected.

Substituting the simulation values in Table II into equation (32), the transfer function is:

$$\frac{\dot{x}_P}{I} = \frac{19165.8}{s^2 + 91.1s + 115838.1} \qquad (111)$$

And, the Laplace transform of second order system of

$$F(s) = \frac{\beta}{(s+\alpha)^2 + \beta^2} \qquad (112)$$

is equivalent to the Z transform below.

$$F(z) = \frac{c\ z\ \sin(\beta h)}{z^2 - 2\ c\ z\ \cos(\beta h) + c^2} \qquad (113)$$

where,

$$c = e^{-\alpha h} \qquad (114)$$

h : discrete time step

Using the simulation data in Figure (34), when the current is 5 mA the parameters were estimated. Equations (77) and (78) are used for estimation (see Appendix J). The time step for estimation is 1 msec. The parameters are estimated from 1 msec, because there is a response delay of 1 msec in time domain response. The estimation results are shown in Appendix K. Although there remains small perturbation, the parameters can be determined in the following manner:

$$\frac{\dot{x}p(z)}{I(z)} = \frac{0.0745\ z}{z^2 - 1.43\ z + 0.88} \qquad (115)$$

where,

$$I = \begin{cases} 0 & t < 0.001 \text{ sec} \\ 5 & t \geq 0.001 \text{ sec} \end{cases}$$

Based on the relationship between equation (112) and (113) the equation (115) is equivalent to:

$$\frac{xp(s)}{I(s)} = \frac{86380.6}{s^2 + 127.8\ s + 499698.6} \qquad (116)$$

The discrete response using equation (115) is compared with the continuous time domain response in Figure (33) at 5 mA, (see Figure 70). Certain responses and the damping constants are very similar but there is some discrepancy in natural frequency.

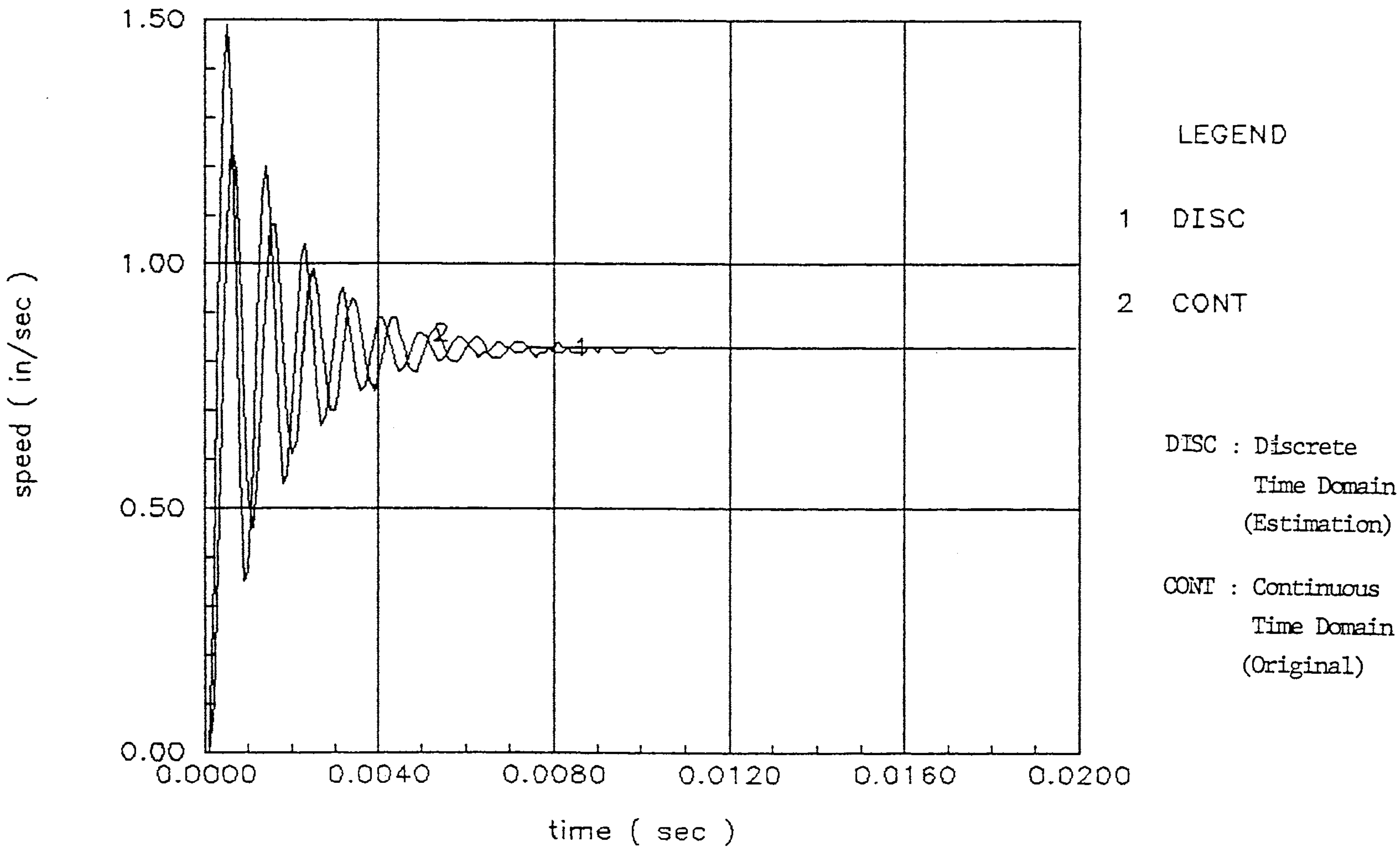

Figure 70. Dynamic Responses - Before & After Parameter Estimation

Control

Velocity Control

PID Feedback Control

The PID optimal gains are obtained using Powell's optimization method (see Appendix L). For the convergence criteria, 0.01 was used. The reference velocity was the steady-sate velocity at 5 mA valve current. The gains obtained are:

Proportional gain, $K_p = 12.98$

Integrational gain, $K_i = 0.44$

Derivative gain, $K_d = 39.60$

Figure 71 shows the velocity response of PID feedback control using the preceding PID gains (see Appendix M). Figure 72 shows the effect of optimized PID feedback control. The transient response was improved using PID.

VSS Control

According to the equations (89) and (90) for the existence of the sliding surface of the system (116), the sliding regime and control scheme are established in the following manner:

$$\sigma = 10X_1 + X_2 \qquad (117)$$

$$u = \psi_1 X_1 + \psi_1 X_2 \qquad (118)$$

where,

$$\psi_1 = \begin{cases} 30, & \text{if } \sigma X_1 > 0 \\ -30, & \text{if } \sigma X_1 < 0 \end{cases} \qquad (119)$$

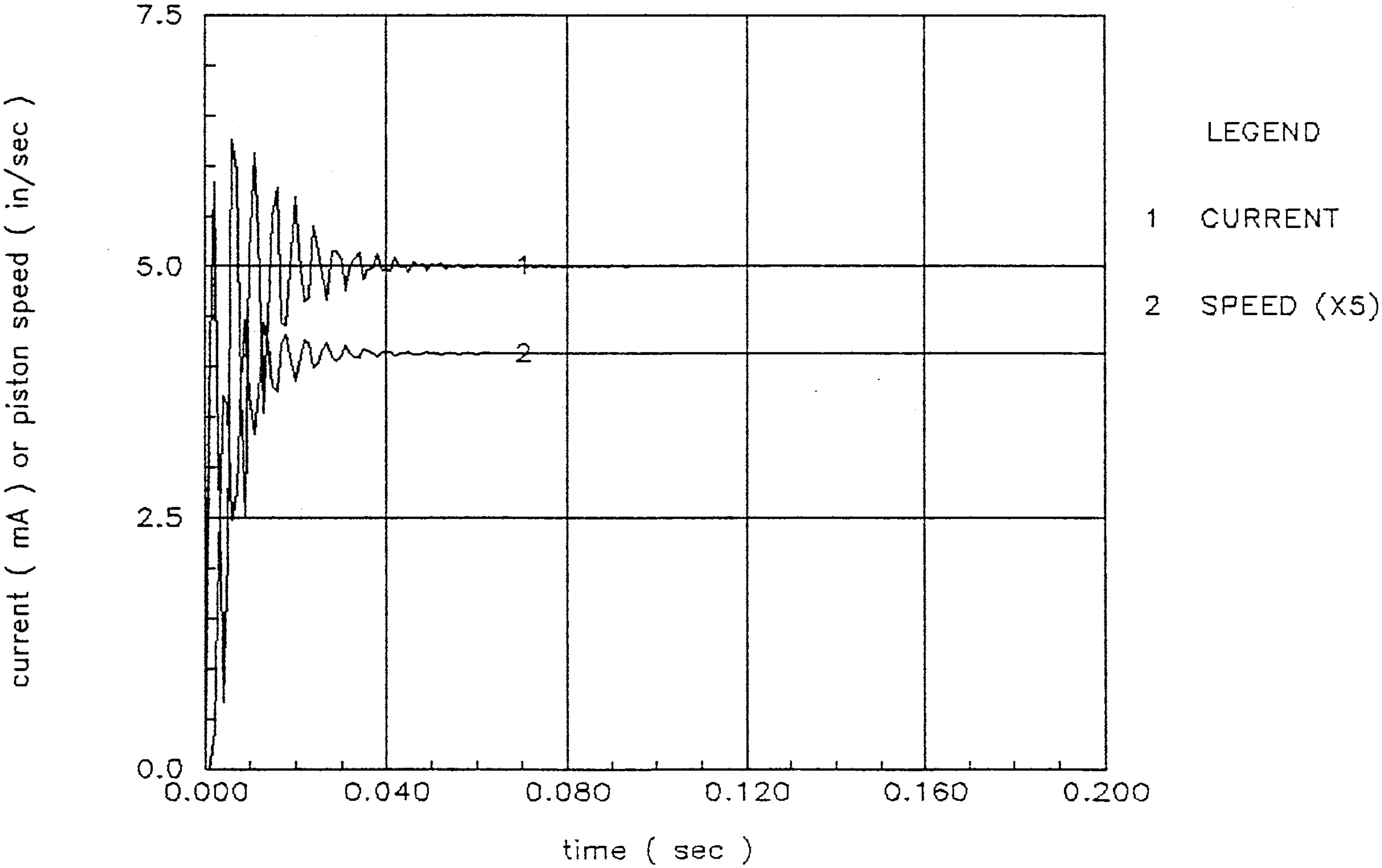

Figure 71. Dynamic Response - PID Feedback Control (L = 10 inch)

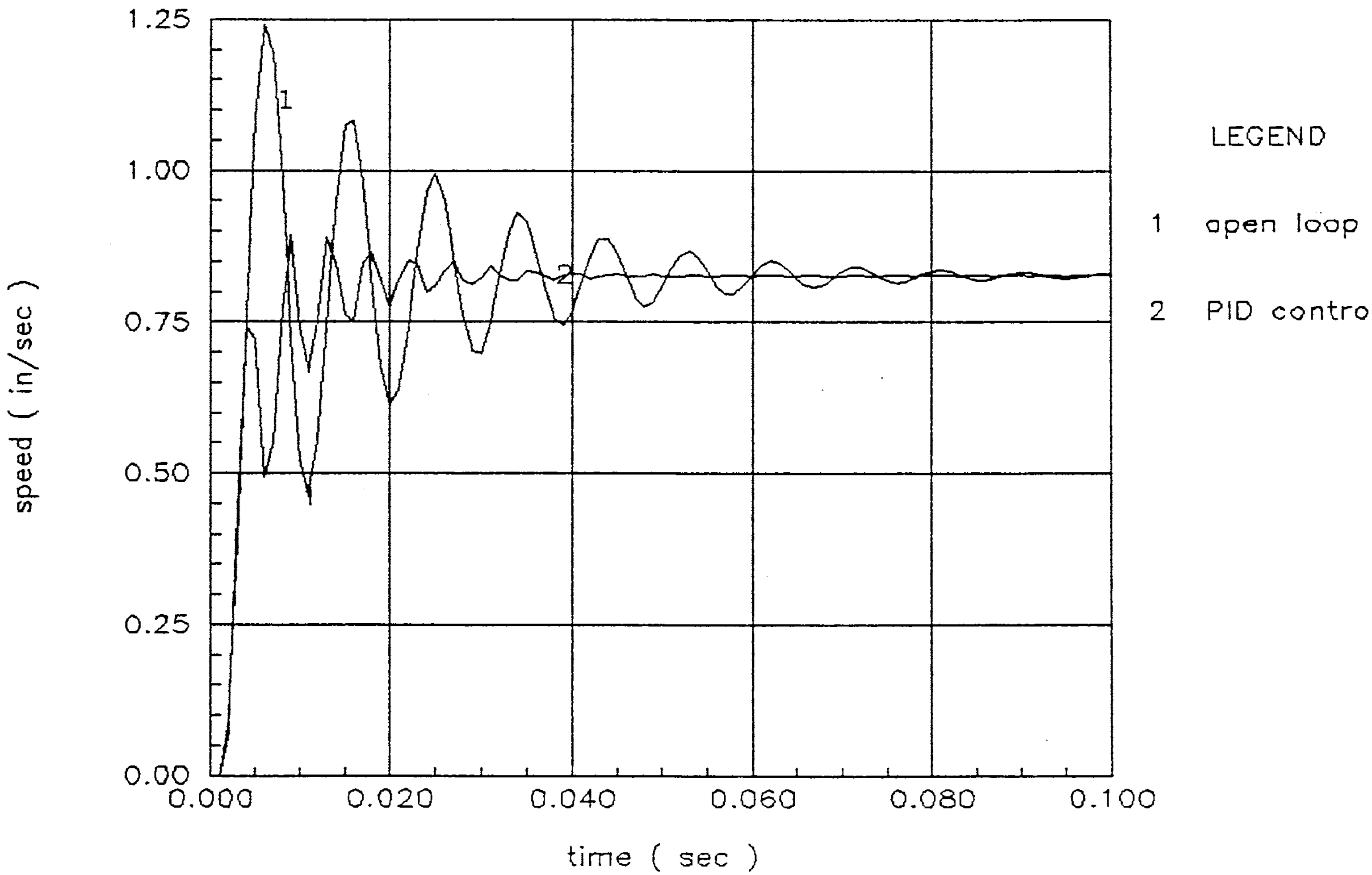

Figure 72. Velocity Control - Open Loop VS PID

$$\psi_2 = \begin{cases} 0.005, & \text{if } \sigma X_2 > 0 \\ -0.001, & \text{if } \sigma X_2 < 0 \end{cases} \qquad (120)$$

Figure 73 shows the response of VSS feedback control using the preceding sliding regime and control scheme (see Appendix N). A ramp signal was used for the reference signal. Chattering happened at the steady state condition. Figure 74 shows that VSS control can improve the transient response. And the comparision of PID and VSS to the open loop response is shown on Figure 75.

Position Control

PID feedback control

The controller gains are obtained as shown in Appendix O. The reference position is 1 inch. The gains obtained are:

$K_p = 6.25$

$K_i = 4.46$

$K_d = 1.64$

Figure 76 illustrates the displacement of the piston rod using PID feedback control.

VSS Control

The sliding regime and the control scheme are established in the following manner:

$$\sigma = 495630.6\ X_1 + 60\ X_2 + X_3 \qquad (121)$$

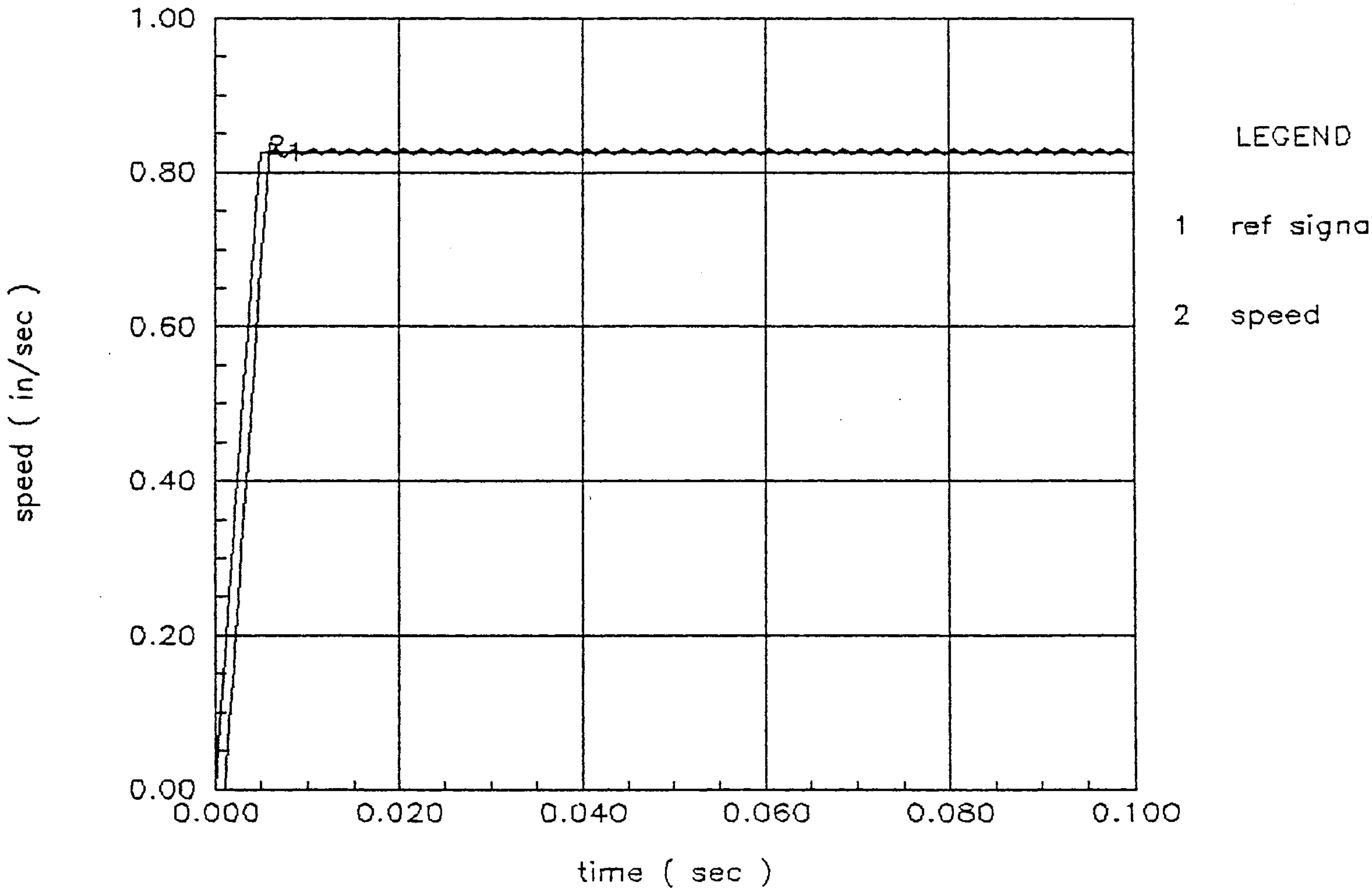

Figure 73. VSS Velocity Control

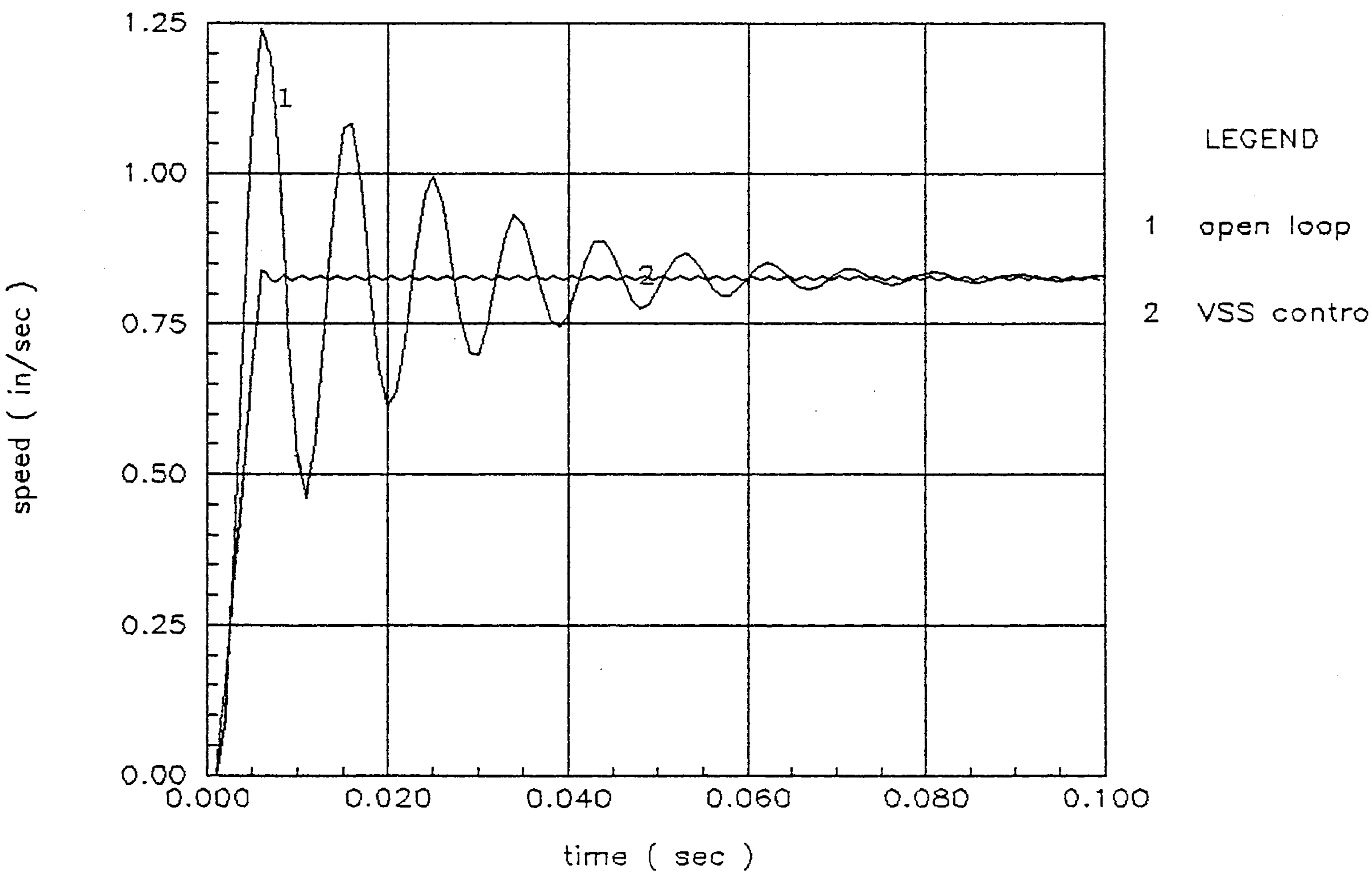

Figure 74. Velocity Control - Open Loop VS VSS

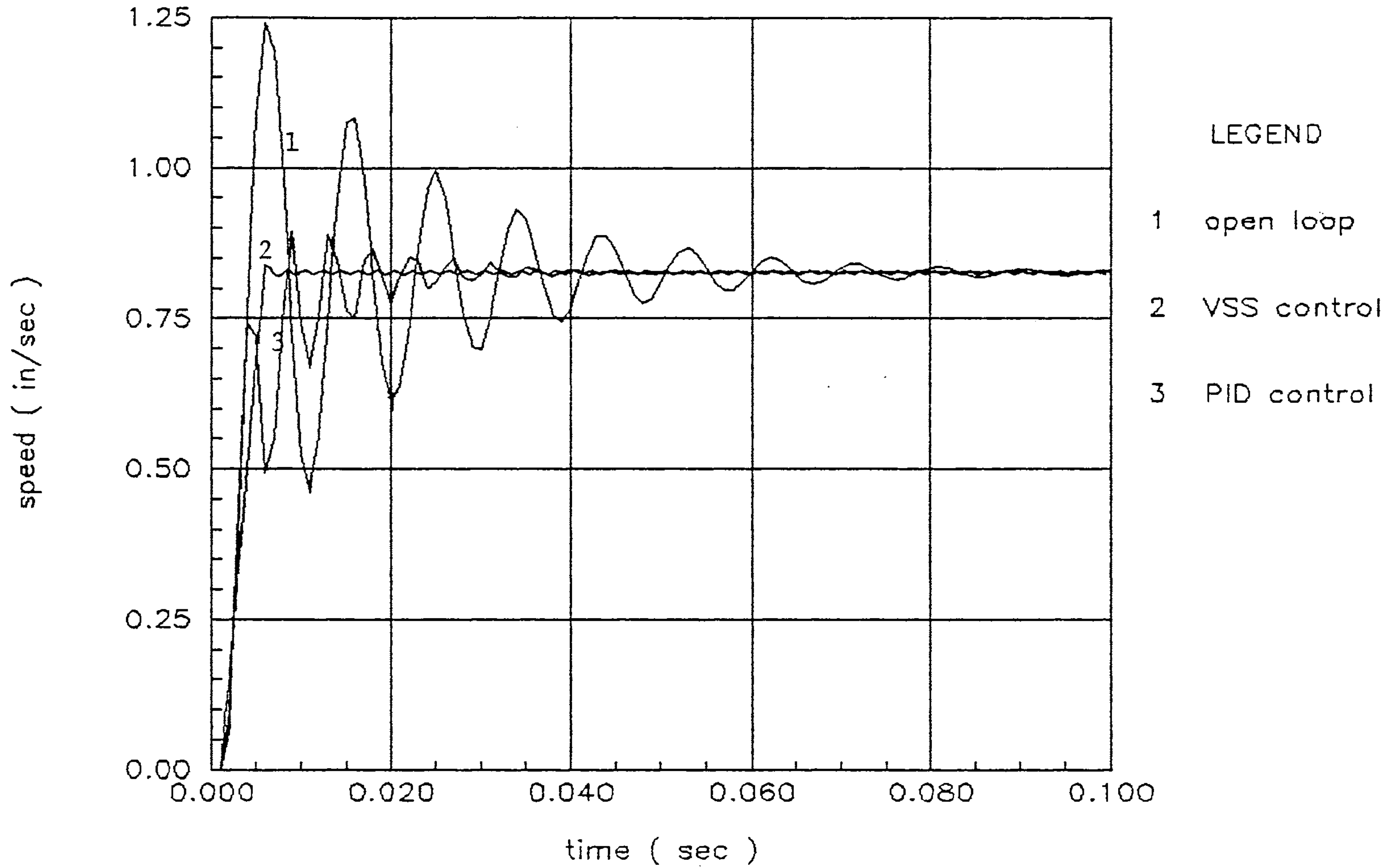

Figure 75. Velocity Control - Open Loop VS PID VS VSS

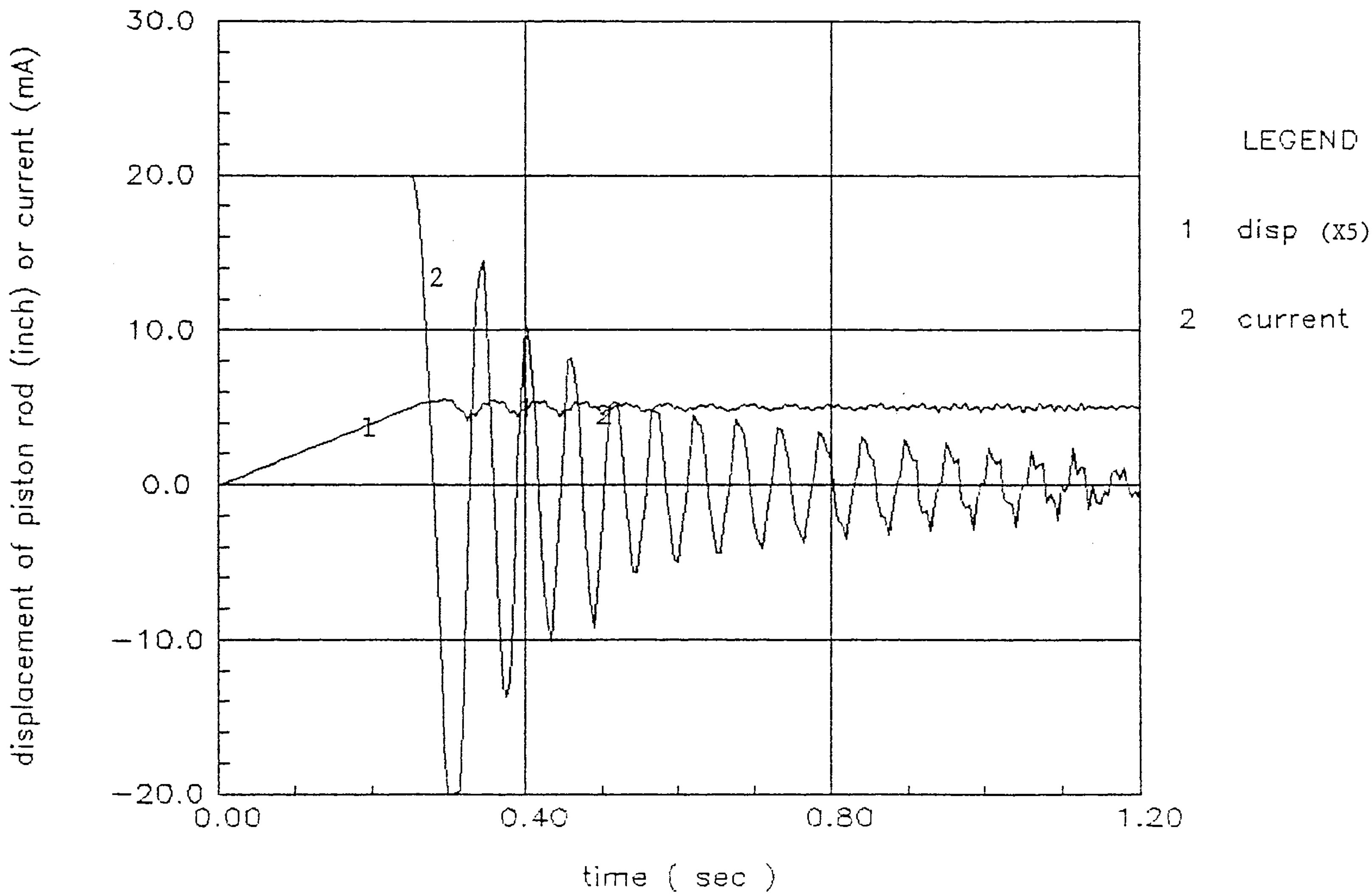

Figure 76. PID Position Control

$$u = \psi_1 X_1 + \psi_2 X_2 \tag{122}$$

where,

$$\psi_1 = \begin{cases} 10, & \text{if } \sigma X_1 > 0 \\ -800, & \text{if } \sigma X_1 < 0 \end{cases} \tag{123}$$

$$\psi_2 = \begin{cases} 40, & \text{if } \sigma X_2 > 0 \\ -4, & \text{if } \sigma X_2 < 0 \end{cases} \tag{124}$$

Figure 77 shows the displacement of the piston rod using preceding VSS control. The Variations of the error X_1 and derivatives X_2, X_3 are shown in Figures 78 and 79. Figure 80 shows the comparision of PID and VSS control. See Appendix P.

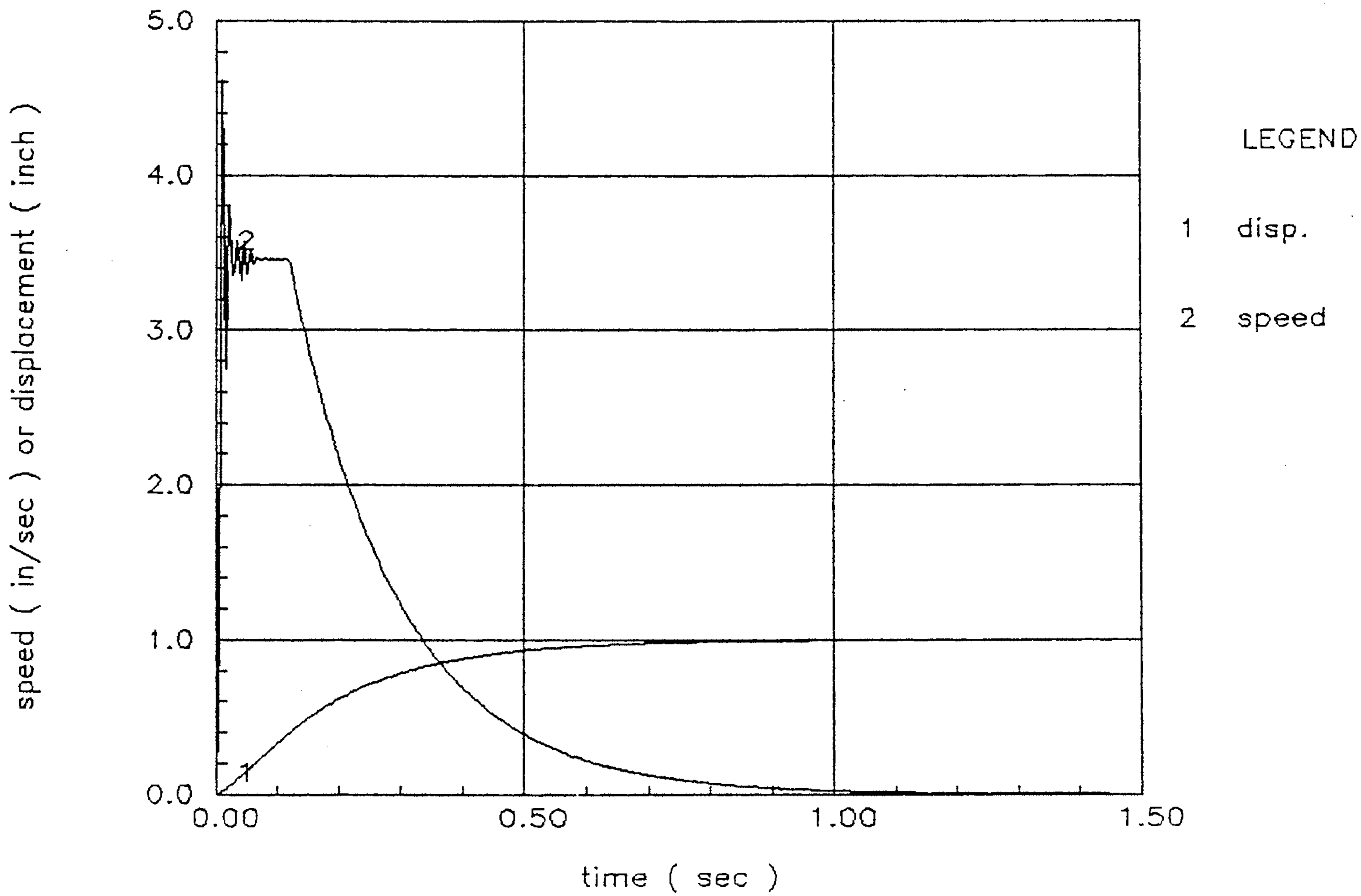

Figure 77. VSS Position Control

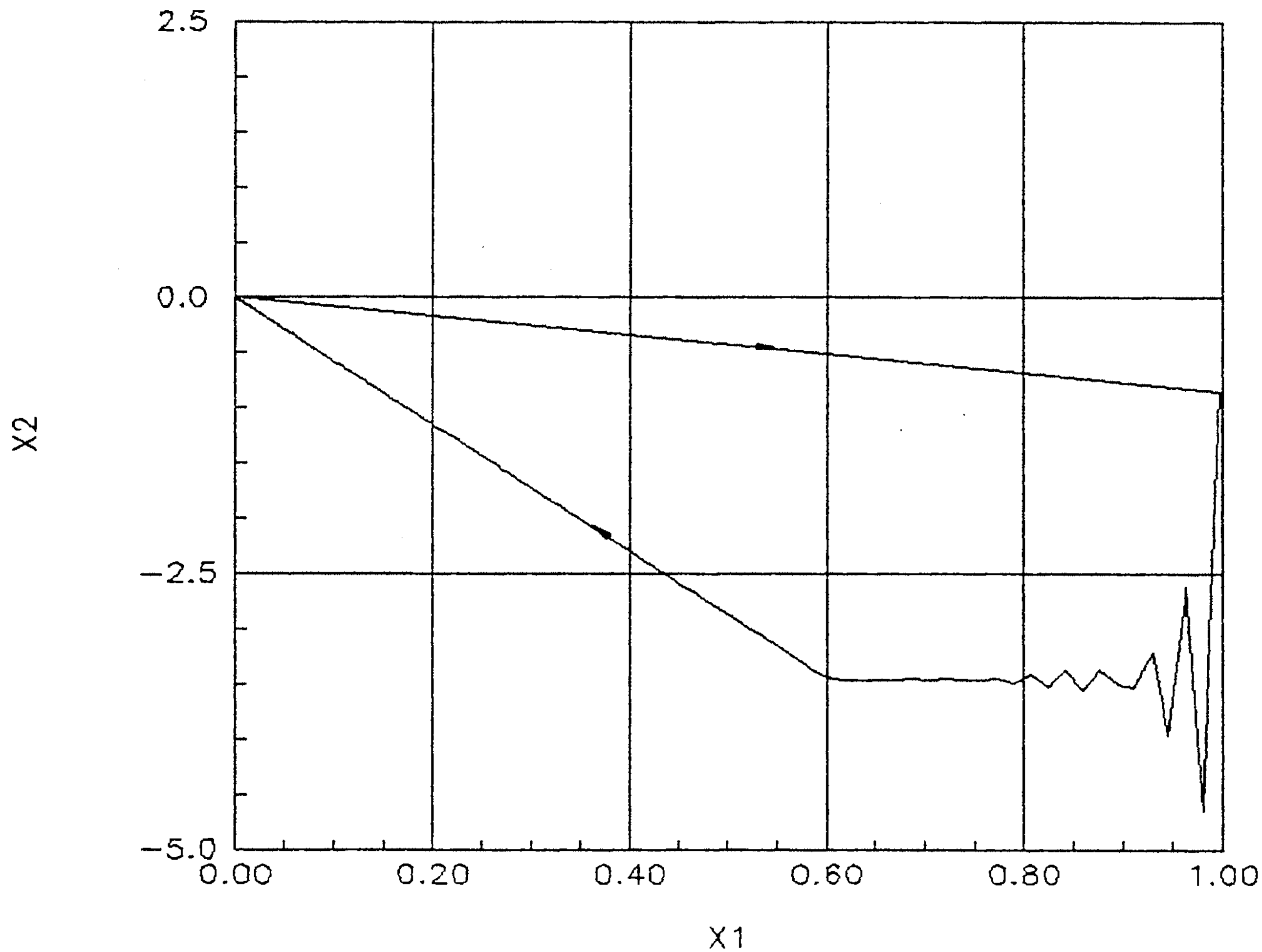

Figure 78. VSS Control - X1 VS X2

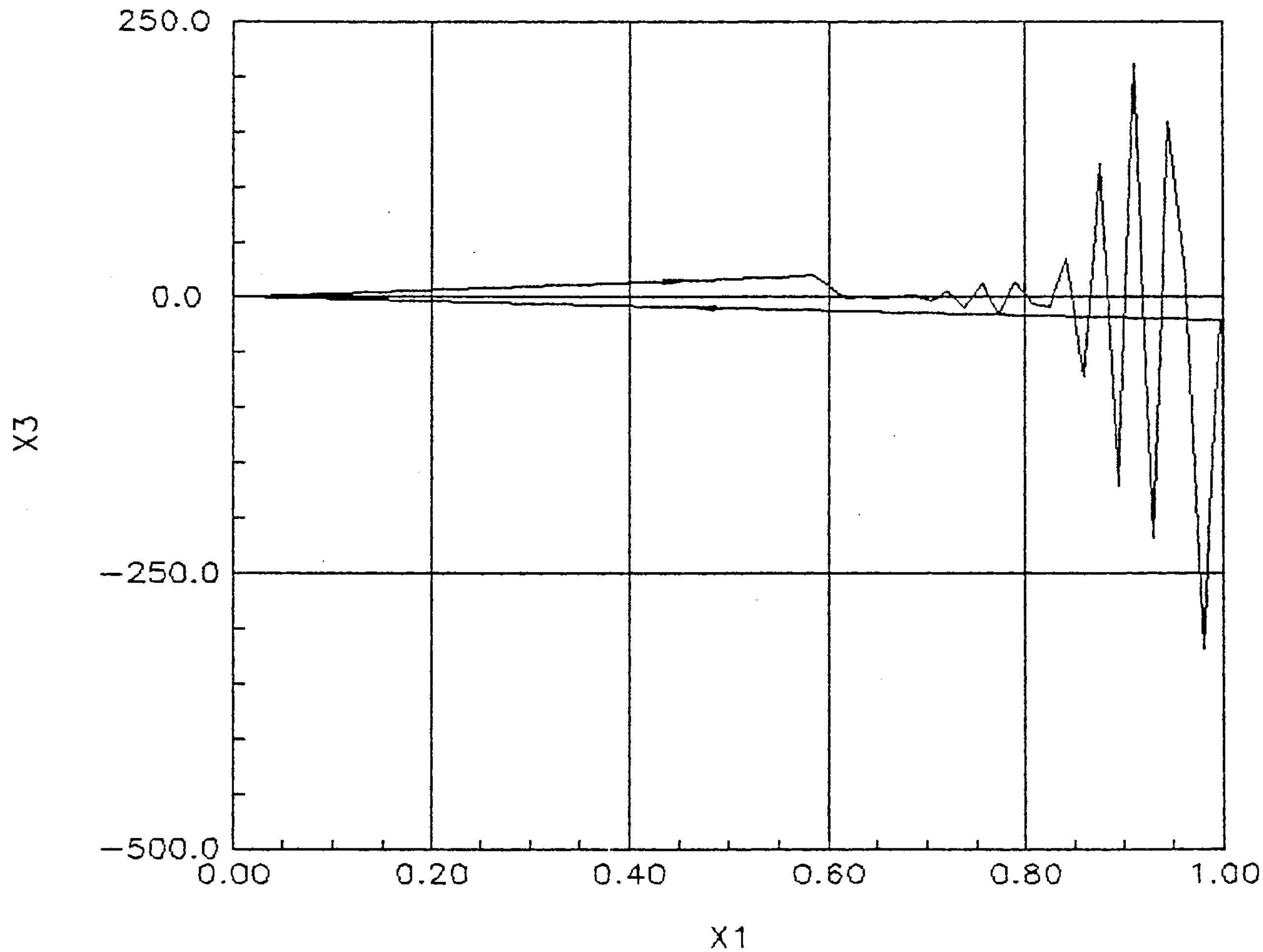

Figure 79. VSS Control - X1 VS X3

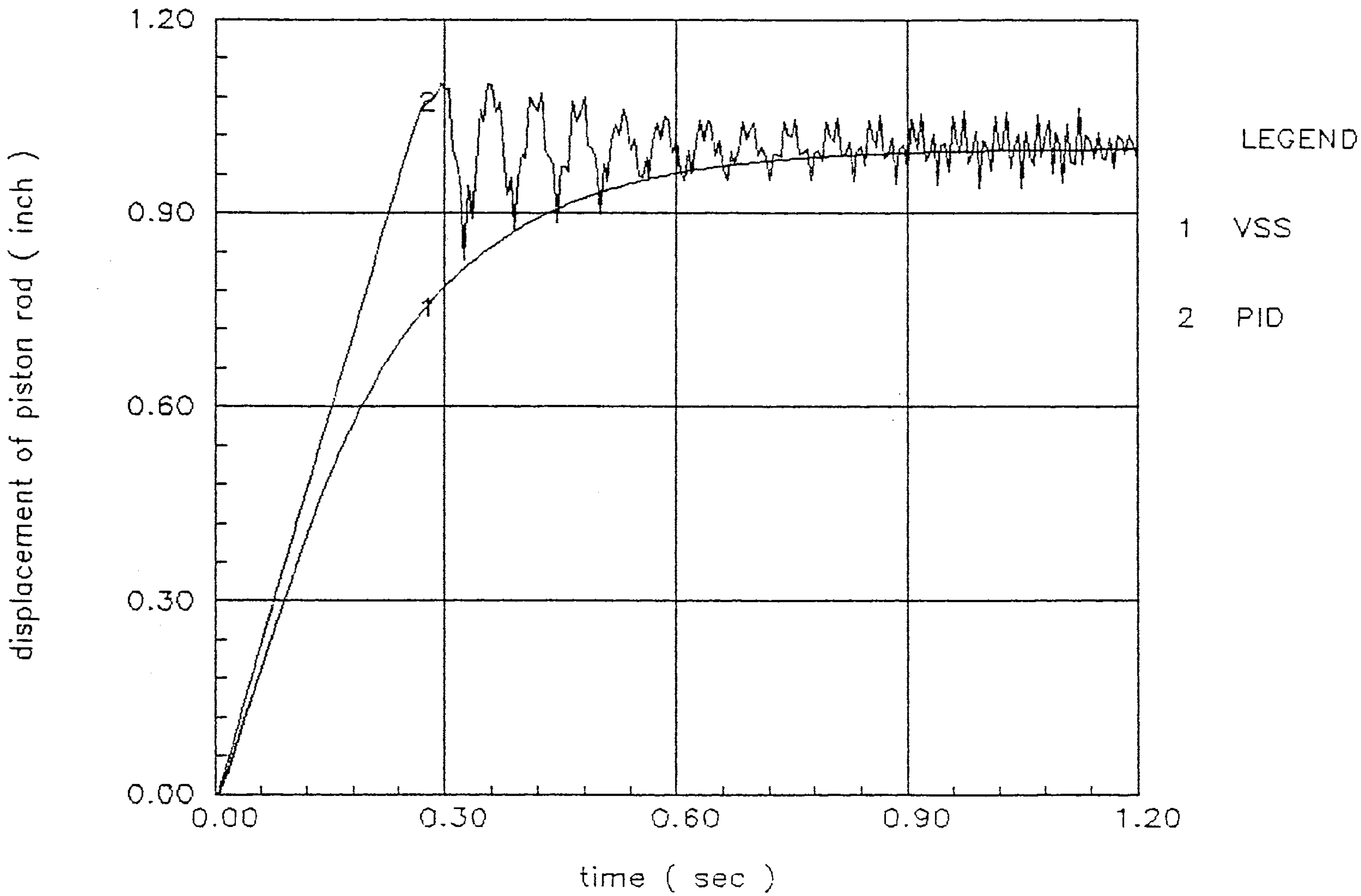

Figure 80. Comparision of PID and VSS

CHAPTER V

EXPERIMENT

Experimental Scheme and Preparation

Experimental Apparatus

The schematic diagram for this experiment is shown in Figure 81. Illustrations view of the experimental equipment and the data control instruments are shown in Figures 82 and 83. The electrical circuit diagram for the experiment is shown in Figure 84.

A servovalve controlled symmetric cylinder system was prepared. The displacement of the piston rod was measured using a Linear Voltage Transducer. A resistor-capacitor circuit was installed at the output line of the Linear Voltage Transducer in order to measure the velocity of the piston rod. In order to increase the momentum of inertia, an external load was connected to the piston rod. The supply pressure to the servovalve was maintained at 500 psig using a relief valve and an adjustable throttle valve in the by-pass line. The oil temperature was maintained at 120 °F. Two electrical power supplies were used to generate the ± 12 volts d.c. required in the system. A digital data control instrument coupled with a personal digital computer

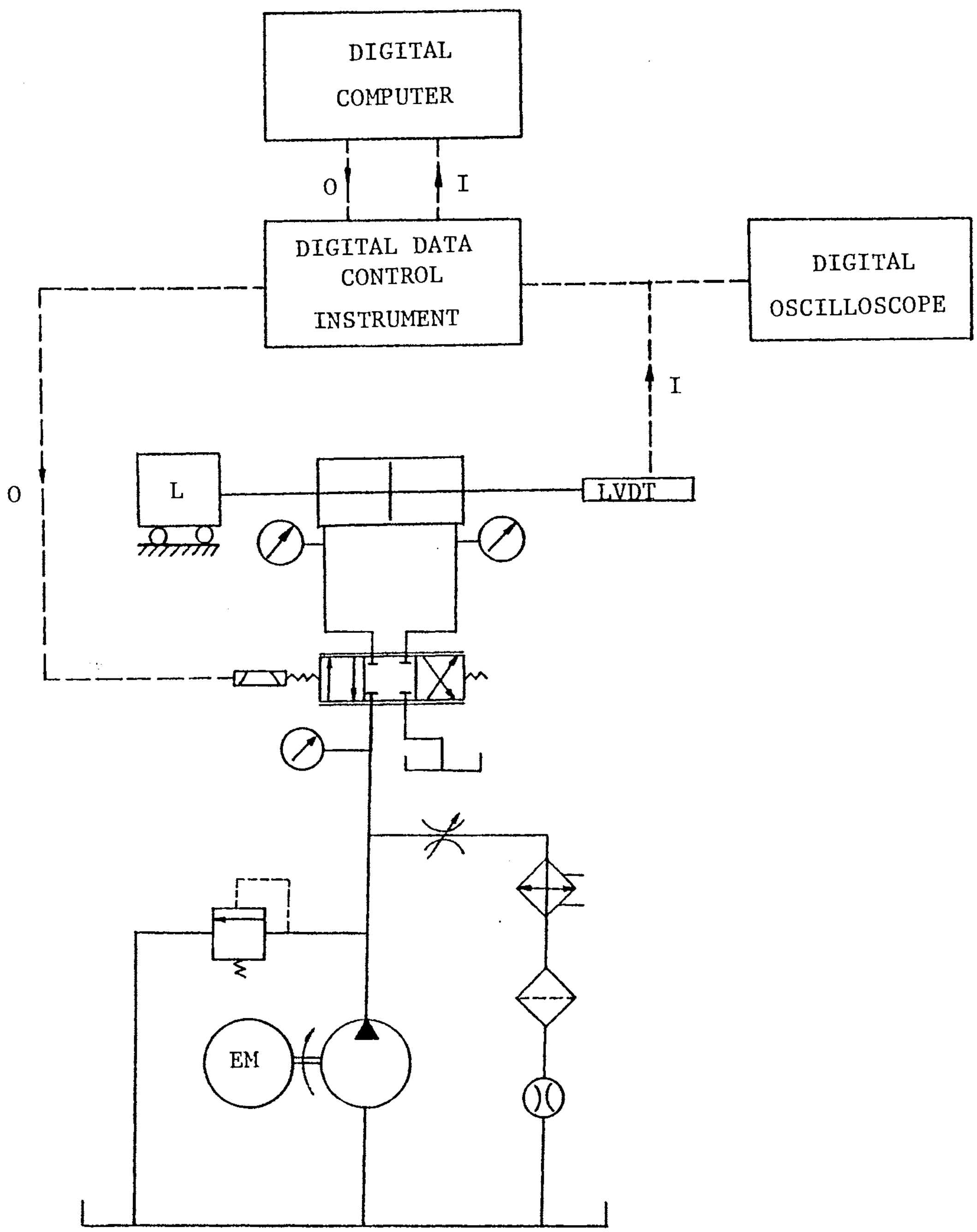

Figure 81. Schematic Diagram for Experimental Apparatuses

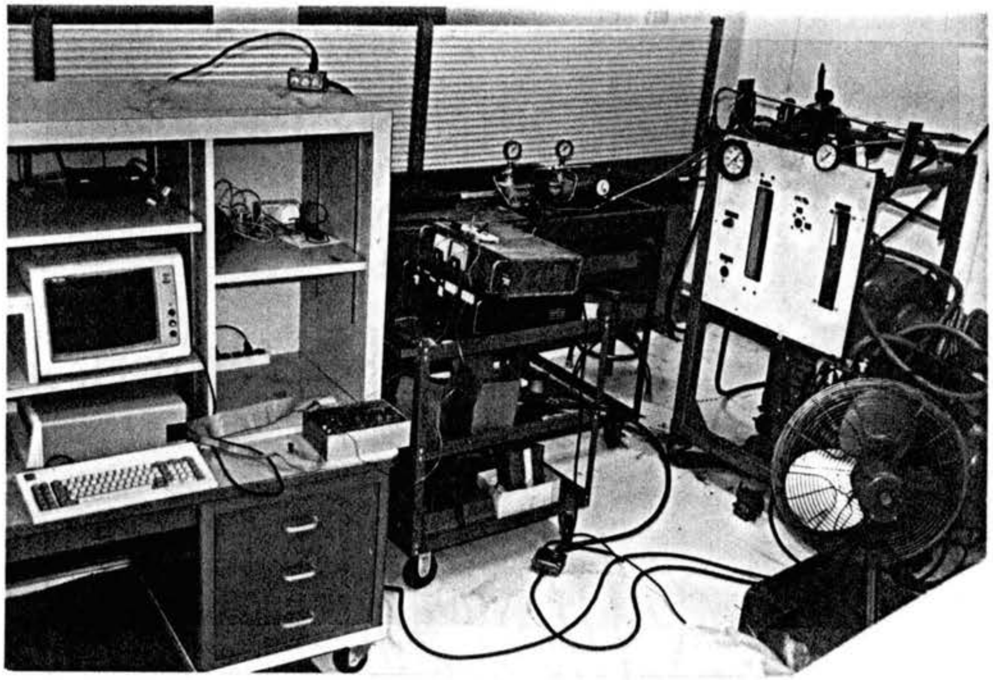

Figure 82. Overall View of Experimental Apparatus

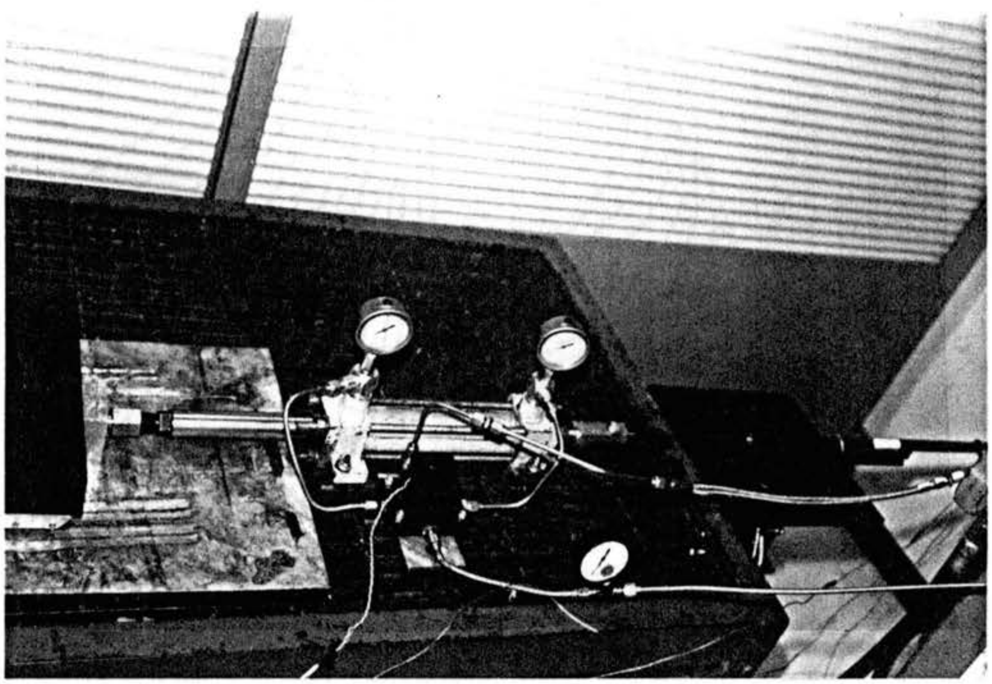

Figure 83. View of A Servovalve - Cylinder System

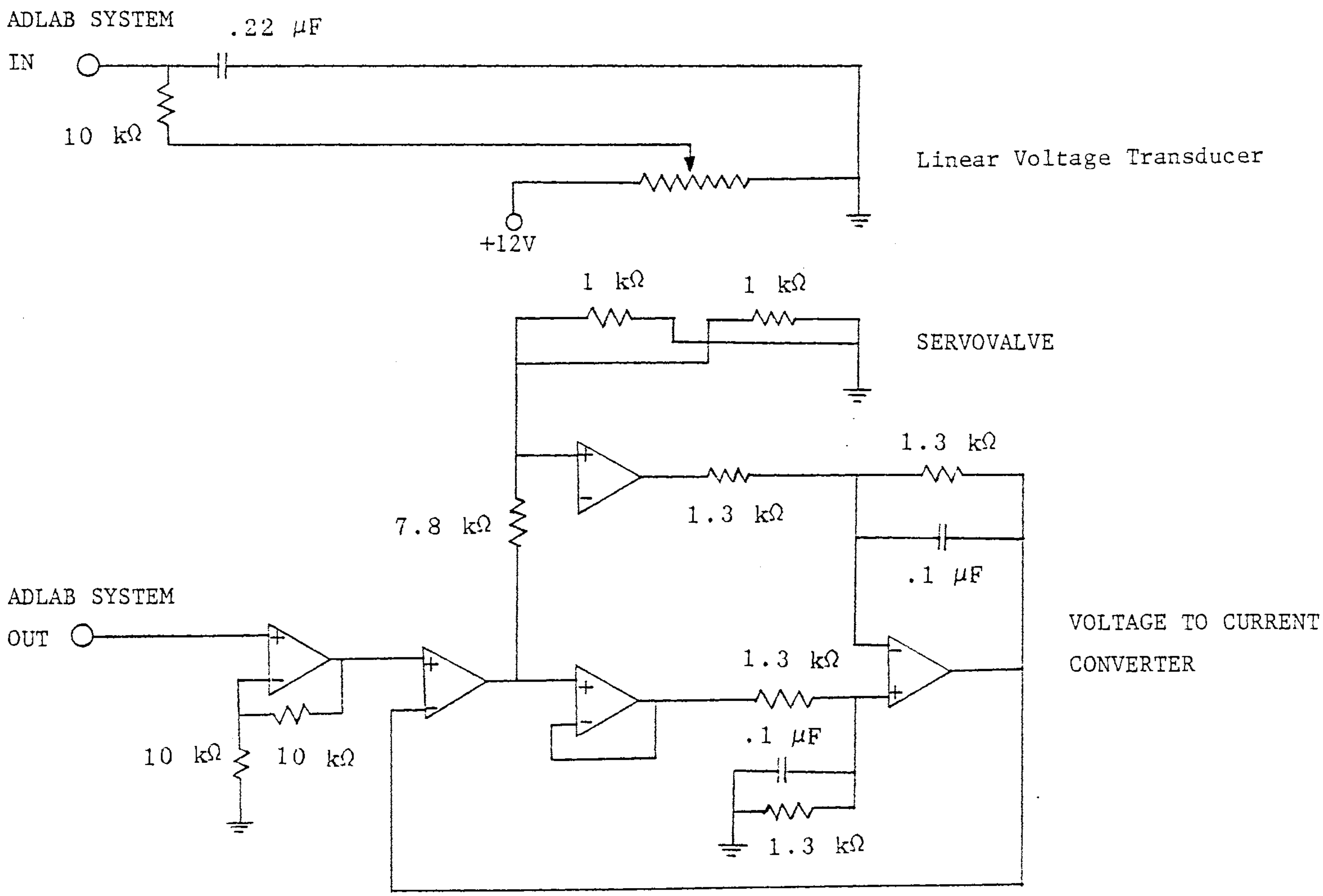

Figure 84. Electrical Circuit Diagram For Experiment

was used for acquiring the data and controlling the system. The sampling rate of this instrument was 16000 sample/sec (0.0625 msec/sample). A digital oscilloscope was also used for data acquisition. The main features of the experimental apparatuses are shown in Table VI.

TABLE VI

EXPERIMENTAL APPARATUSES

Equipments	Product Company	Ser. No.	Model No.
Servo Valve	Dynamic Valve Inc.	1459	10
Cylinder	Sheffer Co.	356920	2HHSL - 7ADKY
Linear Volt. Transducer	Bourns		357D
Digital Data Control Instrument	Interactive Microwave Inc.		ADLAB - PC™
Power Supply	Autonetics	TR 36 - 4M	A2004
	Sorensen		DCR 20 - 50B
Digital Oscilloscope	Nicholet		NIC-310
Personal Computer	IBM		IBM PC

Calibration and Identification

First, the Linear Voltage Transducer was calibrated. Then other characteristic parameters of the system were identified in the following manner.

Linear Voltage Transducer

Figure 85 reveals the result of the calibration of the Linear Voltage Transducer connecting to the piston rod. A positive 12 Volts d.c. were supplied for bias voltage. The linear relationship between the output voltage and the displacement of piston rod is:

$$\text{voltage (V)} = 0.5413 \text{ displacement (inch)} + 5.1651 \quad (125)$$

Oil Density

Hydraulic oil MIC138-CJ was used. The density was measured as oil temperature changed. Figure 86 reveals that the linear relationship between the oil density and the temperature is:

$$\text{density (lbm/ft}^3\text{)} = -0.0219 \text{ temperature (}^\circ\text{F)} + 55.5464 \quad (126)$$

Oil Viscosity

The viscosity was measured as oil temperature changed. As shown in Figure 87, the Walther equation was used to obtain the relationship between the viscosity and the

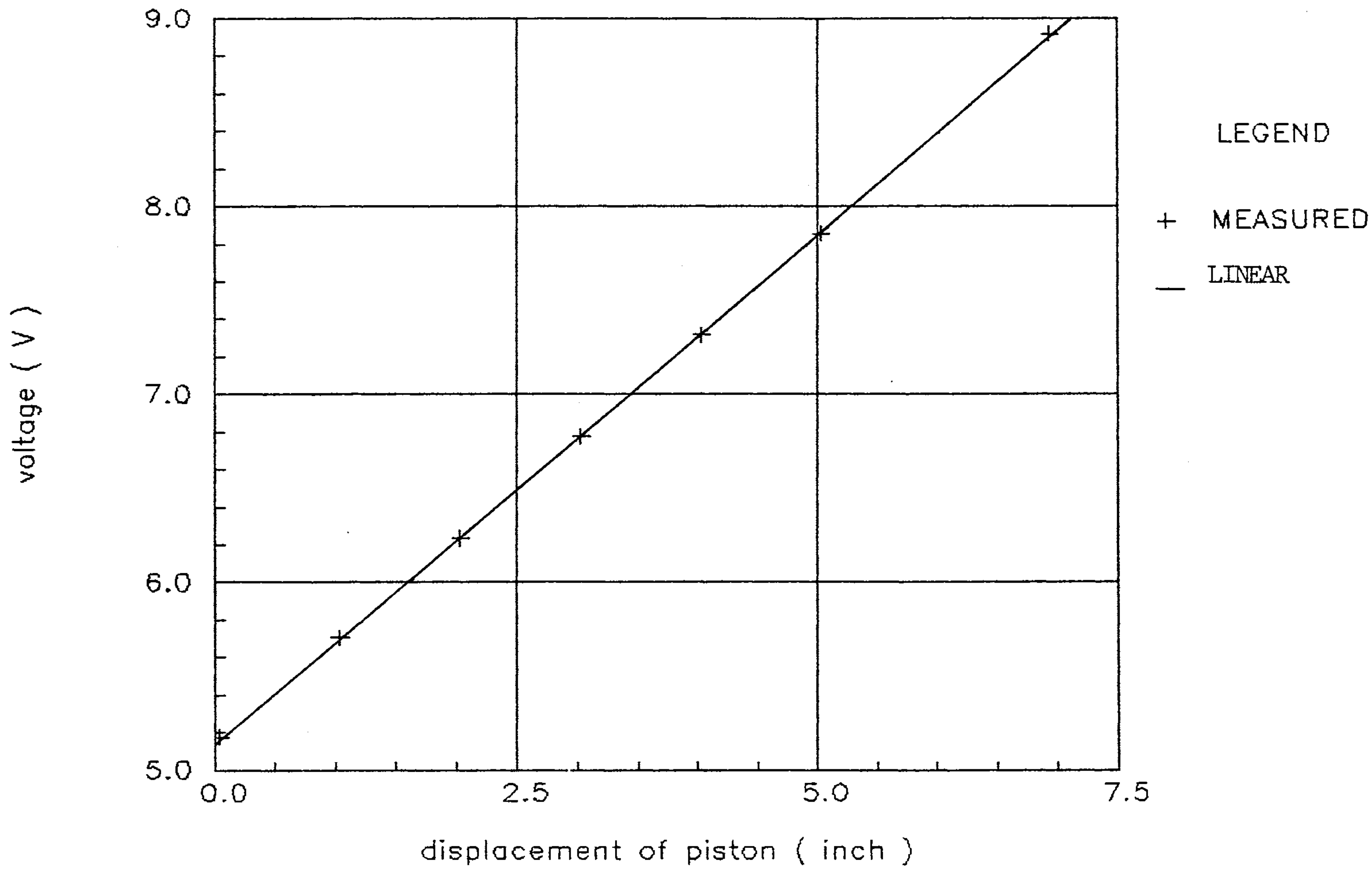

Figure 85. Linear Voltage Transducer Calibration

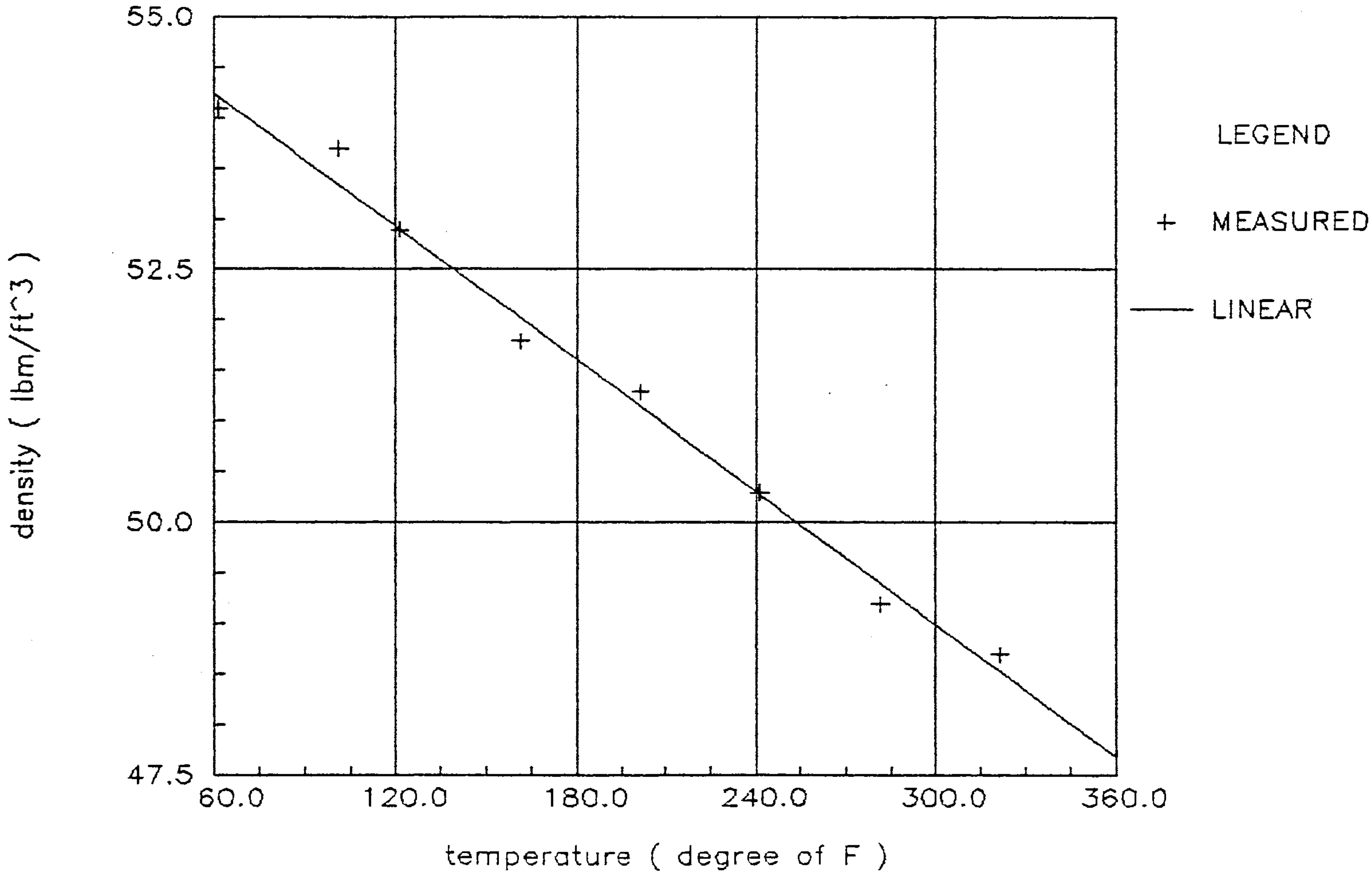

Figure 86. Variation of Density

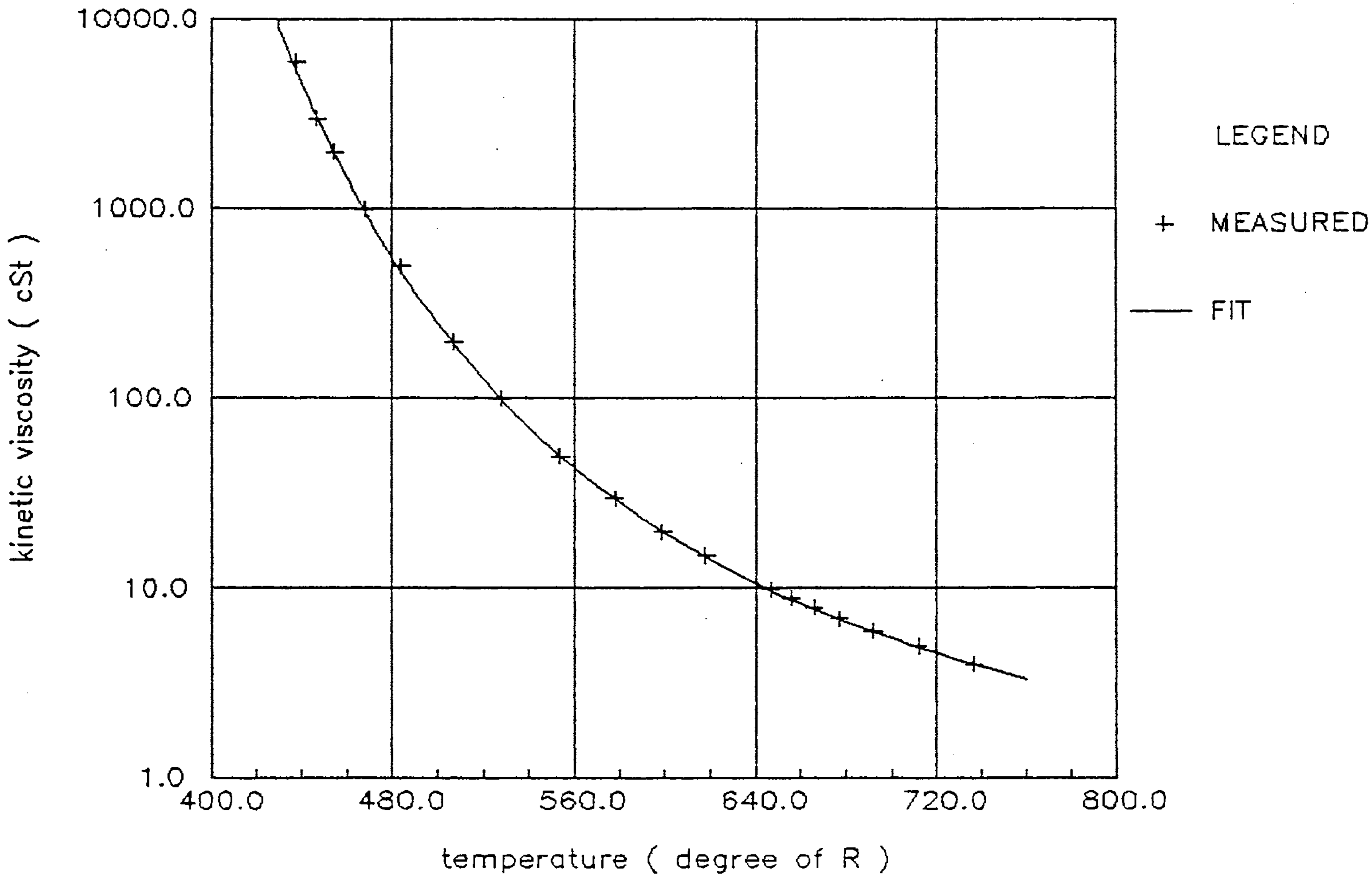

Figure 87. Variation of Viscosity

temperature.

$$\log_{10}(\log_{10}(\text{viscosity (cSt)} + 0.6)) = -3.3328 \log(\text{temperature}^{\circ}(\text{ R})) + 9.3743 \quad (127)$$

Servovalve

Figure 88 reveals alterations in the flow rate of the servovalve as the servovalve current alternates on a cycle of rate current: + 20 mA to -20 mA to + 20 mA. The hysterisis of this valve is revealed. It appears to have linear flow gain within ± 30 % of rate current. The connection of the servovalve and the cylinder is shown in Figure 89. Therefore, negative current created positive displacement of the piston rod in the tests.

The features of the hydraulic components were shown in Table VII.

Experimental Procedure

The procedure for this experiment is:

(1) Turn on the electrical and hydraulic power supply. Adjust the voltage of one electrical power supply to + 12 volts d.c. and the other to be - 12 Volts d.c. Adjust the throttle valve for the supply pressure to the servovalve to 500 psig. Maintain the oil temperature 120 °F.

(2) Move the piston rod at full retraction position.

(3) Execute the control program using a personal computer.

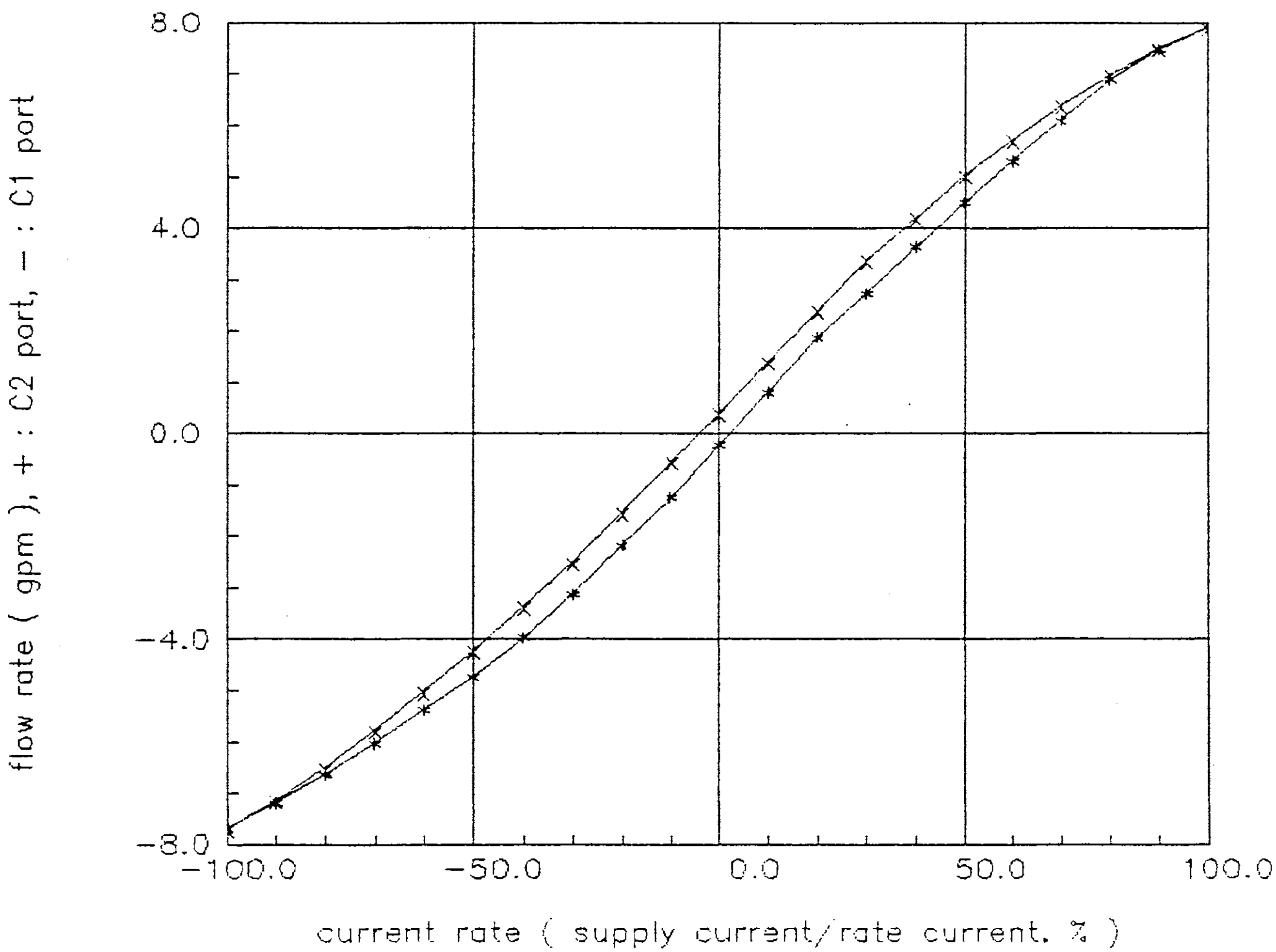

Figure 88. Flow Gain of the Servovalve

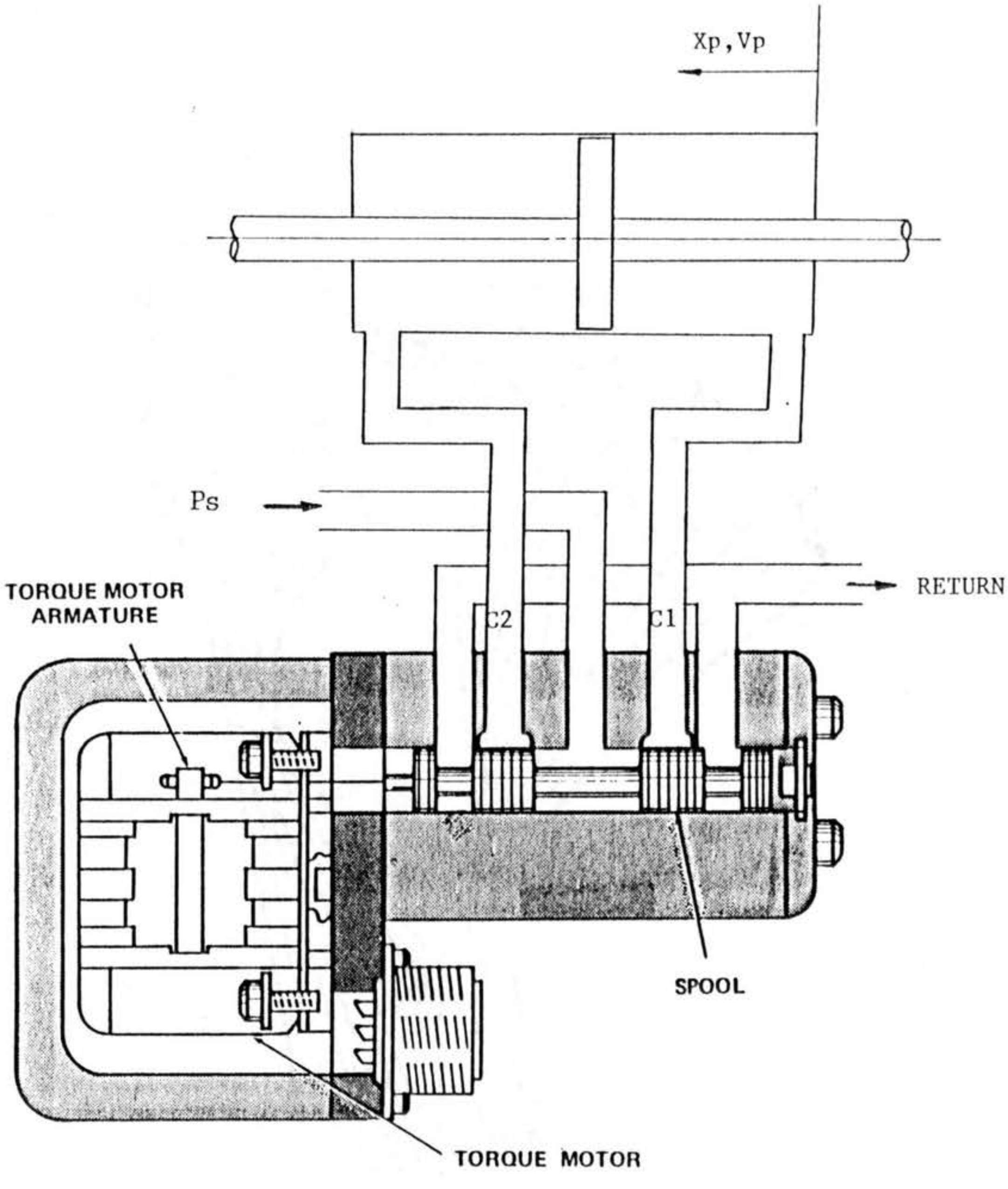

Figure 89. Connection of Servovalve and Cylinder

TABLE VII

EXPERIMENTAL DATA

Parameters	symbol	value	unit
Bulk modulus	β_e	1.8×10^5	psig
Density of oil	ρ	0.793×10^{-4}	$lb\text{-}sec^2/in^4$
Absolute viscosity	μ	3.47×10^{-6}	$lb\text{-}sec/in^2$
Supply pressure	P_s	500	psig
Total weight of piston and load	M_w	400	lb_f
Pressure Difference at 2.5 mA	ΔP	30	psid
Tube Inner Dia.	D_i	5/32	inch
Length	L_t	10	inch (short)
		100	inch (long)
Piston Diameter	D_r	1.378	inch
Cylinder Inner Diameter	D_c	1.976	inch
Piston Stroke	L_r	6.9	inch
Valve Flow Gain	K_{qi}	0.0689	in^3/sec / $\sqrt{psi}$ mA

(See Appendices Q and R for the computer program.)

(4) Obtain the data using the digital data control instrument and the digital oscilloscope.

(5) Change the values (such like the valve current and the controller gains). Repeat procedure (2) through (4).

(6) Change the line. Repeat procedure (2) through (5).

Dynamic Response and System Identification

Dynamic Velocity Responses

The velocity responses of the piston rod at 2.5 mA were recorded when the line length are 10 inch and 100 inch (see Figures 90 and 91). The displacement signal from Linear Voltage Transducer was differentiated to obtain velocity response. Some electrical noises were affeted through differentiation electrical circuit. The frequency of the noise was about 60 Hz regardless of line length. Therefore it is believed that the noise was resulted from the electrical power source. Response delay were obvious : 0.036 seconds for 10 inches line and 0.075 seconds for 100 inches line. Both responses were overdamped second order responses.

This systems were simultated numerically and the results were compared to the preceding experimental data (see Figures 92 and 93). The figures reveal that both responses, simulation and experiment, have approximately same damping ratios and natural frequencies. However, response delays in experiment were greater than those of

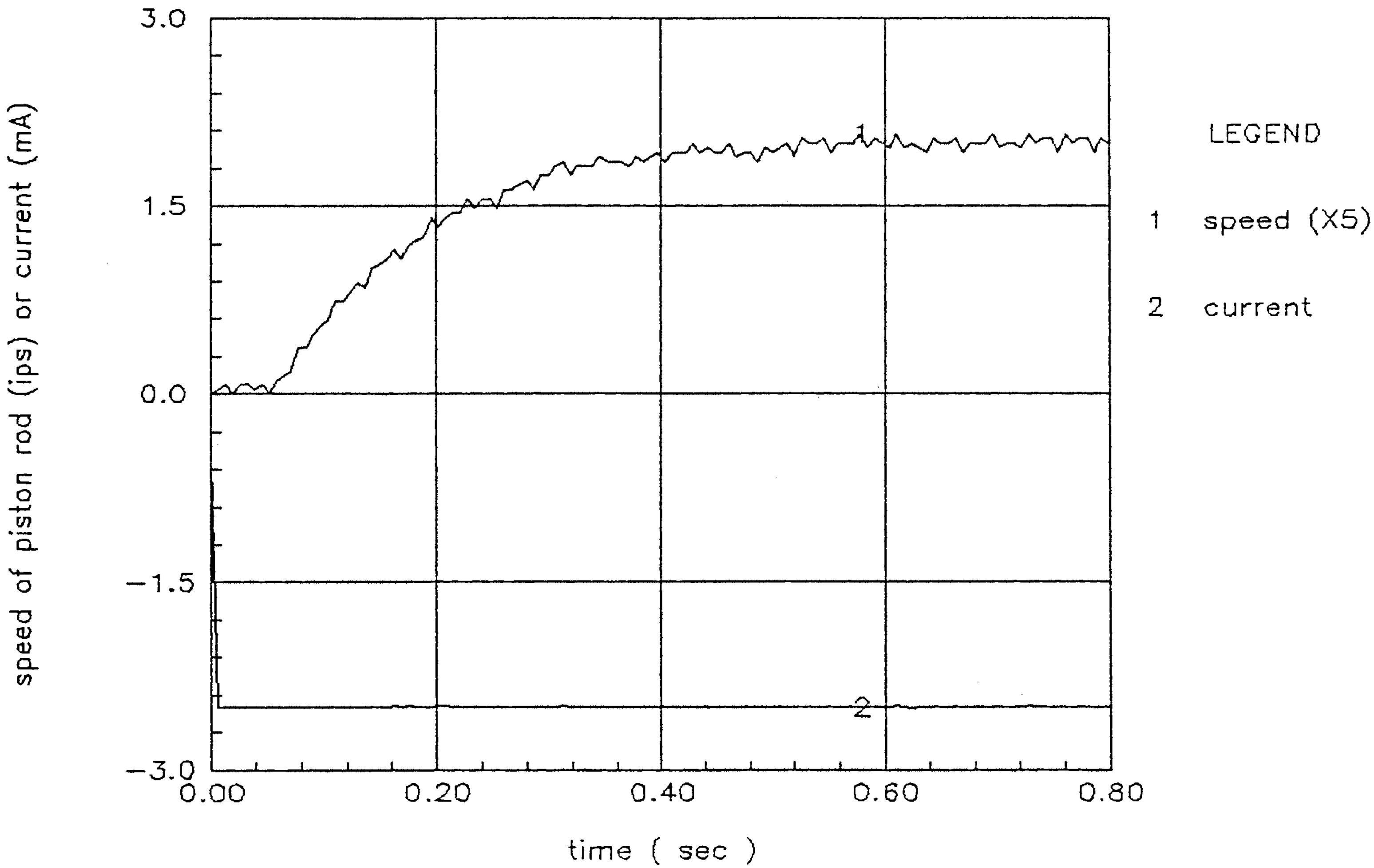

Figure 90. Dynamic Response (10 inch)

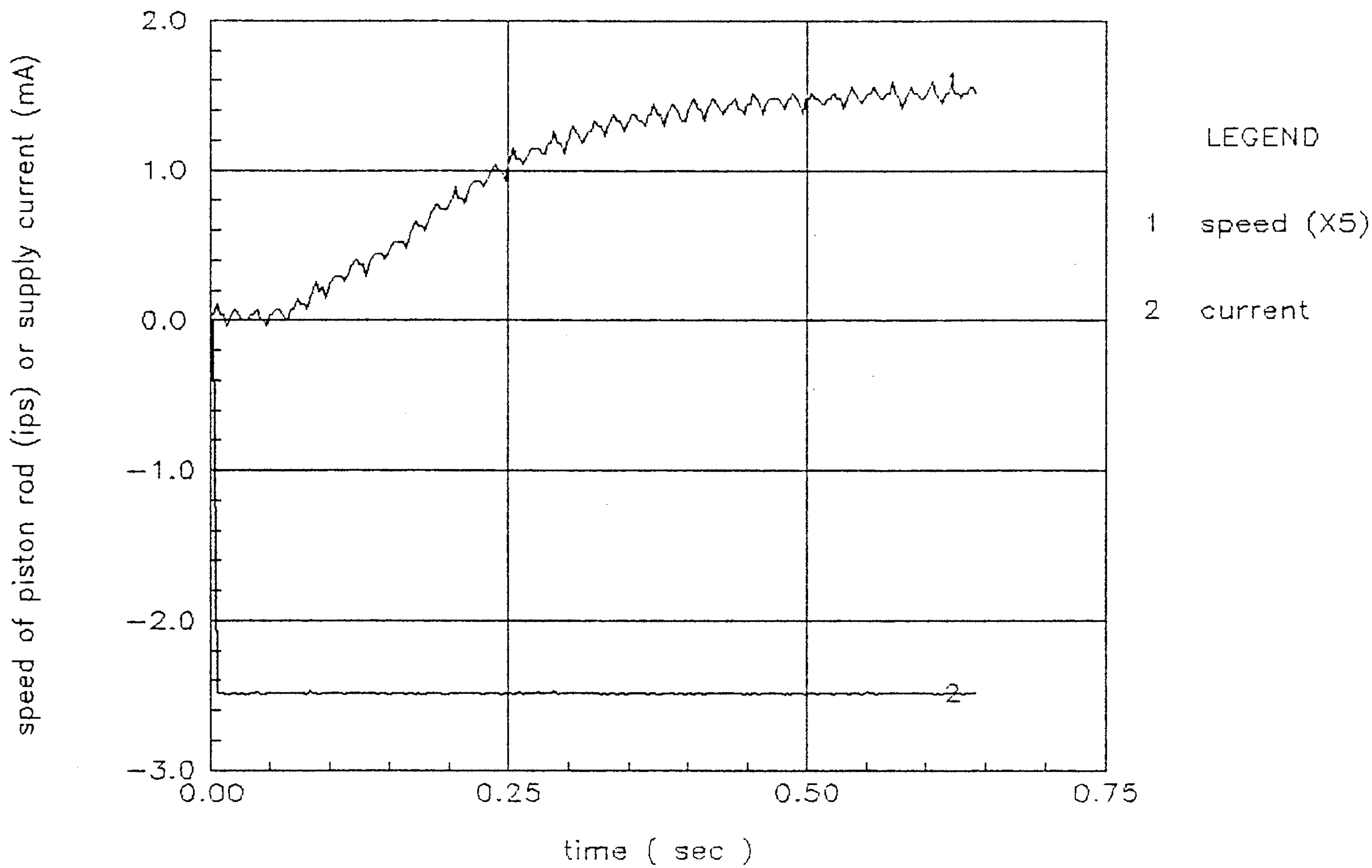

Figure 91. Dynamic Response (100 inch)

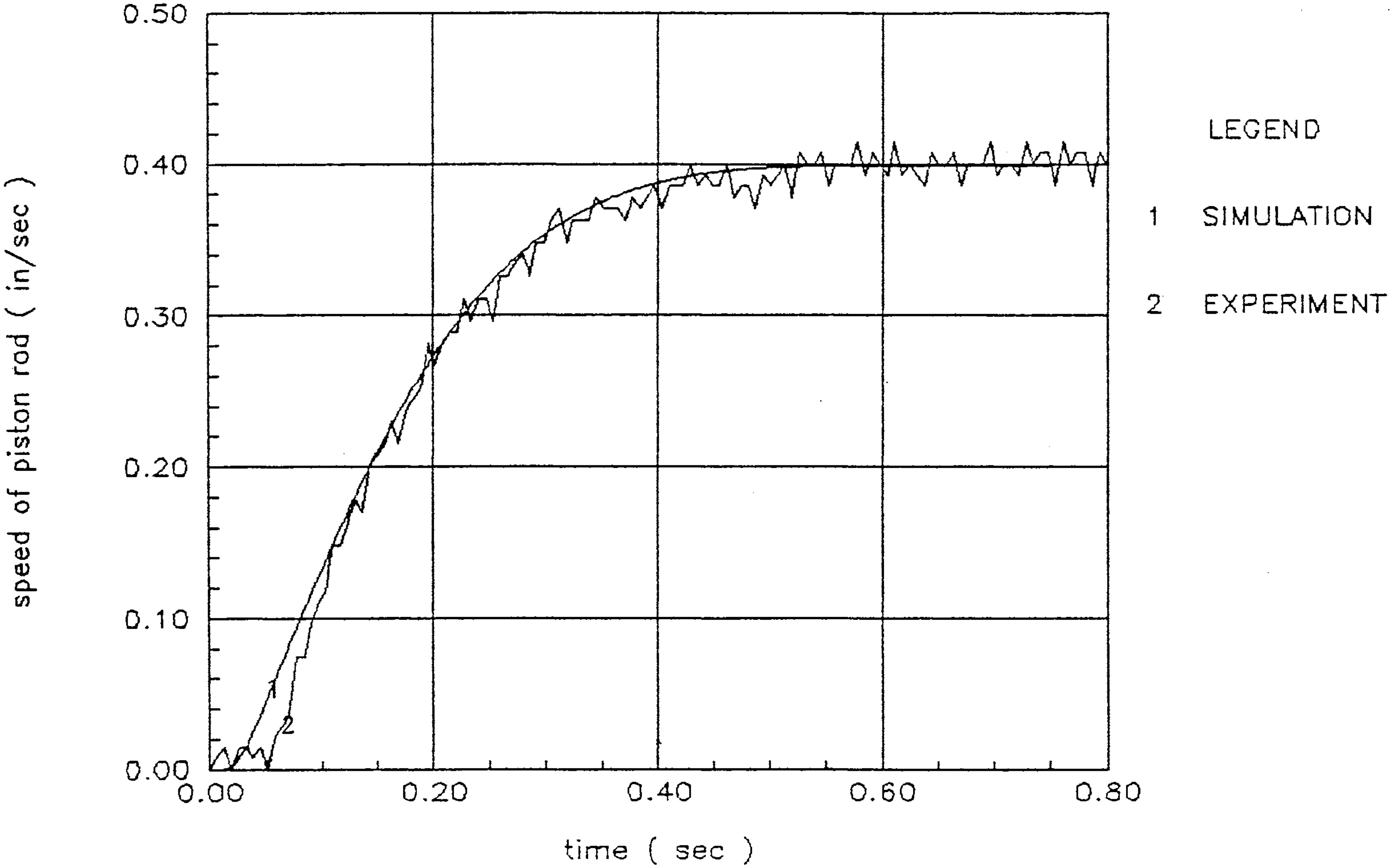

Figure 92. Comparision of Dynamic Responses (10 inch)

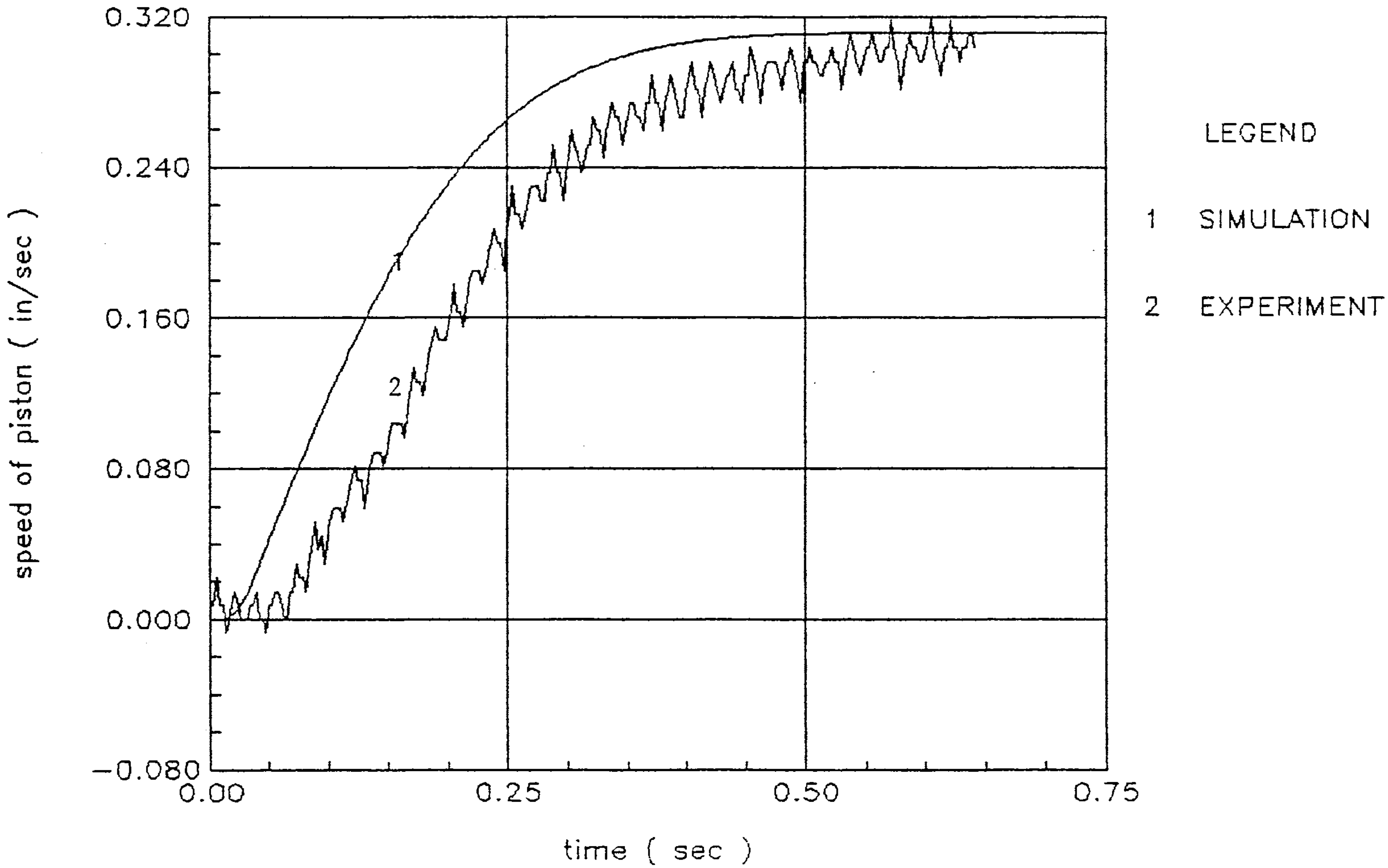

Figure 93. Comparision of Dynamic Responses (100 inch)

simulation. This is assumed to be resulted from the dynamic reaction time of the servovalve which could not be identified.

System Identification

The system parameters for the velocity were identified using second order transfer function. The results are shown in Figures 94 and 95. The transfer functions are:

For 10 inches line:

$$\frac{Vp}{I} = \frac{27}{s^2 + 90\ s + 625} \tag{128}$$

$$I = \begin{cases} 0 & \text{time} < 0.035 \text{ sec} \\ 2.5 \text{ mA} & \text{time} \geq 0.035 \text{ sec} \end{cases}$$

For 100 inches line:

$$\frac{Vp}{I} = \frac{13.28}{s^2 + 68\ s + 400} \tag{129}$$

$$I = \begin{cases} 0 & \text{time} < 0.07 \text{ sec} \\ 2.5 \text{ mA} & \text{time} \geq 0.07 \text{ sec} \end{cases}$$

Table VIII shows the damping constants and the natural frequencies of both systems.

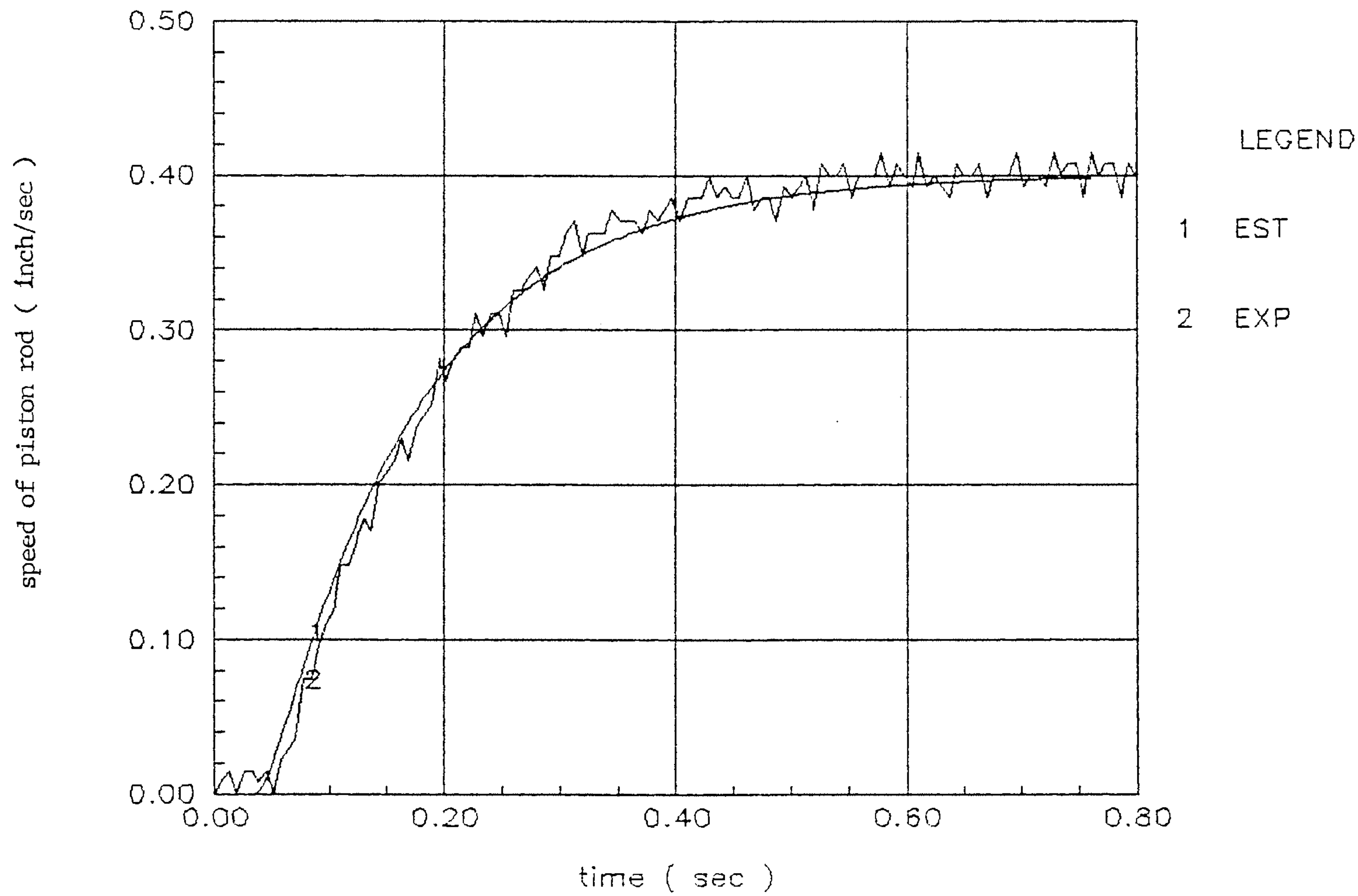

Figure 94. Identification of 10 inches Line System

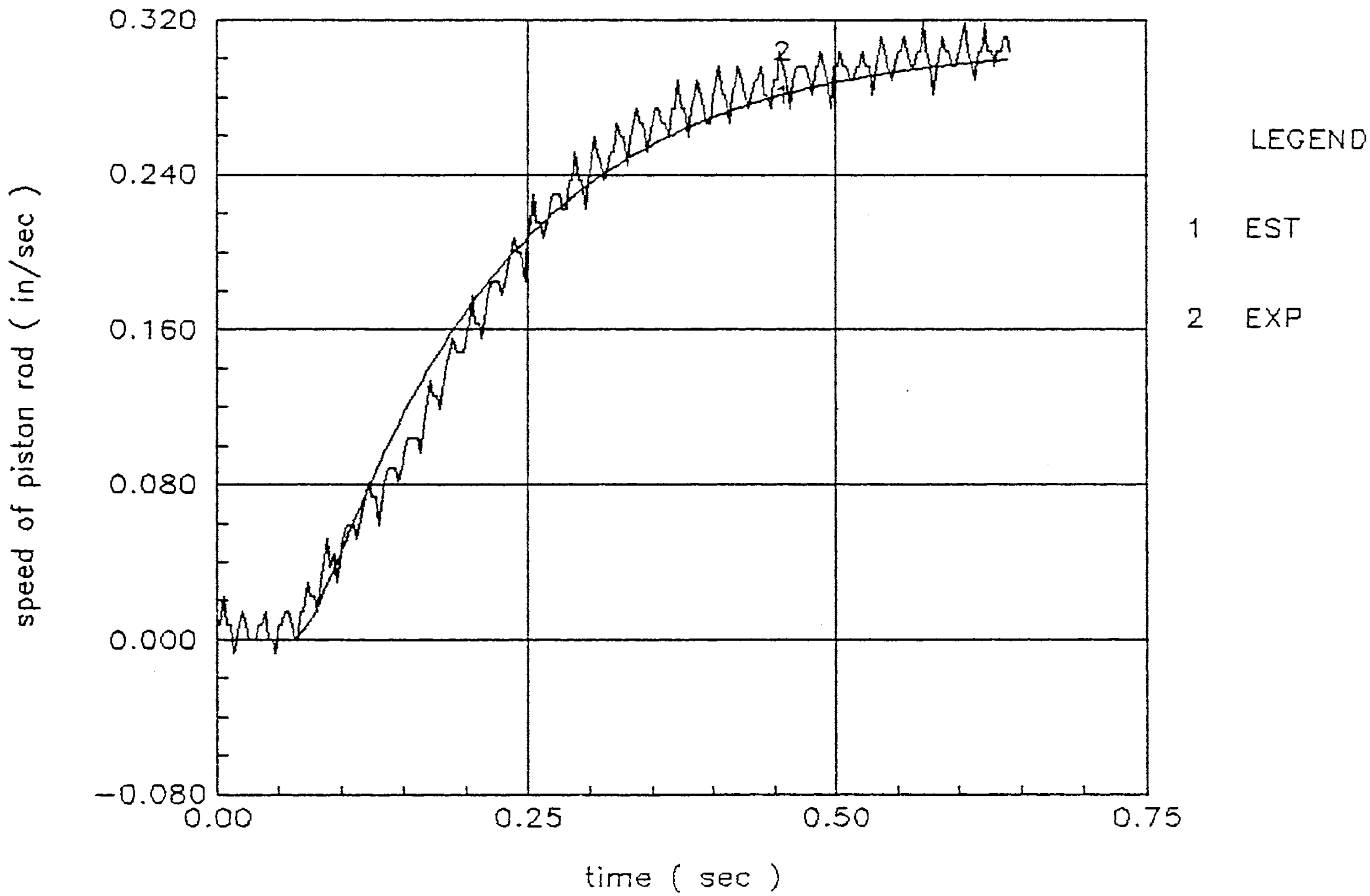

Figure 95. Identification of 100 Inches Line System

TABLE VIII

DAMPING CONSTANTS AND NATURAL FREQUENCIES

	10" System	100" System
Damping Constant	1.8	1.7
Natural Frequency (Hz)	4.0	3.2

As line length increased, damping constant and natural frequency decreased. This coincides with the results in computer simulation in Chapter 4. These systems have high damping constants and low natural frequencies. These means that a smaller cylinder, a larger valve, bigger oil volume are required in order to obtain underdamped and higher natural frequency response. The system natural frequency was so low that the transmission line dynamics could not appeared apparently.

Control

PI and PID Position Control

The velocity responses were well developed overdamped response. Therefore, position control was only considered.

The derivative term in PID controller may make the response chattering at the location of steady state value. Thus, PI control was applied first. The gains were determined by Ziegler-Nichols Method. Positive and negative 10 mA was used as the saturation current of the valve to assure the linearity of the flow gain.The following PI and PID gains were used both of 10 inches line system and 100 inches line system:

PI control: K_P = 0.2259

K_i = 10.7263

PI control: K_P = 0.3012

K_i = 21.4526

K_d = 0.0010

Figures 96 through 99 show that the piston reaches tothe designated location (5.77 inch of piston displacement, equivalent to 1.5 V of LVDT difference voltage) with little chattering. The response delays are:

PI, 10 inches line : 0.0195 sec

PID, 10 inches line : 0.0130 sec

PI, 100 inches line : 0.0325 sec

PID, 100 inches line : 0.0260 sec

Comparing to the response delays in previous open loop responses(10 inches line : 0.035 sec, 100 inches line : 0.070 sec), the response dealys reduced. The servovalve currents oscillated even at the steady state position, although the average value was offset current. It was sure that the piston rod moved to the designated position, and converged. Therefore, It is believed that the electrical noise make the proportional gain in PID controller create the oscillating magnitude signal.

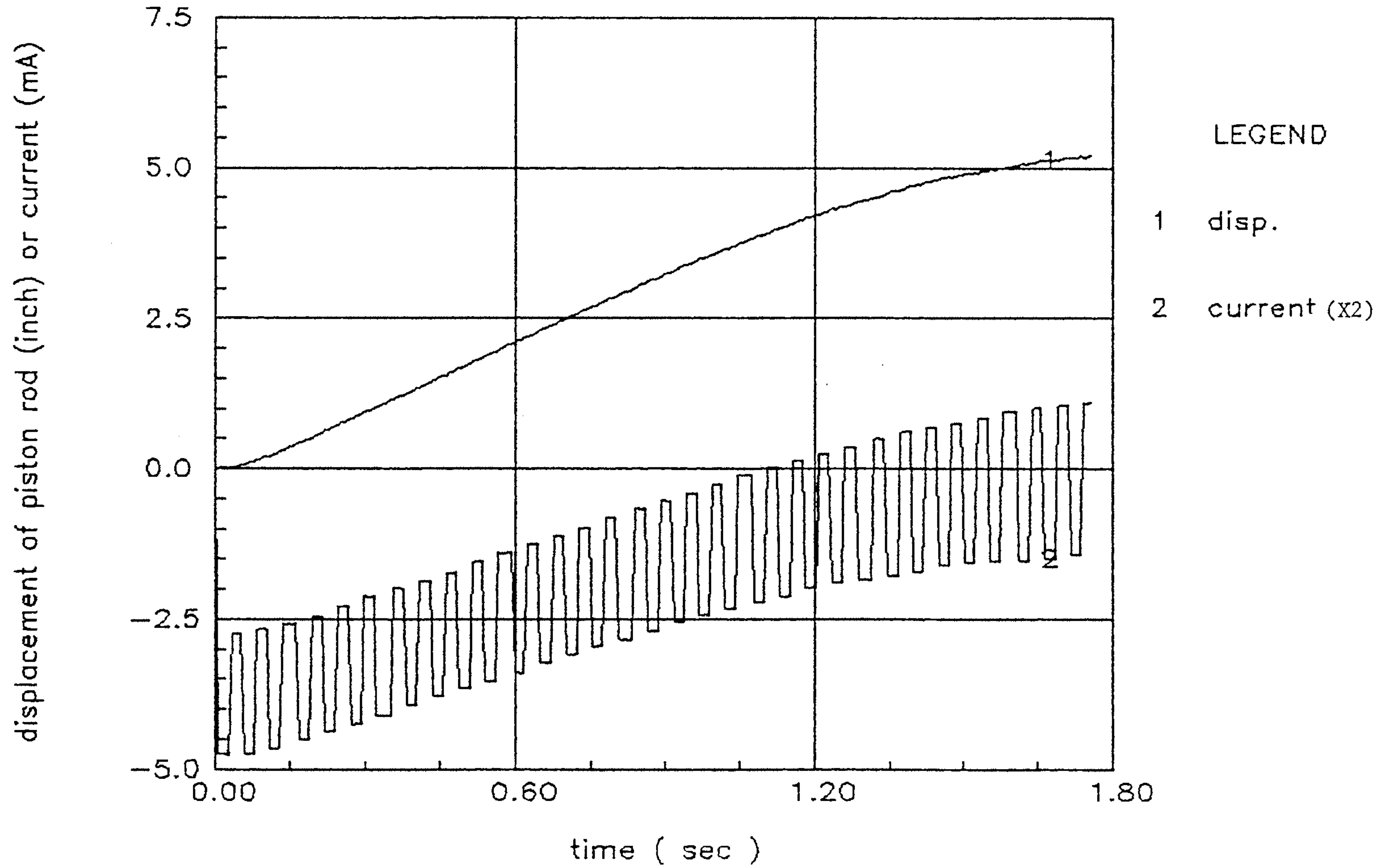

Figure 96. PI Position Control (10 inch)

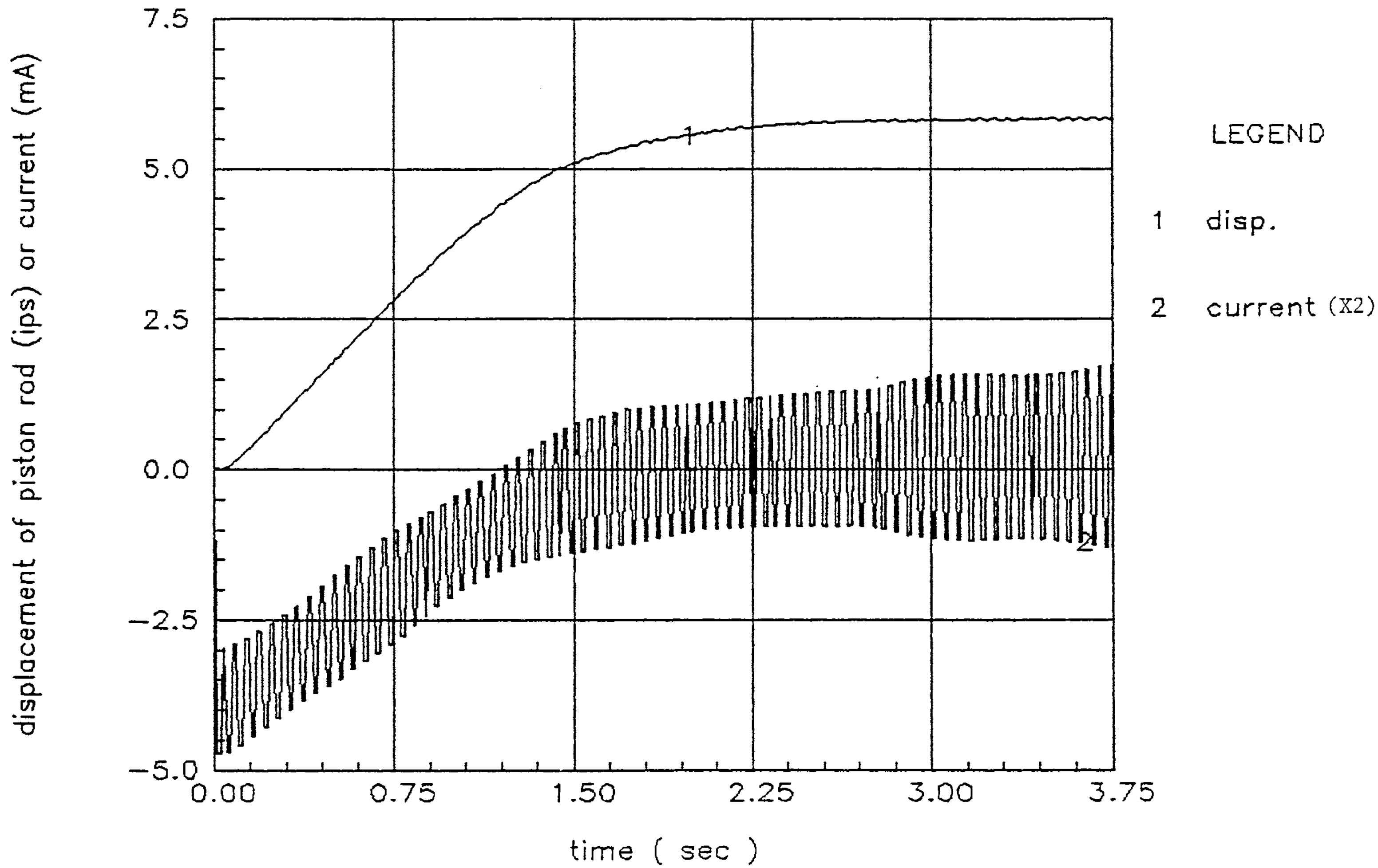

Figure 97. PID Position Control (10 inch)

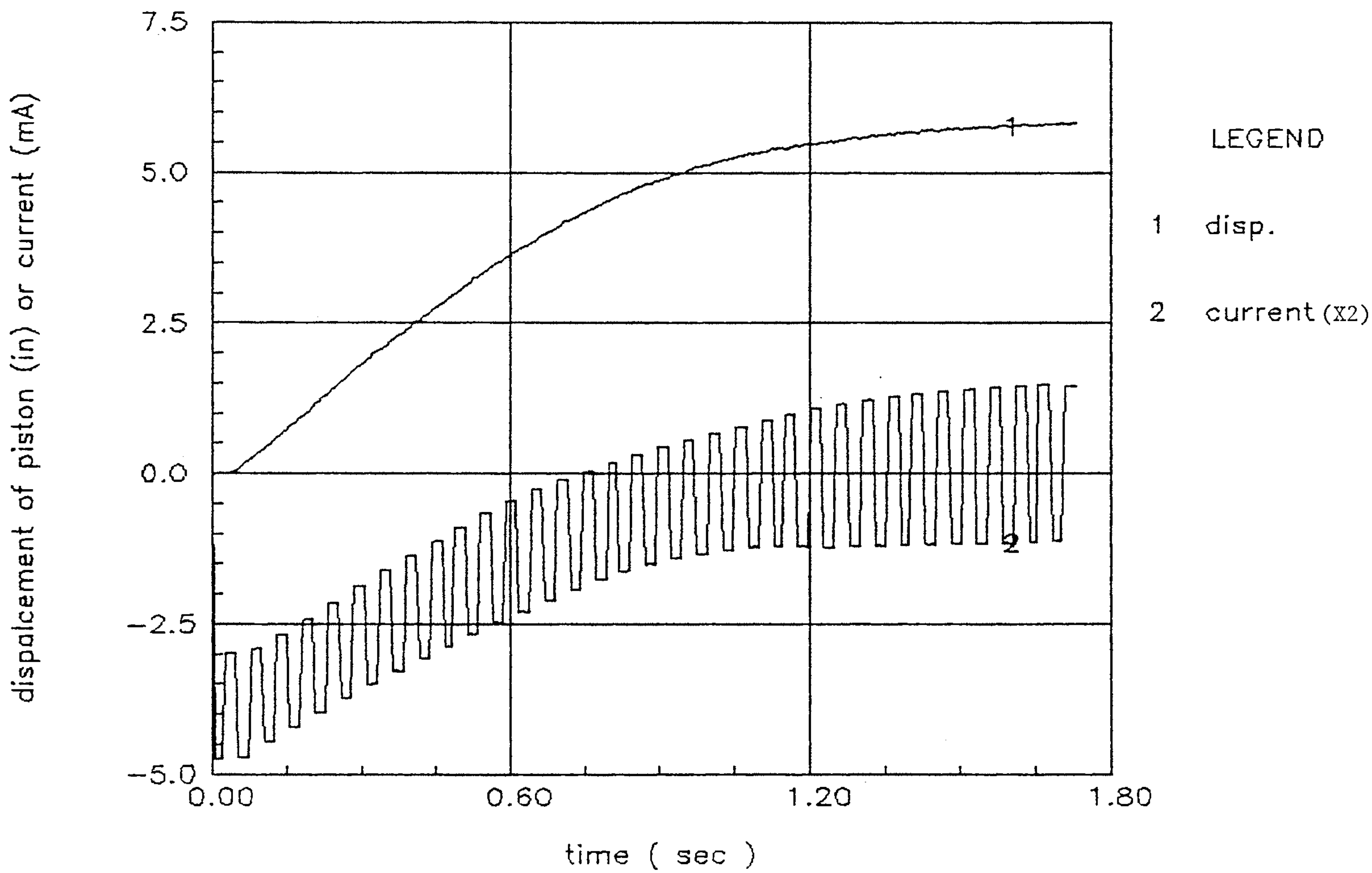

Figure 98. PI Position Control (100 inch)

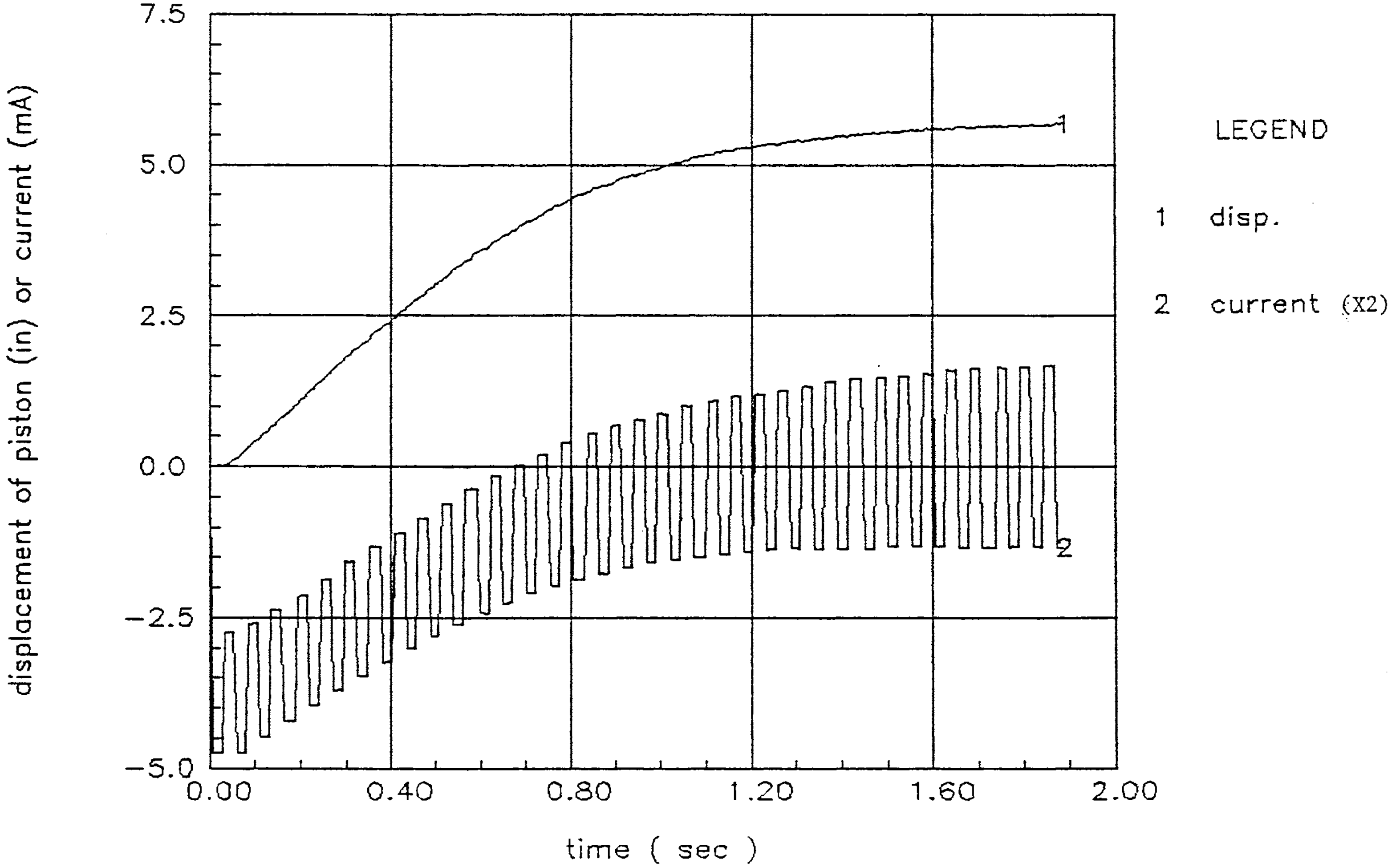

Figure 99. PID Postion Control (100 inch)

VSS Position Control

Based on equations (128) and (129), the control scheme was established in the following manner:

For 10inches line:

$$\text{sliding surface, } \sigma = 44X_1 + 7X_2 + X_3 \quad (130)$$

$$\text{control signal, } u = \psi_1 X_1 + \psi_2 X_2 \quad (131)$$

where,

$$\psi_1 = \begin{cases} \alpha_1 = 300, & \text{for } \sigma X_1 > 0 \\ \beta_1 = -300, & \text{for } \sigma X_1 < 0 \end{cases} \quad (132)$$

$$\psi_2 = \begin{cases} \alpha_2 = 15, & \text{for } \sigma X_2 > 0 \\ \beta_2 = -15, & \text{for } \sigma X_2 < 0 \end{cases} \quad (133)$$

For 100inches line:

$$\text{sliding surface, } \sigma = 28X_1 + 6X_2 + X_3 \quad (134)$$

$$\text{control signal, } u = \psi_1 X_1 + \psi_2 X_2 \quad (135)$$

where,

$$\psi_1 = \begin{cases} \alpha_1 = 100, & \text{for } \sigma X_1 > 0 \\ \beta_1 = -160, & \text{for } \sigma X_1 < 0 \end{cases} \quad (136)$$

$$\psi_2 = \begin{cases} \alpha_2 = 10, & \text{for } \sigma X_2 > 0 \\ \beta_2 = -20, & \text{for } \sigma X_2 < 0 \end{cases} \quad (137)$$

Figure 100 and 101 show the variation of the displacement of the piston rod when the preceding VSS control algorithm were implemented. The response dealay was 0.012 seconds for 10 inches line and0.013 seconds for 100 inches line. The response delay reduced comparing with open loop responses, PI control, andPID control. However the

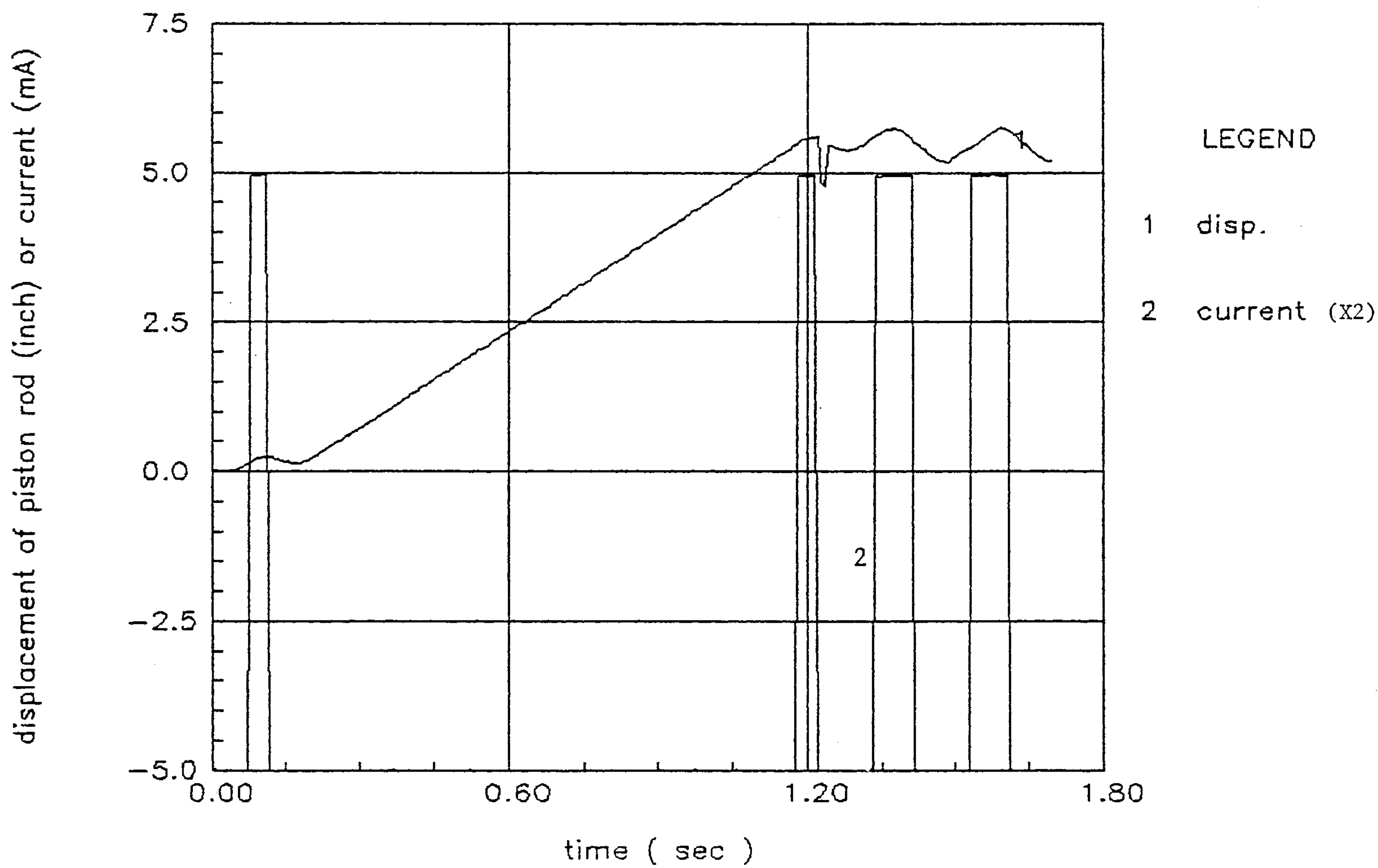

Figure 100. VSS Position Control (10 inch)

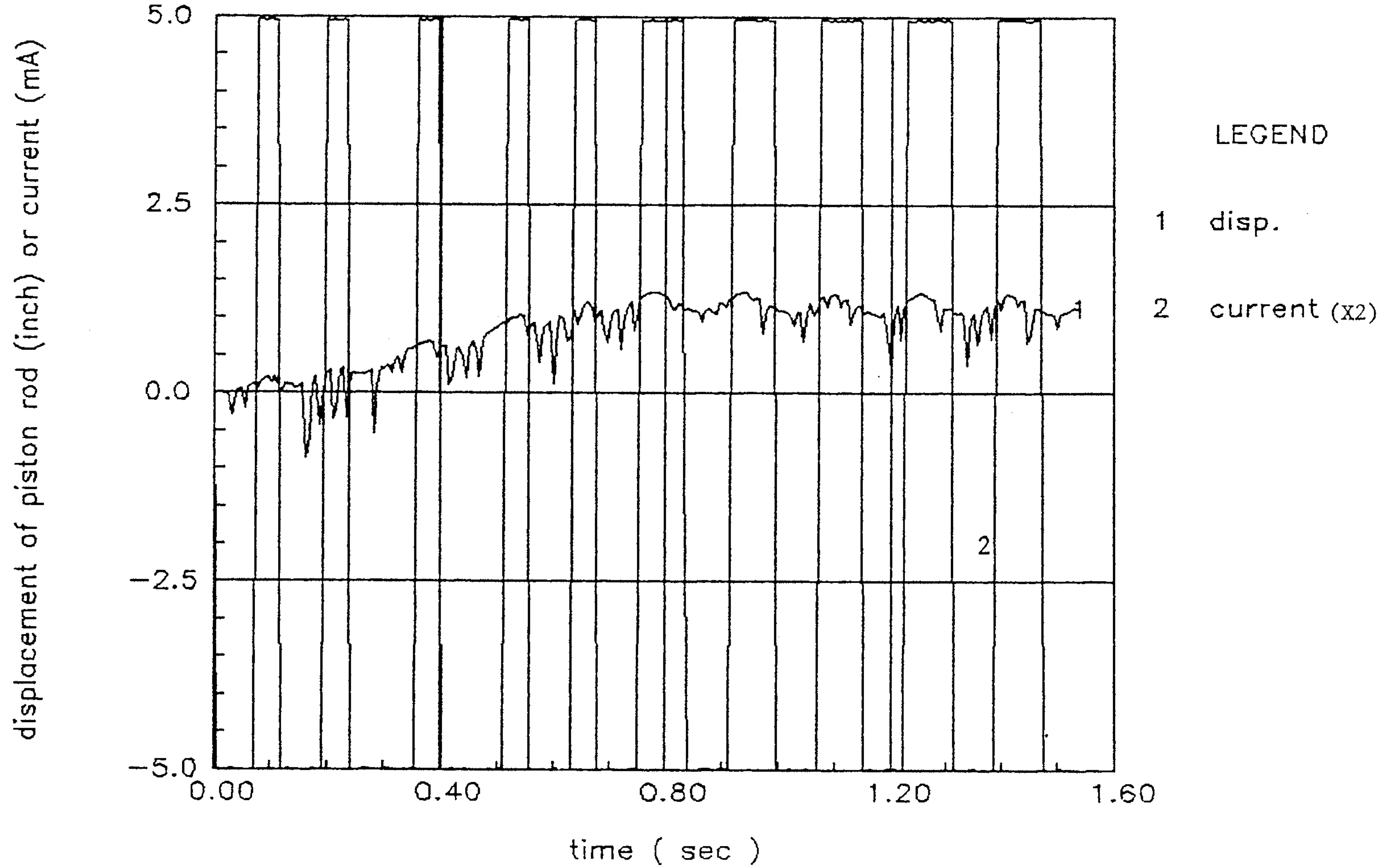

Figure 101. VSS position Control (100 inch)

chattering at steady state position existed. Especially for the 100 inches line, the sliding surface seems to be pierced. The reasons which can be thought are:

(1) The inaccuracy of parameter estimation.

(2) The electric signal delay between the computer to servovalve including computer executing time. For instance, the time step used in Chapter IV for theoretical VSS control was 0.25 msec. However, 26 msec was consumed in the test in order to generate one feedback signal.

(3) The nonlinearity of the servovalve especially at null position

CHAPTER VI

CONCLUSIONS

Through this research following tasks were carried out to investigate the effect of line dynamics and its control in the servovalve controlled - double rod cylinder system:

For theoretical approach:

- Lumped parameter line modelling by four pole equation
- Distributed parameter line modelling by the method of characteristics and the FDM
- Coupling method of distributed parameter lines and other lumped parameter hydraulic components in order to obtain time domain response
- Frequency response analysis using transfer functions of lumped mass and lumped parameter line
- System identification using pulse testing mathod
- Parameter estimation using Orthogonal Projection Algorithm
- PID velocity and position control with the gains determined by Powell's optimization method
- VSS velocity and position control

For experimental support:

- Velocity response of piston rod, and comparision to the computer simulation result

- Identification of system tranfer function
- PI and PID position control
- VSS position control

Taking observation of the results of above tasks lead to the following conclusions:

1. The method of characteristics is more reliable than the FDM for the distributed parameter line modelling. The method of characteristics is less time comsuming for computation, more stable for time step, and closer to the inherent characteristic of a line than the FDM.

2. Quasi-explicit method using the FDM is more efficient than implicit method. The alternating order of difference equations for the pressure P and the flow rate Q creates little difference. Internal iteration for convergence in the implicit method is not absolute.

3. The coupling method of distributed parameter line modelling and other lumped parameter hydraulic components, addressed in this study for a first time should be used to obtain the accurate time domain response. No matter how complex the system is, the distributed parameter lines are conveniently coupled with other hydraulic components. Alikeness to the damping constants and the natural frequencies existed between numerical computation and experiment.

4. Linearized transfer function method with lumped mass line should be used only for the short line.

Distributed parameter line modelling must be used for the long line system.

5. When the line is divided more than 4 sections for the distributed parameter line modelling using method of characteristics, the response delay is abnormally long. This supports four sections division of the transmission line using the method of characteristics.

6. Pulse testing method can be used for system identification regardless of line length. Linearized transfer function with lumped mass line can be used only for the short line. Linearized transfer function with lumped parameter line modelling can be used for the long line.

7. The line dynamics effect is apparent especially when the natural frequency of a system is high. The long line, coupled with the servovalve controlled actuator system of higher natural frequency affects the total hydraulic system response.

8. PI or PID feedback control can be used for controlling the line dynamics in the servovalve controlled - double rod cylinder system. PI or PID control can also reduces the response delay. Both of PI and PID feedback controls accomplished successfuly the position control without chattering at the steady state position in the tests.

9. Theoretically, VSS control creates better transient response than PID feedback control does. However, the

computer execution speed must be fast enough to meet the response speed of the system for the VSS control algorithm to be implemented practically. The VSS control can be used for controlling line dynamics if chattering at steady state value can be eliminated. It can reduce the response delay more efficiently than PID.

LITERATURE CITED

1. F. D. Ezekiel, H. M. Paynter, Fluid Power Control edited by Blackburn, Reethof and Shearer, The Technology Press of M.I.T. and John Wiley & Sons, Inc., (1960), pp. 130 - 140.

2. R. Oldenburger, J. Donelson Jr., AIEE Transactions, (1962), paper No. 62-167,

3. J. D. Regetz, NASA Technical Note D-576, (1960).

4. R. Oldenburger, R. E. Goodson, Journal of Basic Engineering, ASME, Seried D, (1964), Vol. 86, pp. 1 - 10.

5. F. T. Brown, Journal of Basic Engineering, ASME, (1962), pp. 547 - 553.

6. W. Zielke, Journal of Basic Engineering, ASME, (1968), pp. 109 - 114.

7. F. T. Brown, Journal of Basic Engineering, ASME, (1969), pp. 217 - 227.

8. G. R. Keller, Hydraulic System Analysis, Hydraulics & Pneumatics, (1972).

9. A. J. Healy, D. A. Hullender, Fluid Transmission Line Dynamics, Winter Annual Meeting of the ASME, (1981), pp. 57 - 108.

10. D. A. Hullender, R. L. Woods, C. H. Hsu, Fuild Transmission Line Dynamics, Winter Annual Meeting of the ASME, (1983), pp. 78 - 96.

11. T. J. Viersma, Analysis, Synthesis and Design of Hydraulic Servosystems and Piplines, Elsevier, (1980), pp. 129 - 229.

12. J. Watton, Transmission Line Dynamics 1983, The Winter Annual Meeting of the ASME, (1983), pp. 98 - 112.

13. J. Watton, Transmission Line Dynamics 1983, The Winter Annual Meeting of The ASME, (1983), pp. 113 - 125.

14. J. Watton, Journal of Dynamic Systems, Measurement, and Control, Vol. 106, (1984), pp. 157 - 162.

15. J. Watton, Journal of Dynamic Systems, Measurement, and Control, Vol. 109, (1987), pp. 15 - 18.

16. J. Zongxia, C. Yigang, S. Jingchao, Fluid Power Transfer and Control - Proceedings of The 2nd International Conference, (1989), pp. 151 - 155.

17. W. Bache, W. Hoffmann, BHRA 6th International Fluid Power Simposium, Cambridge, (1981)

18. S. I. Pai, Viscous Flow Theory, Vol. 1, Van Nostrand Company, Inc., (1956), p. 38.

19. A. F. D'souza, R. Oldenburger, Journal of Basic Engineering, (1964), pp. 589 - 598.

20. J. Watton, Fluid Power Systems - Modelling, Simulation, Analog and Microcomputer Control, Prentice Hall, (1989).

21. R. Oldenburger, R. E. Goodson, Journal of Basic Engineering, (1962), paper No. 62-WA-55, PP. 1 - 8.

22. J. M. Kirshner, S. Katz, Design Theory of Fluidic Components, Academic Press,(1965).

23. H. E. Merritt, Hydraulic Control Systems, John Wiley & Sons, (1967), pp. 145 - 149.

24. R. D. Richhart, Basic Fluid Power Research Program, Annual report number 14, Fluid Power Research Center, Oklahoma State University, (1981), pp. 185 - 191.

25. I. T. Hong, J. C. Lee, SAE Technical Paper Series No. 891849, (1989)

26. W. L. Luben, Process Modelling, Simulation,and Control for Chemical Engineers, McGraw-Hill Book Co., (1975), PP. 279 - 296.

27. G. C. Goodwin, K. S. Sin, Adaptive Filtering Prediction and Control, Prentice Hall, Inc., (1984), pp. 57 - 58.

28. K. J. Åström, B. Wittenmark, Adaptive Control, Addison -Wesley Publishing Co., (1989), pp. 389 - 395.

29. Y. Takahashi, M. J. Rabins, D. M. Auslander, Control and Dynamic Systems, Addison-Wesley Publishing

Co., (1970), pp. 476 - 479.

30. W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, Numerical Recipes - The Art of Scientific Computing, Cambridge University Press ,(1986), pp. 294 - 299.

31. Y. R. Xu, Z. C. He, Basic Fluid Power Research Program, Vol. 18, No. 1, (1985), pp. 37 - 43.

32. K. J. Åström, B. Wittenmark, Computer Controlled Systems - Theory and Design, Prentice Hall, (1989), pp. 180 - 187.

33. U. Itkis, Control Systems of Variable Structure, John Wiley & Sons, (1976).

34. V. I. Utkin, IEEE Transactions on Automatic Control, Vol. AC-22, (1977), pp. 212 - 222.

REFERENCES

Atkinson, K. E. (1978). *An Introduction to Numerical Analysis.* John Wiley & Sons Ltd.

Brophy, J. J. (1977). *Basic electronics for scientists.* McGraw-Hill Inc.

Currie, I. G. (1974). *Fundamental mechanics of fluids.* McGraw-Hill Inc.

Duntemann, J. (1989). *Complete Turbo Pascal, 3rd ed.* Scott, Foresman and Company.

Frederick, C. (1978). *Modeling and analysis of dynamic systems.* Houghton Mifflin Co.

Goodson, R. E., Leonard, R.G. (1972). *A survey of modelling techniques for line transients.* Journal of Basic Engineering, Transactions of the ASME, 474 - 482.

Hullender, D. A., Healey, A. J. (1981). *Rational polynomial approximations for fluid transmission line models.* Fluid Transmission Line Dynamics, The Winter Annual Meeting of the ASME, 33 - 55.

Jacques, J., Slotine, E. (1984), *Sliding controller design for non-linear systems.* International Journal of Control, 40(2), 421 - 434.

Johnson, J. E. (1977). *Electrohydraulic servo systems.* Penton Publishing, Inc.

Jung, W. G. (1976). *IC OP-Amp cookbook.* Howard W. sams & Co., Inc.

Merriam III, C. W. (1964). *Optimization theory and the design of feedback control systems.* McGraw-Hill Inc.

Mitchell, A. R. (1969). *Computational methods in partial differential equations.* John Wiley & Sons Ltd.

Ogata, K. (1970). *Modern control Engineering.* Prentice-Hall, Inc., 563 - 620.

Opitz, H. P. (1986). *Robustness properties of discrete-variable structure controllers.* International

Journal of Control, 43(3), 1003 - 1014.

Poloni, M., Winterbone D. E., & Nichols, J. R. (1987). *Comparision of unsteady flow calculations in a pipe by the method of characteristics and the two-step differential Lax-Wendroff Method.* International Journal of Mechanical Science, 29(5), 367 - 378.

Slotine, J. J., Sastry, S. S. (1983). *Tracking control of non-linear systems using sliding surfaces, with application to robot manipulators.* International Journal of Control, 38(2), 465 - 492.

Vichnevetsky, R. (1981). *Computer methods for partial differential equations, volume I.* Prentice-Hall, Inc.

APPENDIXES

APPENDIX A

TRANSFER FUNCTION - LUMPED MASS LINE

```
(* This is a program to obtain the dynamic response of the cylinder in  *)
(* the time domain from the linearized transfer function of the system. *)

PROGRAM TRANS;

CONST
{ constants for simulation }
  n=6; { state variables number }
  h=1.0e-4; { simulation time step, sec }
  final_time=1.0; { simulation time limit, sec }
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Di=0.25; { tube inner diameter, in }
  Lt=20; { total transmission line length, in }
  Va=5; { total volume of oil except line volume, in^3 }
  Kqi=0.0012; { valve current - dispalcement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ctp=0.0; { total leakage coefficient in the cylinder, in^3/sec/psi }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

TYPE
  glnarray=array[1..n] of real;
  glarray=array[1..4,0..300] of real;

VAR
  i,j,k,it                              :integer;
  A1,A2,B1,B2,B3,Ff,time,Vp,Xp,
  Ki,Vt,Kc,Kce,ss,
  current,output_time                   :real;
  y,dydx,yout                           :glnarray;
  r                                     :glarray;
  t                                     :array [0..300] of real;
  f                                     :text;


{-----------------------------------------------------------------------------}
 Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{-----------------------------------------------------------------------------}
 begin
   dydx[1]:=y[2];
   dydx[2]:=y[3];
   dydx[3]:=(-1/A1)*y[2]+(-A2/A1)*y[3]+(B1/A1)*current;
   dydx[4]:=y[5];
   dydx[5]:=y[6];
   dydx[6]:=(-1/A1)*y[5]+(-A2/A1)*y[6]+((A2*B3/A1-B2)/A1)*Ff;
 end;
```

```
{-----------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
              VAR yout: glnarray);
{-----------------------------------------------------------------------}

VAR
   i: integer;
   xh,h6,hh: real;
   dym,dyt,yt: glnarray;
BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;
END;

{-----------------------------------------------------------------------}
 Procedure initialize_parameters;
{-----------------------------------------------------------------------}
 var
   Ai,Ap,Mt,Kq,wh,dh,Xv,Ff,PL : real;

 begin
   time:=0.0;
   it:=0;
   j:=0;

   Ai:=pi*Di*Di/4; { line inner area, in^2 }
   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
   Vt:=Va+Ai*Lt; { total volume of oil, in^2 }
   Ff:=Mw*fc; { friction force, psig }
   PL:=Ff/Ap; { load pressure, psig }
   Kq:=Cd*Wd*sqrt((Ps-PL)/ro); { flow gain - valve displacement, in^2/sec }
   Ki:=Kq*Kqi; { flow gain - current, in^3/sec-mA }
   Xv:=current*Kqi; { spool displacement, in }
```

```
   Kc:=Cd*Wd*Xv*sqrt((Ps-PL)/ro)/(2*(Ps-PL));
                   { flow-pressure coefficient of valve, in^3/sec/psi }
   Kce:=Kc + Ctp; { total flow-pressure coefficient, in^3/sec/psi }
   wh:=sqrt(4*beta*Ap*Ap/(Vt*Mt)); { hydraulic natural frequency, rad/sec }
   dh:=(Kce/Ap)*sqrt(beta*Mt/Vt); { damping ratio }
   Ff:=Mw*fc; { friction force, lbf }

   A1:=1/(wh*wh);
   A2:=2*dh/wh;
   B1:=Ki/Ap;
   B2:=Kce/(Ap*Ap);
   B3:=Vt/(4*Ap*Ap*beta);

   for i:=1 to n do begin
     y[i]:=0;
     DYDX[I]:=0.0;
   end;

   t[0]:=0;
   for i:=1 to 4 do
     r[i,0]:=0;
{
   ss:=B1*current;
   writeln('ss = ',ss:10:4,'w = ',wh:10:4,' ','d = ',dh:10:4);
}
 end;

{------------------------------ MAIN PROGRAM -------------------------------}

BEGIN
  assign(f,'d:Trans.out');
  rewrite(f);
  current:=5;
  k:=1;
  repeat
  initialize_parameters;
  repeat
    it:=it+1;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
    Vp:=y[2]+y[5];
    Xp:=y[1]+y[4];
    writeln(time:6:4,' ',Xp:10:4,' ',Vp:10:4);
    if it>=10 then
      begin
        j:=j+1;
        t[j]:=time;
        r[k,j]:=Vp;
        it:=0;
      end;
   until (time>final_time) or (j>=200);
   k:=k+1;
```

```
   current:=current+15;
   until k>=3;

   for i:=0 to 200 do
     begin
       write(f,t[i]:6:4,' ',r[1,i]:10:4,' ',r[2,i]:10:4);
       writeln(f);
       write(t[i]:6:4,' ',r[1,i]:10:4,' ',r[2,i]:10:4);
       writeln;
     end;

   close(f);
end.
```

APPENDIX B

FINITE DIFFERENCE METHOD

```
(* This ia a program for distributed parameters line modelling using    *)
(* QUASI-EXPLICIT finite difference method.                              *)

PROGRAM LINE_DYNAMICS;

const
  n = 5;  { total node number devided }
  beta = 1.5e5;  { bulk modulus }
  dt = 0.0005;  { sampling time }
  dx = 50;  { devided line length }
  ro = 0.78e-4; { oil density }
  mu = 2.8e-6; { absolute viscosity }
  end_time =  0.15; { simulation time }
  d = 0.125; { line inner diameter }

var
  i : integer;
  p,q : array [1..n] of real;
  time,area,R,L,C,ra : real;
  f : text;

begin
  assign(f,'d:line.out');
  rewrite(f);
  area:= pi*d*d/4;
  R:=128*mu/(pi*d*d*d*d);
  L:=ro/area;
  C:=area/beta;
  ra:=dt/dx;
  time:=0;
  for i:=1 to n do
    begin
      q[i]:=0;
      p[i]:=0
    end;

  repeat
    time := time+dt;
    q[1] := 50*area;
    q[n] := -(ra/L)*(p[n]-p[n-1]) + q[n] -(R*dt/L)*q[n];
    for i:=2 to n-1 do
      q[i] := -0.5*(ra/L)*(p[i+1]-p[i-1]) + q[i] -(R*dt/L)*q[i];
    p[1] := -(ra/C)*(q[2]-q[1]) + p[1];
    if p[1]<0 then p[1]:=0;
    p[n]:=0;
    for i:=2 to n-1 do
      begin
        p[i] := -0.5*(ra/C)*(q[i+1]-q[i-1]) + p[i];
        if p[i]<0 then p[i]:=0;
      end;
    write(f,time:6:4);

    for i:=1 to n do
      write(f,' ',q[i]);
    writeln(f);
```

```
   until time>=end_time;
   close(f);
end.
```

APPENDIX C

METHOD OF CHARACTERISTICS

```
(* This is a program for a distributted parameters line modelling by *)
(* explicit METHOD OF CHARCTERISTIC.                                   *)

PROGRAM LINE_DYNAMICS;

const
  n = 4; { total divided node number = 2*n + 1 }
  beta = 1.5e5; { bulk modulus }
  dt = 0.0005; { smapling time }
  dx = 50; { divided line length }
  ro = 0.78e-4; { oil density }
  mu = 2.8e-6; { absolute viscosity }
  end_time = 0.15; { simulation time }
  d = 0.125; { inner tube diameter }
  nn = 9; { total divided node number }

var
  i,j : integer;
  p,q : array [1..nn] of real;
  a,R,L,Co,time : real;
  f : text;

begin
  assign(f,'d:line.out');
  rewrite(f);
  a:=pi*d*d/4;
  R:=128*mu/(pi*d*d*d*d);
  L:=ro/a;
  Co:=dx/(dt/2);

  time:=0;
  for i:=1 to 2*n+1 do
    begin
      Q[i]:=0;
      P[i]:=0
    end;
    Q[1] := 50*a;
    P[2*n+1]:=0;

  repeat
    time := time+dt;
    for i:=1 to n do
      begin
        j:=i*2;
        P[j]:=(P[j-1]+P[j+1])/2 +
              (Co*L-R*dx/4)*(Q[j-1]-Q[j+1])/2;
        Q[j]:=((Co*L-R*dx/4)*(Q[j-1]+Q[j+1]) + (P[j-1]-P[j+1]))
              /(2*(Co*L+R*dx/4));
        if P[j]<0 then P[j]:=0;
      end;
    P[1]:=P[2]+Co*L*(Q[1]-Q[2])+R*(Q[1]+Q[2])*dx/4;
    if P[1]<0 then  P[1]:=0;
    for i:=1 to n-1 do
      begin
        j:=i*2+1;
```

```
        P[j]:=(P[j-1]+P[j+1])/2 +
              (Co*L-R*dx/4)*(Q[j-1]-Q[j+1])/2;
        Q[j]:=((Co*L-R*dx/4)*(Q[j-1]+Q[j+1]) + (P[j-1]-P[j+1]))
              /(2*(Co*L+R*dx/4));
        if P[j]<0 then P[j]:=0;
      end;
    Q[2*n+1]:=((Co*L-R*dx/4)*Q[2*n]-P[2*n+1]+P[2*n])/(Co*L+R*dx/4);
    write(time:10:4,' ');
    write(f,time:10:4,' ');
    for i:=0 to n do
      begin
        j:=2*i+1;
        write(Q[j]:10:4,' ');
        write(f,Q[j]:10:4,' ');
      end;
    writeln;
    writeln(f);
   until time>end_time;
   close(f);
end.
```

APPENDIX D

DISTRIBUTED COMPONENTS MODELLING
(LUMPED MASS LINE)

```
(*  This is a program for distributed components modelling, in which     *)
(*  the line is still considered as a lumped mass.                       *)

PROGRAM DIST;

CONST
{ constants for simulation }
  n=4;
  h=1.0e-4;
  final_time=0.3;
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Di=0.25; { tube inner diameter, in }
  Lt=200; { total transmission line length, in }
  Va=5; { total volume of oil except line volume, in^3 }
  Kqi=0.0012; { valve current - displacement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

TYPE
  glnarray=array[1..n] of real;
  gtnarray=array[1..200] of real;
  grnarray=array[1..4,1..200] of real;

VAR
  i,j,k,it,data_no                :integer;
  Ff,time,Ap,Vp,Xp,Mt,Kc,
  output_time,
  Xmin,Xmax,Xv,Vt,current       :real;
  y,dydx,yout                   :glnarray;
  r                             :grnarray;
  t                             :gtnarray;
  f                             :text;


{-----------------------------------------------------------------------------}
 Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{-----------------------------------------------------------------------------}
 var
   P1,P2,P3,P4,V5,X6,S5,
   rate_current,sir,Cd1,Cd2,Qd1,Qd2,Qd3,Qd4,
   change_time,CC1,CC2,CC3,Qc1,Qc2,FC3,
   Bb,Ks,FM1,
   DQ2,DQ3,DF5,DX6                              : real;

 begin
```

```
P1:=Ps; P2:=Y[1]; P3:=Y[2]; P4:=0; V5:=Y[3]; X6:=Y[4];

{ Power source }


{ Servo Valve }
  Xv:= current*Kqi;
  Kc:= pi*Xv*Xv*Wd/(32*vis);
  Cd1:=Cd; Cd2:=Cd;

  if current=0 then
    begin
      Qd1:=0;
      Qd2:=0;
      Qd3:=0;
      Qd4:=0;
    end
  else
    begin
      if current >0 then
        begin

          if P1 < P2 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P2-P1)/ro)
                     else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P2)/ro);
          if P3 < P4 then Qd3:=Cd2*Xv*Wd*sqrt(2*(P4-P3)/ro)
                     else Qd3:=-Cd2*Xv*Wd*sqrt(2*(P3-P4)/ro);
{
          if (time < 0.002) then
            begin
              Qd1:=Qd1*(time)/0.002;
              Qd3:=Qd3*(time)/0.002;
            end;

          if P1 < P2 then Qd1:=0.2*(current/10)*sqrt(P2-P1)
                     else Qd1:=-0.2*(current/10)*sqrt(P1-P2);
          if P3 < P4 then Qd3:=0.2*(current/10)*sqrt(P4-P3)
                     else Qd3:=-0.2*(current/10)*sqrt(P3-P4);
}
          Qd2:=-Qd1;
          Qd4:=-Qd3;
        end
      else
        begin
          if P1 < P3 then Qd1:=Cd1*sqrt(P3-P1)
                     else Qd1:=-Cd1*sqrt(P1-P3);
          if P2 < P4 then Qd2:=Cd2*sqrt(P4-P2)
                     else Qd2:=-Cd2*sqrt(P2-P4);
          if (time < 2.002) then
            begin
              Qd1:=Qd1*(time-0.2)/0.002;
              Qd2:=Qd2*(time-0.2)/0.002;
            end;
          Qd3:=-Qd1;
          Qd4:=-Qd2;
```

```
        end;
      end;

  { Cylinder }
    CC1:=0.00; CC2:=0.00; Xmax:=10.0; Xmin:=0.0;
    Qc1:=-(Ap*V5+CC1*(P2-P3)+CC2*P2);
    Qc2:=Ap*V5+CC1*(P2-P3)-CC2*P3;
    FC3:=Ap*(P2-P3);

  { Load }
    FM1:=-Ff;

  DQ2:=Qd2+Qc1;
  DQ3:=Qc2+Qd3;
  DF5:=FC3+FM1;
  if (DF5>=0) and (X6>=Xmax) then DF5:=0.0;
  if (DF5<=0) and (X6<=Xmin) then DF5:=0.0;
  DX6:=V5;

  DYDX[1]:=(BETA/(Vt/2+X6*Ap))*DQ2;
  DYDX[2]:=(BETA/(Vt/2-X6*Ap))*DQ3;
  DYDX[3]:=DF5/Mt;
  DYDX[4]:=DX6;

END; { OF DERIVS }


{---------------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
              VAR yout: glnarray);
{---------------------------------------------------------------------------}

VAR
   i: integer;
   xh,h6,hh: real;
   dym,dyt,yt: glnarray;
BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
```

```
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;

   if (yout[3]>0) and (yout[4]>=Xmax) then
     begin
       yout[3]:=0.0;
       yout[4]:=Xmax;
     end;
   if (yout[3]<0) and (yout[4]<=Xmin) then
     begin
       yout[3]:=0.0;
       yout[4]:=Xmin;
     end;
   for j:=1 to 2 do
     if yout[j]<0 then yout[j]:=0;

END;

{-----------------------------------------------------------------------------}
 Procedure initialize_parameters;
{-----------------------------------------------------------------------------}
 begin
   time:=0.0;
   it:=0;
   data_no:=0;

   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Vt:=Va + Lt*pi*Di*Di/4;
   Mt:=Mw/gravity; { total mass, lb }
   Ff:=Mw*fc; { friction force, lbf }

   for i:=1 to n do begin
     y[i]:=0;
     DYDX[I]:=0.0;
   end;
 end;

{------------------------------ MAIN PROGRAM ---------------------------------}

BEGIN
  assign(f,'d:Dist.out');
  rewrite(f);
  current:=5;
  k:=0;
  repeat;
  k:=k+1;
  initialize_parameters;
  repeat
    it:=it+1;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
```

```
    writeln(current:6:2,' ',time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,'
 ',y[4]:10:4);
    if it>=10 then
      begin

        writeln(f,time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,' ',y[4]:10
:4);

        data_no:=data_no+1;
        t[data_no]:=time;
        r[k,data_no]:=y[3];
        it:=0;
      end;
   until (time>final_time) or (data_no>=200);
   current:=current+5;
   until k>=1;
{
   for i:=1 to 200 do
     begin
       write(f,t[i]:6:4,' ',r[1,i]:10:4,' ',r[2,i]:10:4,
                       ' ',r[3,i]:10:4,' ',r[4,i]:10:4);
       writeln(f);
       write(t[i]:6:4,' ',r[1,i]:10:4,' ',r[2,i]:10:4,
                     ' ',r[3,i]:10:4,' ',r[4,i]:10:4);
       writeln;
     end;
}
   close(f);
end.
```

APPENDIX E

DISTRIBUTED COMPONENTS MODELLING
(DISTRIBUTED PARAMETER LINE)

```
(* This is a program for the dynamic response of the system while        *)
(* distributed parameter line modelling by explicit method of            *)
(* characteristics being used.                                           *)

PROGRAM DISTRIBUED_PARA_LINE_MODELLING_SYSTEM;

CONST
{ constants for simulation }
  n=6;
  h=1.0e-4;
  final_time=0.1;
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Va=5; { total volume of oil, in^3 }
  Kqi=0.0012; { valve current - spool displacement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

{ constant for simulating line }
  m = 4; { total divided node number nn= 2*m + 1 }
  dx = 2.5; { divided line length }
  d = 0.25; { inner tube diameter }
  nn = 9; { total divided node number }
  mu = 2.8e-6;

TYPE
  glnarray=array[1..n] of real;
  glmarray=array[1..nn] of real;

VAR
  i,j,k,it,data_no              :integer;
  hh,dt,
  Ff,time,Ap,Vp,Xp,Mt,
  QL1,QL2,QL3,QL4,Vt,Xv,Kc,
  current,output_time,
  R,L,Co,
  Xmin,Xmax,Atube,tube_volume   :real;
  y,dydx,yout                   :glnarray;
  Qa,Qb,Pa,Pb                   :glmarray;
  tout                          :array [1..200] of real;
  rout                          :array [1..4,1..200] of real;
  f                             :text;

{---------------------------------------------------------------------------}
 procedure LineA;
{---------------------------------------------------------------------------}
```

```
var
  i,j : integer;

begin

  Pa[1] := y[1];
  Pa[2*m+1]:=y[2];

  for i:=1 to m do
    begin
      j:=i*2;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
             (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
            (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
            /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  Qa[1]:=((Co*L-R*dx/4)*Qa[2]+Pa[1]-Pa[2])/(Co*L+R*dx/4);
  Qa[2*m+1]:=((Co*L-R*dx/4)*Qa[2*m]-Pa[2*m+1]+Pa[2*m])/(Co*L+R*dx/4);
  QL1:=-Qa[1];
  QL2:=Qa[2*m+1];
end;

{-----------------------------------------------------------------------------}
 procedure LineB;
{-----------------------------------------------------------------------------}
var
  i,j : integer;

begin

  Pb[1] := y[3];
  Pb[2*m+1]:=y[4];

  for i:=1 to m do
    begin
      j:=i*2;
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
             (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
```

```
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
            (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
            /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  Qb[1]:=((Co*L-R*dx/4)*Qb[2]+Pb[1]-Pb[2])/(Co*L+R*dx/4);
  Qb[2*m+1]:=((Co*L-R*dx/4)*Qb[2*m]-Pb[2*m+1]+Pb[2*m])/(Co*L+R*dx/4);
  QL3:=-Qb[1];
  QL4:=Qb[2*m+1];
end;

{-----------------------------------------------------------------------------}
 Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{-----------------------------------------------------------------------------}
 var
   P1,P2,P3,P4,P5,P6,V7,X8,S5,
   Cd1,Cd2,Qd1,Qd2,Qd3,Qd4,
   change_time,CC1,CC2,CC3,Qc1,Qc2,FC3,
   FM1,
   DQ2,DQ3,DQ4,DQ5,DF7,DX8                               : real;

 begin
   P1:=Ps; P2:=Y[1]; P3:=Y[2]; P4:=y[3]; P5:=y[4]; P6:=0;
   V7:=Y[5]; X8:=Y[6];

  { Power source }

  { Supply Line }

  { Signal }
    s5:=10.0;


  { Servo Valve }
    Xv:= current*Kqi;
    Cd1:=Cd; Cd2:=Cd;

    if current=0 then
      begin
        Qd1:=0;
        Qd2:=0;
        Qd3:=0;
        Qd4:=0;
      end
    else
      begin
        if current >0 then
          begin
            if P1 < P2 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P2-P1)/ro)
                       else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P2)/ro);
            if P5 < P6 then Qd3:=Cd2*Xv*Wd*sqrt(2*(P6-P5)/ro)
                       else Qd3:=-Cd2*Xv*Wd*sqrt(2*(P5-P6)/ro);
            if  (time < 0.002) then
              begin
```

```
                Qd1:=Qd1*(time)/0.002;
                Qd3:=Qd3*(time)/0.002;
              end;
            Qd2:=-Qd1;
            Qd4:=-Qd3;
          end
        else
          begin
            if P1 < P5 then Qd1:=Cd1*sqrt(P5-P1)
                       else Qd1:=-Cd1*sqrt(P1-P5);
            if P2 < P6 then Qd2:=Cd2*sqrt(P6-P2)
                       else Qd2:=-Cd2*sqrt(P2-P6);
            if (time < 2.002) then
              begin
                Qd1:=Qd1*(time-0.2)/0.002;
                Qd2:=Qd2*(time-0.2)/0.002;
              end;
            Qd3:=-Qd1;
            Qd4:=-Qd2;
          end;
      end;

  { Cylinder }
    CC1:=0.0; CC2:=0.0; Xmax:=10.0; Xmin:=0.0;
    Qc1:=-(Ap*V7+CC1*(P3-P4)+CC2*P3);
    Qc2:=Ap*V7+CC1*(P3-P4)-CC2*P4;
    FC3:=Ap*(P3-P4);

  { Load }
    FM1:=-Ff;

  { Drain Line }

  DQ2:=Qd2+QL1;
  DQ3:=QL2+Qc1;
  DQ4:=Qc2+QL3;
  DQ5:=QL4+Qd3;
  DF7:=FC3+FM1;
  if (DF7>=0) and (X8>=Xmax) then DF7:=0.0;
  if (DF7<=0) and (X8<=Xmin) then DF7:=0.0;
  DX8:=V7;

  DYDX[1]:=(BETA/0.5)*DQ2;
  DYDX[2]:=(BETA/((Va/2-0.5)+X8*Ap))*DQ3;
  DYDX[3]:=(BETA/((Va/2-0.5)-X8*Ap))*DQ4;
  DYDX[4]:=(BETA/0.5)*DQ5;
  DYDX[5]:=DF7/Mt;
  DYDX[6]:=DX8;

END; { OF DERIVS }


{-----------------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
```

```
                VAR yout: glnarray);
{----------------------------------------------------------------------------}

VAR
   i          : integer;
   xh,h6      : real;
   dym,dyt,yt: glnarray;

BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;

   if (yout[5]>0) and (yout[6]>=Xmax) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmax;
     end;
   if (yout[5]<0) and (yout[6]<=Xmin) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmin;
     end;
   for j:=1 to 4 do
     if yout[j]<0 then yout[j]:=0;

END;

{----------------------------------------------------------------------------}
 Procedure initialize_parameters;
{----------------------------------------------------------------------------}
 var
   Kq,wh,dh : real;

 begin
   time:=0.0;
   it:=0;
   data_no:=0;
```

```
  dt:=h;

  Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
  Mt:=Mw/gravity; { total mass, lb }
  Kq:=Cd*Wd*sqrt(Ps/ro); { flow gain, in^4/sec }
  Ff:=Mw*fc; { friction force, lbf }
  Atube:=pi*d*d/4;
  tube_volume:=Atube*m*dx;
  Vt:=Va+tube_volume*2;
  R:=128*mu/(pi*d*d*d*d);
  L:=ro/Atube;
  Co:=dx/(dt/2);

  for i:=1 to nn do
    begin
      Qa[i]:=0;
      Qb[i]:=0;
      Pa[i]:=0;
      Pb[i]:=0;
    end;
  for i:=1 to n do
    begin
      y[i]:=0;
      DYDX[I]:=0.0;
    end;
 end;

{------------------------------ MAIN PROGRAM -------------------------------}

BEGIN
  assign(f,'d:CLINE1.out');
  rewrite(f);
  current:=5; k:=1;
  repeat
  initialize_parameters;
  repeat
    it:=it+1;
    LineA;LineB;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
    writeln(current:5:1,' ',time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,'
 ',
                          y[4]:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
    if it>=10 then
      begin
        data_no:=data_no+1;
        tout[data_no]:=time;
        rout[k,data_no]:=y[5];
        it:=0;
      end;
  until (time>final_time) or (data_no>=200);
  current:=current+5; k:=k+1;
  until current>20;
```

```
   for i:=1 to 200 do
     begin
       writeln(f,tout[i]:8:4,' ',rout[1,i]:10:4,' ',rout[2,i]:10:4
                             ,' ',rout[3,i]:10:4,' ',rout[4,i]:10:4);
       writeln(tout[i]:8:4,' ',rout[1,i]:10:4,' ',rout[2,i]:10:4
                             ,' ',rout[3,i]:10:4,' ',rout[4,i]:10:4);
     end;
   close(f);
end.
```

APPENDIX F

FREQUENCY RESPONSE - LINERAIZED TRANSFER FUNCTION WITH LUMPED MASS LINE

```
(* This is a program to obtain the FREQUENCY RESPONSE of the system     *)
(* from the linearized transfer function of the system.                 *)

PROGRAM TRANS;

CONST
{ constants for simulation }
  n=6; { state variables number }
  h=1.0e-4; { simulation time step, sec }
  final_time=1.0; { simulation time limit, sec }
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Di=0.25; { tube inner diameter, in }
  Lt=20; { total transmission line length, in }
  Va=5; { total volume of oil except line volume, in^3 }
  Kqi=0.0012; { valve current - dispalcement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ctp=0.0; { total leakage coefficient of piston, in^3/sec/psi }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

TYPE
  glnarray=array[1..n] of real;
  glarray=array[1..4,0..300] of real;

VAR
  i,j,k,it                              :integer;
  A1,A2,B1,B2,B3,Ff,time,Vp,Xp,
  Ki,Vt,Kc,Kce,ss,Ap,
  current,output_time,
  w,wh,dh,fint,fr1,fr2,frr,fss          :real;
  y,dydx,yout                           :glnarray;
  r                                     :glarray;
  t                                     :array [0..300] of real;
  fr                                    :array [1..100] of real;
  db,theta                              :array [1..4,1..200] of real;
  f                                     :text;

{-----------------------------------------------------------------------------}
 Procedure initialize_parameters;
{-----------------------------------------------------------------------------}
 var
   Ai,Mt,Kq,Xv,PL : real;

 begin
   time:=0.0;
   it:=0;
```

```
   j:=0;

   Ai:=pi*Di*Di/4; { line inner area, in^2 }
   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
   Ff:=Mw*fc; { friction force, lbf }
   PL:=Ff/Ap; { load pressure, psig }
   Vt:=Va+Ai*Lt; { total volume of oil, in^2 }
   Kq:=Cd*Wd*sqrt((Ps-PL)/ro); { flow gain - valve displacement, in^2/sec }
   Ki:=Kq*Kqi; { flow gain - current, in^3/sec-mA }
   Xv:=current*Kqi; { spool displacement, in }
   Kc:=pi*Wd*Xv*sqrt((Ps-PL)/ro)/(2*(Ps-PL));
                    { flow-pressure coefficient of valve, in^3/sec/psi }
   Kce:=Kc + Ctp; { total flow-pressure coefficient, in^3/sec/psi }
   wh:=sqrt(4*beta*Ap*Ap/(Vt*Mt)); { hydraulic natural frequency, rad/sec }
   dh:=(Kce/Ap)*sqrt(beta*Mt/Vt); { damping ratio }

{
   writeln('ss = ',ss:10:4,'w = ',wh:10:4,' ','d = ',dh:10:4);
}
 end;

{------------------------------ MAIN PROGRAM -------------------------------}

BEGIN
  assign(f,'d:freq.out');
  rewrite(f);
  current:=5;  k:=0;
  repeat
    k:=k+1;
    initialize_parameters;
    w := 0.1*2*pi;
    fint := 20.0;
   for i:=1 to 100 do
      begin
        fr1 := 1.0 - w*w/(wh*wh);
        fr2 := 2.0*dh*(w/wh);
        frr := fr1*fr1 + fr2*fr2;
        frr := sqrt(frr);
        fr[i] := w/(2.0*pi);
        db[k,i] := 20.0*ln(Ki/Ap)/ln(10.0)-20.0*ln(frr)/ln(10.0);
        fss := fr2/fr1;
        theta[k,i] :=-arctan(fss);
        if (theta[k,i] > 0.0) then theta[k,i]:=-90.0+(theta[k,j]-90.0);
        w := w + fint;
      end;
     current:=current+5;
   until current>20;


  for j:=1 to 100 do
    begin
      writeln(fr[j]:10:4,' ',db[1,j]:10:4);
      writeln(f,' ',fr[j]:10:4,' ', db[1,j]:10:4,' ',db[2,j],' ',
                   db[3,j]:10:4,' ',db[4,j]:10:4,
```

```
                    theta[1,j]:10:4,' ',theta[2,j]:10:4,' ',theta[3,j]:10:4
                    ,' ',theta[4,j]:10:4);
    end;
  close(f);
end.
```

APPENDIX G

FREQUENCY RESPONSE - LINEARIZED TRANSFER FUNCTION WITH LUMPED PARAMETER LINE

```
(* This is a program to obtain the FREQUENCY RESPONSE of the system     *)
(* using linearized transfer function in which the line is modelled     *)
(* by lumped modelling with lossless line theory.                       *)

PROGRAM TRANS;

CONST
{ constants for simulation }
  n=6; { state variables number }
  h=1.0e-4; { simulation time step, sec }
  final_time=1.0; { simulation time limit, sec }
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Di=0.25; { tube inner diameter, in }
  Lt=200; { total transmission line length, in }
  Va=5; { total volume of oil except line volume, in^3 }
  Kqi=0.0012; { valve current - dispalcement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ctp=0.0; { total leakage coefficient of piston, in^3/sec/psi }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

TYPE
  glnarray=array[1..n] of real;
  glarray=array[1..4,1..300] of real;

VAR
  i,j,k,it                              :integer;
  A1,A2,B1,B2,B3,Ff,time,Vp,Xp,
  Ap,Ki,Vt,Kc,Kce,ss,st,
  Co,Lc,Zca,Gr,Rv,Lm,alpa,
  current,output_time,check,
  w,wh,dh,fint,fr1,fr2,frr,fss          :real;
  fr                                    :array [1..100] of real;
  r,db,theta,scale                      :glarray;
  f                                     :text;

{----------------------------------------------------------------------------}
 Procedure Initialize_Parameters;
{----------------------------------------------------------------------------}
 var
   Ai,Mt,Kq,Rl,Xv,PL  : real;
 begin

   Ai:=pi*Di*Di/4; { line inner area, in^2 }
   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
```

```
    Vt:=Va+Ai*Lt; { total volume of oil, in^2 }
    Ff:=Mw*fc; { friction force, lbf }
    PL:=Ff/Ap; { load pressure, pisg }
    Kq:=Cd*Wd*sqrt((Ps-PL)/ro); { flow gain - valve displacement, in^2/sec }
    Ki:=Kq*Kqi; { flow gain - current, in^3/sec-mA }
    Xv:=current*Kqi; { spool displacement, in }
    Kc:=Cd*Wd*Xv*sqrt((Ps-PL)/ro)/(2*(Ps-PL));
                     { flow-pressure coefficient of valve, in^3/sec/psi }
    Kce:=Kc + Ctp; { total flow-pressure coefficient, in^3/sec/psi }

    Co:=sqrt(beta/ro); { velocity of sound propogation in the oil }
    Lc:=ro/Ai; { line inductance per unit length }
    Zca:=Lc*Co; { characteristic impedance }
    Gr:=(Lt/2)/Co; { ratio }
    Rv:=1/Kce + 128*vis*Lt/(pi*Di*Di*Di*Di); { total system resistance }
    Rl:=128*vis*Lt/(pi*Di*Di*Di*Di);
    Rv:=Rl/(Kce*(Rl+1));
    Lm:=Mt/(Ap*Ap); { load inductance }
    alpa:=Lc*(Lt/2)/Lm; { the ratio of load inductance }

  end;

{------------------------------ MAIN PROGRAM -------------------------------}

BEGIN
  assign(f,'d:freq.out');
  rewrite(f);
  current:=5;  k:=0; check:=0;
  repeat
    k:=k+1;
    initialize_parameters;
    w := 0.1*2*pi;
    if w<1000 then fint := 20.0
              else fint := 50.0;
    for i:=1 to 100 do
      begin
        fr1 := cos(Gr*w);
        fr2 := (Zca/Rv)*sin(Gr*w) + (Gr*w/(2*alpa))*(1+Zca/Rv)*cos(Gr*w);
        frr := fr1*fr1 + fr2*fr2;
        frr := sqrt(frr);
        fr[i] := w/(2.0*pi);
        db[k,i] := 20.0*ln(Ki/Ap)/ln(10.0)-20.0*ln(frr)/ln(10.0);
        fss := fr2/fr1;
        theta[k,i] :=-arctan(fss);
        if (theta[k,i] > 0.0) then theta[k,i]:=-90.0+(theta[k,j]-90.0);
        w := w + fint;
      end;
    current:=current+5;
  until current>20;

  for i:=1 to 4 do
    begin
      st:=-1;
      for j:=2 to 100 do
        begin
```

```
          if abs(theta[i,j]-theta[i,j-1])>90 then
            begin
              st:=st+1;
              scale[i,j]:=st
            end
          else scale[i,j]:=st;
          if scale[i,j]<0 then scale[i,j]:=0;
        end;
      scale[i,i]:=0;
    end;
  for i:=1 to 4 do
    for j:=1 to 100 do
      theta[i,j]:=theta[i,j]-scale[i,j]*180;
  for j:=1 to 100 do
      writeln(f,' ',fr[j]:10:4,' ', db[1,j]:10:4,' ',db[4,j]:10:4,
                    theta[1,j]:10:2,' ',theta[4,j]:10:2);
  close(f);
end.
```

APPENDIX H

PULSE TESTING - WATERHAMMER EFFECT

```
(* This is a program to obatined the data for frequency analysis        *)
(* using pulse testing method while distributed parameter line          *)
(* modelling by method of characteristics being used.                   *)

PROGRAM DISTRIBUED_PARA_LINE_MODELLING_SYSTEM;

CONST
{ constants for simulation }
  n=6;
  h=1.0e-4;
  final_time=1.0;
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Va=5; { total volume of oil except line volume, in^3 }
  Kqi=0.0012; { valve current - spool displacement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

{ constant for simulating line }
  m = 8; { total divided node number nn= 2*m + 1 }
  dx = 25; { divided line length }
  d = 0.25; { inner tube diameter }
  nn = 17; { total divided node number }
  mu = 2.8e-6;

TYPE
  glnarray=array[1..n] of real;
  glmarray=array[1..nn] of real;

VAR
  i,j,k,it,data_no               :integer;
  hh,dt,
  Ff,time,Ap,Vp,Xp,Mt,
  QL1,QL2,QL3,QL4,Vt,Xv,Kc,
  current,scur,output_time,
  R,L,Co,
  Xmin,Xmax,Atube,tube_volume    :real;
  y,dydx,yout                    :glnarray;
  Qa,Qb,Pa,Pb                    :glmarray;
  tout,cout                      :array [1..300] of real;
  rout,pout1,pout2               :array [1..4,1..300] of real;
  f                              :text;

{---------------------------------------------------------------------------}
 procedure LineA;
{---------------------------------------------------------------------------}
```

```
var
  i,j : integer;

begin

  Pa[1] := y[1];
  Pa[2*m+1]:=y[2];

  for i:=1 to m do
    begin
      j:=i*2;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
             (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
            (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
            /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  Qa[1]:=((Co*L-R*dx/4)*Qa[2]+Pa[1]-Pa[2])/(Co*L+R*dx/4);
  Qa[2*m+1]:=((Co*L-R*dx/4)*Qa[2*m]-Pa[2*m+1]+Pa[2*m])/(Co*L+R*dx/4);
  QL1:=-Qa[1];
  QL2:=Qa[2*m+1];
end;

{--------------------------------------------------------------------------}
 procedure LineB;
{--------------------------------------------------------------------------}
var
  i,j : integer;

begin

  Pb[1] := y[3];
  Pb[2*m+1]:=y[4];

  for i:=1 to m do
    begin
      j:=i*2;
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
             (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
```

```
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
             (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
             /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  Qb[1]:=((Co*L-R*dx/4)*Qb[2]+Pb[1]-Pb[2])/(Co*L+R*dx/4);
  Qb[2*m+1]:=((Co*L-R*dx/4)*Qb[2*m]-Pb[2*m+1]+Pb[2*m])/(Co*L+R*dx/4);
  QL3:=-Qb[1];
  QL4:=Qb[2*m+1];
end;

{-----------------------------------------------------------------------------}
 Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{-----------------------------------------------------------------------------}
 var
   P1,P2,P3,P4,P5,P6,V7,X8,S5,
   Cd1,Cd2,Qd1,Qd2,Qd3,Qd4,
   change_time,CC1,CC2,CC3,Qc1,Qc2,FC3,
   FM1,
   DQ2,DQ3,DQ4,DQ5,DF7,DX8                              : real;

 begin
   P1:=Ps; P2:=Y[1]; P3:=Y[2]; P4:=y[3]; P5:=y[4]; P6:=0;
   V7:=Y[5]; X8:=Y[6];

  { Power source }

  { Supply Line }


  { Servo Valve }
    Xv:= current*Kqi;
    Cd1:=Cd; Cd2:=Cd;

    if current=0 then
      begin
        Qd1:=0;
        Qd2:=0;
        Qd3:=0;
        Qd4:=0;
      end
    else
      begin
        if current >0 then
          begin
            if P1 < P2 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P2-P1)/ro)
                       else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P2)/ro);
            if P5 < P6 then Qd3:=Cd2*Xv*Wd*sqrt(2*(P6-P5)/ro)
                       else Qd3:=-Cd2*Xv*Wd*sqrt(2*(P5-P6)/ro);
{
            if  (time < 0.002) then
              begin
                Qd1:=Qd1*(time)/0.002;
                Qd3:=Qd3*(time)/0.002;
```

```
               end;
}
             Qd2:=-Qd1;
             Qd4:=-Qd3;
           end
         else
           begin
             if P1 < P5 then Qd1:=Cd1*sqrt(P5-P1)
                        else Qd1:=-Cd1*sqrt(P1-P5);
             if P2 < P6 then Qd2:=Cd2*sqrt(P6-P2)
                        else Qd2:=-Cd2*sqrt(P2-P6);
             if (time < 2.002) then
               begin
                 Qd1:=Qd1*(time-0.2)/0.002;
                 Qd2:=Qd2*(time-0.2)/0.002;
               end;
             Qd3:=-Qd1;
             Qd4:=-Qd2;
           end;
       end;

  { Cylinder }
    CC1:=0.0; CC2:=0.0; Xmax:=10.0; Xmin:=0.0;
    Qc1:=-(Ap*V7+CC1*(P3-P4)+CC2*P3);
    Qc2:=Ap*V7+CC1*(P3-P4)-CC2*P4;
    FC3:=Ap*(P3-P4);

  { Load }
    if V7>=0 then FM1:=-Ff
             else FM1:= Ff;

  { Drain Line }

  DQ2:=Qd2+QL1;
  DQ3:=QL2+Qc1;
  DQ4:=Qc2+QL3;
  DQ5:=QL4+Qd3;
  DF7:=FC3+FM1;
  if (DF7>=0) and (X8>=Xmax) then DF7:=0.0;
  if (DF7<=0) and (X8<=Xmin) then DF7:=0.0;
  DX8:=V7;

  DYDX[1]:=(BETA/0.5)*DQ2;
  DYDX[2]:=(BETA/((Va/2-0.5)+X8*Ap))*DQ3;
  DYDX[3]:=(BETA/((Va/2-0.5)-X8*Ap))*DQ4;
  DYDX[4]:=(BETA/0.5)*DQ5;
  DYDX[5]:=DF7/Mt;
  DYDX[6]:=DX8;

END; { OF DERIVS }


{------------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
```

```
                VAR yout: glnarray);
{---------------------------------------------------------------------------}

VAR
   i         : integer;
   xh,h6     : real;
   dym,dyt,yt: glnarray;

BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;

   if (yout[5]>0) and (yout[6]>=Xmax) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmax;
     end;
   if (yout[5]<0) and (yout[6]<=Xmin) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmin;
     end;
   for j:=1 to 4 do
     if yout[j]<0 then yout[j]:=0;

END;

{---------------------------------------------------------------------------}
 Procedure initialize_parameters;
{---------------------------------------------------------------------------}
 var
   Kq,wh,dh,PL : real;

 begin
   time:=0.0;
   it:=0;
   data_no:=0;
```

```
   dt:=h;

   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
   Ff:=Mw*fc; { friction force, lbf }
   PL:=Ff/Ap; { load pressure, psig }
   Kq:=Cd*Wd*sqrt((Ps-PL)/ro); { flow gain, in^4/sec }
   Atube:=pi*d*d/4;
   tube_volume:=Atube*m*dx;
   Vt:=Va+tube_volume*2;
   R:=128*mu/(pi*d*d*d*d);
   L:=ro/Atube;
   Co:=dx/(dt/2);

   for i:=1 to nn do
     begin
       Qa[i]:=0;
       Qb[i]:=0;
       Pa[i]:=0;
       Pb[i]:=0;
     end;
   for i:=1 to n do
     begin
       y[i]:=0;
       DYDX[I]:=0.0;
     end;
 end;
{------------------------------ MAIN PROGRAM ------------------------------}

BEGIN
  assign(f,'d:CLINE1.out');
  rewrite(f);
  scur:=10.0; k:=1;
  initialize_parameters;
  repeat
    if time<0.002 then current:=scur*time/0.002;
    if (time>=0.002) and (time<0.004) then current:=scur;
    if (time>=0.004) and (time<0.006) then current:=3*scur*(1-time/0.006);
    if (time>=0.006) then current:=0;

    it:=it+1;
    LineA;LineB;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
    writeln(time:6:4,' ',current:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
    if it>=10 then
      begin
        data_no:=data_no+1;
        tout[data_no]:=time;
        cout[data_no]:=current;
        pout1[k,data_no]:=y[2];
```

```
        pout2[k,data_no]:=y[3];
        rout[k,data_no]:=y[5];
        it:=0;
      end;
   until (time>final_time) or (data_no>=300);
   for i:=1 to 300 do
     begin
       writeln(f,tout[i]:8:4,' ',cout[i]:10:4,' ',pout1[1,i]:10:4,' ',
                  pout2[k,i]:10:4,' ',rout[1,i]:10:4);
       writeln(tout[i]:8:4,' ',rout[1,i]:10:4);
     end;
   close(f);
end.
```

APPENDIX I

PULSE TESTING - FREQUENCY RESPONSE

```
(*   This is a program for drawing a BODE plot from reading external       *)
(*   date file obtained by pulse testing method.                           *)

PROGRAM BODE_PLOT;
USES CRT;
CONST
  h=1.0e-4;  { time step for numerical integration by SIMPSON'S 1/3 raw }

TYPE
  glnarray=array [1..700] of real;
  glcolumn=array [1..6] of real;

var
  i,j,k,m,n,col_no,inno,otno,count,start:integer;
  Kint,hhh,w,winc,maxw,aaa,bbb,ccc,ddd,nu1,nu2,num,den:real;
  column:glcolumn;
  wo: real;    { array of frequency }
  ph,gg: real; { array of phase angle and magnitue ratio }
  int,inp,ouq,af,bf,cf,df:glnarray;
  int_method:char;
  inf,outf,infile,outfile:string;
  title:array [1..12] of string;
  fi,fo:text;

{-------------------------------------------------------------------------}
 Function Integration(od,ti:glnarray; step:real; n:integer):real;
{-------------------------------------------------------------------------}
  var
    j,k : integer;
    area : real;

  begin
       area:=0.0;
  (* numerical integration by SIMPSON's 1/3 raw of constant time step *)
       if int_method in ['s','S'] then
         begin
           k:=0;
           for j:=1 to n do
             begin
               k:=k+1;
               if k=1 then area:=area+4*od[j];
               if k=2 then
                 begin
                   area:=area+2*od[j];
                   k:=0
                 end;
             end;
           area:=area*step/3.0;
         end;
  (* numerical integration for ADAPTIVE time step by TRAPEZOIDAL method *)
       if int_method in ['t','T'] then
         for j:=2 to n do
           area:=area+(od[j]+od[j-1])*(ti[j]-ti[j-1])/2;
       Integration:=area;
  end;
```

```
{--------------------------- MAIN OF BODE PLOT ------------------------------}

begin
   clrscr;
   writeln('   >>>> ENTER data file name for FREQUENCY ANALYSIS <<<<');
   write('             INPUT file name ? ==> '); readln(inf);
   infile:=inf;
   count:=length(inf);
   delete(inf,count-2,3);
   outfile:=inf+'.ff';
   writeln;
   assign(fi,infile);
   assign(fo,outfile);
   reset(fi);
   rewrite(fo);
   writeln(fo,'Plot Data File');
   writeln(fo,'3');
   writeln(fo,'  Freq');
   writeln(fo,'  Amp');
   writeln(fo,'  Phase');
   writeln(fo);
   writeln(fo,'     Freq        Amp         Phase');
   int_method:='t';
   readln(fi);
   readln(fi,col_no);
   for i:=1 to col_no do
     readln(fi,title[i]);
   for i:=1 to 2 do
     readln(fi);
   writeln('      >> Select data column for FREQUENCY ANALYSIS <<');
   writeln('         (total column number in ',infile,' = ',col_no:2,' )');
   for i:=1 to col_no do
   writeln('                    column[',i:1,'] = ',title[i]); writeln;
   write('               INPUT PULSE COLUMN ? ==> '); readln(inno);
   write('               OUTPUT DATA COLUMN ? ==> '); readln(otno);
   writeln;
   writeln('      >> Enter fequency STEP size & analysis RANGE <<');
   write('            frequency step (rad/sec) ? ==> '); readln(winc);
   write('            maximum frequency (Hz)   ? ==> '); readln(maxw);
   maxw:=maxw*2*pi;
   m:=trunc(maxw/winc);   { total iteration number for frequency analysis }
   i:=0;
   repeat
     i:=i+1;
     for j:=1 to col_no do
       read(fi,column[j]);
     readln(fi);
     int[i]:=column[1];
     inp[i]:=column[inno];
     ouq[i]:=column[otno];
   until EOF(fi);
   close(fi);
   n:=i;
```

```
  hhh:=h;
  if ouq[n]<>0.0 then
    begin
      Kint:=ouq[n]/Integration(inp,int,hhh,n);
      for i:=2 to n do
        ouq[i]:=ouq[i]-Kint*Integration(inp,int,hhh,i);
    end
  else Kint:=0;

  w:=0.2*pi;
  for i:=1 to m do
    begin
      wo:=w/(2.0*pi);
      for j:=1 to n do
      begin
        af[j]:=ouq[j]*cos(w*int[j]);
        bf[j]:=ouq[j]*sin(w*int[j]);
        cf[j]:=inp[j]*cos(w*int[j]);
        df[j]:=inp[j]*sin(w*int[j]);
      end;

      aaa:=Integration(af,int,hhh,n);
      bbb:=Integration(bf,int,hhh,n);
      ccc:=Integration(cf,int,hhh,n);
      ddd:=Integration(df,int,hhh,n);

      den:=ccc*ccc+ddd*ddd;
      nu1:=(aaa*ccc+bbb*ddd)/den;
      nu2:=(aaa*ddd-bbb*ccc)/den-Kint/w;
      num:=nu1*nu1+nu2*nu2;
      num:=sqrt(num);
      gg:=num;
      gg:=20.0*ln(gg)/ln(10.0);
      ph:=arctan(nu2/nu1)*180.0/pi;
      if (ph>0.0) then ph:=-90.0+(ph-90.0);
      writeln(fo,wo:10:4,' ',gg:10:4,' ',ph:10:4);
      w:=w+winc;
    end;
 close(fo);
end.
```

APPENDIX J

ORTHOGONAL PROJECTION ALGORITHM

```
PROGRAM SECOND_ORDER_ORTHOGONAL_PROJECTION_ALGORITHM;

var
  i,j,k,L:integer;
  sum,Down,Rig:real;
  TH,TO,PAO,PA,UpM,DoM: array [1..3] of real;
  y : array [-3..100] of real;
  t,u : array [-2..100] of real;
  P,AP,BP : array [1..3,1..3] of real;
  out:array[1..100,1..3] of real;
  OuM:array[1..3,1..100] of real;
  fi,fo:text;

begin

  assign(fo,'d:oth2.out');
  assign(fi,'d:cline1.out');
  rewrite(fo);
  reset(fi);

  for i:=0 to 150 do
    begin
      read(fi,t[i]);
      read(fi,y[i]);
      readln(fi);
      u[i]:=5;
    end;
  close(fi);

  for i:=1 to 3 do
    begin
      for j:=1 to 3 do
        P[i,j]:=0;
      P[i,i]:=1
    end;

  TH[1]:=-2; Th[2]:=0.93; Th[3]:=0.03;
  k:=1;

  repeat

    writeln(fo,'                  ',k:4,' ',TH[1]:12:4,' ',TH[2]:12:4,' ',TH[3]:1
2:4);

    for i:=1 to 3 do
      out[k,i]:=th[i];

    k:=k+1;
    PA[1]:=-y[k-1]; PA[2]:=-y[k-2]; PA[3]:=u[k-1];

    for i:=1 to 3 do
      begin
        sum:=0;
        for j:=1 to 3 do
          sum:=sum+P[i,j]*PA[j];
        UpM[i]:=sum;
```

```
      sum:=0;
      for j:=1 to 3 do
        sum:=sum+PA[j]*P[j,i];
      DoM[i]:=sum;
    end;
  sum:=0;
  for i:=1 to 3 do
    sum:=sum+DoM[i]*PA[i];
  Down:=sum;
  sum:=0;
  for i:=1 to 3 do
    sum:=sum+PA[i]*TH[i];
  Rig:=y[k]-sum;

  for i:=1 to 3 do
    for j:=1 to 3 do
      AP[i,j]:=UpM[i]*PA[j];

  for i:=1 to 3 do
    for j:=1 to 3 do
      begin
        sum:=0;
        for L:=1 to 3 do
          sum:=sum+AP[i,L]*P[L,j];
      BP[i,j]:=sum;
      end;
  for i:=1 to 3 do
    begin
      TH[i]:=TH[i]+UpM[i]*Rig/Down;
      for j:=1 to 3 do
        P[i,j]:=P[i,j]-BP[i,j]/Down;
    end;
 until k>=150;
 close(fo);
end.
```

APPENDIX K

EXAMPLE OF PARAMETER ESTIMATION

TABLE IX

PARAMETERS ESTIMATION
BY ORTHOGONAL PROJECTION ALGORITHM
(10" LINE, 5 mA)

NO	L1	L2	U1
1	-2.0000	0.9300	-0.0030
2	-2.0008	0.9300	0.0385
3	-1.6467	1.0486	0.0454
4	-1.6311	1.0020	0.0457
5	-1.5760	0.9326	0.0486
6	-1.5495	0.9041	0.0500
7	-1.4760	0.8683	0.0605
8	-1.4755	0.8945	0.0671
9	-1.4743	0.8978	0.0682
10	-1.4744	0.8976	0.0681
11	-1.4736	0.8985	0.0686
12	-1.4726	0.8993	0.0690
13	-1.4696	0.9009	0.0701
14	-1.4607	0.9042	0.0730
15	-1.4371	0.9103	0.0802
16	-1.4268	0.9162	0.0841
17	-1.4385	0.8975	0.0767
18	-1.4465	0.8835	0.0713
19	-1.4484	0.8808	0.0702
20	-1.4476	0.8817	0.0706
21	-1.4454	0.8835	0.0715
22	-1.4414	0.8863	0.0732
23	-1.4346	0.8900	0.0757
24	-1.4264	0.8941	0.0787
25	-1.4237	0.8959	0.0798
26	-1.4291	0.8904	0.0771
27	-1.4357	0.8821	0.0735
28	-1.4388	0.8783	0.0718
29	-1.4390	0.8781	0.0717
30	-1.4377	0.8792	0.0723
31	-1.4347	0.8816	0.0736
32	-1.4310	0.8840	0.0751
33	-1.4265	0.8869	0.0769
34	-1.4249	0.8880	0.0776
35	-1.4272	0.8860	0.0765
36	-1.4312	0.8818	0.0745
37	-1.4340	0.8788	0.0731
38	-1.4350	0.8777	0.0726
39	-1.4343	0.8784	0.0729
40	-1.4328	0.8797	0.0736

TABLE IX (Continued)

NO	L1	L2	U1
41	-1.4302	0.8816	0.0747
42	-1.4277	0.8834	0.0757
43	-1.4264	0.8844	0.0763
44	-1.4273	0.8837	0.0759
45	-1.4294	0.8817	0.0749
46	-1.4316	0.8795	0.0738
47	-1.4326	0.8785	0.0734
48	-1.4326	0.8785	0.0734
49	-1.4318	0.8792	0.0737
50	-1.4302	0.8805	0.0744
51	-1.4287	0.8817	0.0751
52	-1.4276	0.8825	0.0755
53	-1.4278	0.8824	0.0755
54	-1.4289	0.8813	0.0749
55	-1.4303	0.8800	0.0743
56	-1.4312	0.8792	0.0739
57	-1.4315	0.8789	0.0737
58	-1.4311	0.8793	0.0739
59	-1.4303	0.8800	0.0743
60	-1.4291	0.8809	0.0748
61	-1.4286	0.8813	0.0750
62	-1.4283	0.8816	0.0752
63	-1.4288	0.8811	0.0749
64	-1.4298	0.8803	0.0745
65	-1.4305	0.8796	0.0741
66	-1.4307	0.8794	0.0741
67	-1.4307	0.8795	0.0741
68	-1.4302	0.8798	0.0743
69	-1.4294	0.8805	0.0746
70	-1.4291	0.8808	0.0748
71	-1.4287	0.8811	0.0749
72	-1.4291	0.8808	0.0748
73	-1.4295	0.8804	0.0746
74	-1.4299	0.8801	0.0744
75	-1.4304	0.8797	0.0742
76	-1.4303	0.8797	0.0742
77	-1.4300	0.8799	0.0743
78	-1.4296	0.8803	0.0745
79	-1.4294	0.8805	0.0746
80	-1.4291	0.8808	0.0748

TABLE IX (Continued)

NO	L1	L2	U1
81	-1.4292	0.8807	0.0747
82	-1.4295	0.8803	0.0746
83	-1.4296	0.8803	0.0745
84	-1.4301	0.8798	0.0743
85	-1.4300	0.8800	0.0744
86	-1.4301	0.8799	0.0743
87	-1.4296	0.8803	0.0745
88	-1.4296	0.8803	0.0745
89	-1.4294	0.8805	0.0746
90	-1.4293	0.8805	0.0747
91	-1.4295	0.8803	0.0746
92	-1.4295	0.8804	0.0746
93	-1.4299	0.8800	0.0744
94	-1.4299	0.8800	0.0744
95	-1.4298	0.8801	0.0744
96	-1.4297	0.8801	0.0745
97	-1.4296	0.8803	0.0745
98	-1.4295	0.8804	0.0746
99	-1.4294	0.8804	0.0746

APPENDIX L

EXAMPLE OF OPTIMIZATION FOR PID (VELOCITY CONTROL)

TABLE X
OPTIMIZATION FOR PID GAINS
(10" LINE, 5 mA CURRENT'S STEADY-STATE VALUE)

No	Kp	Ki	Kd	ISE
1	1.1000	0.1300	30.0000	2.7727563761E-03
2	1.1000	0.1300	30.0000	2.7727563761E-03
3	1.1000	0.1300	31.0000	2.7644126325E-03
4	1.1000	0.1300	32.6180	2.7535139238E-03
5	1.1000	0.1300	37.2925	2.7404273309E-03
6	1.1000	0.1300	44.8559	3.4209190558E-03
7	1.1000	0.1300	37.2925	2.7404273309E-03
8	1.1000	0.1300	40.1814	2.7864701715E-03
9	1.1000	0.1300	35.5070	2.7412678230E-03
10	1.1000	0.1300	36.4668	2.7397966485E-03
11	1.1000	0.1300	36.5827	2.7397572887E-03
12	1.1000	0.1300	36.6340	2.7397514658E-03
13	1.1000	0.1300	36.6504	2.7397511556E-03
14	1.1000	0.1300	36.6489	2.7397511521E-03
15	1.1000	0.1300	36.6482	2.7397511526E-03
16	1.1000	0.1300	36.6496	2.7397511529E-03
17	1.1000	0.1300	36.6489	2.7397511521E-03
18	2.1000	0.1300	36.6489	2.8456321698E-03
19	-0.5180	0.1300	36.6489	2.7997480593E-03
20	1.1000	0.1300	36.6489	2.7397511521E-03
21	0.4820	0.1300	36.6489	2.7139611267E-03
22	0.1000	0.1300	36.6489	2.7222949014E-03
23	0.4627	0.1300	36.6489	2.7138590462E-03
24	0.4370	0.1300	36.6489	2.7138015084E-03
25	0.4333	0.1300	36.6489	2.7138006869E-03
26	0.4335	0.1300	36.6489	2.7138006831E-03
27	0.4335	0.1300	36.6489	2.7138006832E-03
28	0.4334	0.1300	36.6489	2.7138006835E-03
29	0.4335	0.1300	36.6489	2.7138006831E-03
30	0.4335	1.1300	36.6489	2.4891528327E+04
31	0.4335	-1.4880	36.6489	6.8295248314E-02
32	0.4335	0.1300	36.6489	2.7138006831E-03
33	0.4335	-0.4880	36.6489	1.8157126429E-02
34	0.4335	0.5120	36.6489	5.9562475014E+03
35	0.4335	-0.1790	36.6489	1.0553449353E+02
36	0.4335	0.2759	36.6489	8.2009167091E-02
37	0.4335	0.0120	36.6489	2.0729232185E-02
38	0.4335	0.1857	36.6489	2.9687997424E-03
39	0.4335	0.1057	36.6489	2.9510839135E-03
40	0.4335	0.1451	36.6489	2.6666214940E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
41	0.4335	0.1468	36.6489	2.6658631243E-03
42	0.4335	0.1474	36.6489	2.6658181952E-03
43	0.4335	0.1473	36.6489	2.6658179694E-03
44	0.4335	0.1473	36.6489	2.6658179691E-03
45	0.4335	0.1473	36.6489	2.6658179699E-03
46	-0.2330	0.1646	43.2978	3.0685740300E-03
47	0.4335	0.1473	36.6489	2.6658179691E-03
48	0.4335	0.1473	37.6489	2.6572320881E-03
49	0.4335	0.1473	39.2669	2.6569212737E-03
50	0.4335	0.1473	38.4879	2.6541876135E-03
51	0.4335	0.1473	38.4879	2.6541876135E-03
52	0.4335	0.1473	38.1674	2.6547732314E-03
53	0.4335	0.1473	38.7855	2.6544440590E-03
54	0.4335	0.1473	38.5376	2.6541726954E-03
55	0.4335	0.1473	38.5447	2.6541723645E-03
56	0.4335	0.1473	38.5464	2.6541723528E-03
57	0.4335	0.1473	38.5462	2.6541723528E-03
58	0.4335	0.1473	38.5466	2.6541723530E-03
59	0.4335	0.1473	38.5464	2.6541723528E-03
60	1.4335	0.1473	38.5464	2.6351118462E-03
61	3.0515	0.1473	38.5464	2.7795475974E-03
62	1.4335	0.1473	38.5464	2.6351118462E-03
63	2.0515	0.1473	38.5464	2.6726490071E-03
64	1.0515	0.1473	38.5464	2.6277819527E-03
65	1.0116	0.1473	38.5464	2.6278981789E-03
66	1.0593	0.1473	38.5464	2.6277806724E-03
67	1.0568	0.1473	38.5464	2.6277803332E-03
68	1.0569	0.1473	38.5464	2.6277803333E-03
69	1.0568	0.1473	38.5464	2.6277803335E-03
70	1.0568	0.1473	38.5464	2.6277803332E-03
71	1.0568	1.1473	38.5464	2.3947979365E+04
72	1.0568	-1.4707	38.5464	6.8293320919E-02
73	1.0568	0.1473	38.5464	2.6277803332E-03
74	1.0568	-0.4707	38.5464	1.6161892671E-02
75	1.0568	0.5293	38.5464	5.9915032437E+03
76	1.0568	-0.1617	38.5464	8.6907560479E+01
77	1.0568	0.2932	38.5464	2.8428991094E-02
78	1.0568	0.0293	38.5464	8.9145469444E-03
79	1.0568	0.2031	38.5464	2.7921731106E-03
80	1.0568	0.1706	38.5464	2.5890147204E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
81	1.0568	0.1648	38.5464	2.5861961656E-03
82	1.0568	0.1657	38.5464	2.5860716473E-03
83	1.0568	0.1658	38.5464	2.5860710233E-03
84	1.0568	0.1658	38.5464	2.5860710195E-03
85	1.0568	0.1658	38.5464	2.5860710199E-03
86	1.0568	0.1658	38.5464	2.5860710200E-03
87	1.6802	0.1843	40.4439	2.5521880611E-03
88	1.0568	0.1658	38.5464	2.5860710195E-03
89	1.6802	0.1843	40.4439	2.5521880611E-03
90	2.6888	0.2142	43.5142	2.8023739009E-03
91	1.6802	0.1843	40.4439	2.5521880611E-03
92	2.0655	0.1957	41.6167	2.5872954031E-03
93	1.4421	0.1773	39.7192	2.5562490306E-03
94	1.6103	0.1822	40.2311	2.5518232900E-03
95	1.6255	0.1827	40.2776	2.5517799331E-03
96	1.6275	0.1828	40.2837	2.5517791834E-03
97	1.6279	0.1828	40.2846	2.5517791715E-03
98	1.6278	0.1828	40.2844	2.5517791717E-03
99	1.6279	0.1828	40.2848	2.5517791721E-03
100	1.6279	0.1828	40.2846	2.5517791715E-03
101	1.6279	0.1828	41.2846	2.5976324782E-03
102	1.6279	0.1828	38.6666	2.5351567978E-03
103	1.6279	0.1828	39.0976	2.5351689661E-03
104	1.6279	0.1828	36.0485	2.5670928089E-03
105	1.6279	0.1828	38.6666	2.5351567978E-03
106	1.6279	0.1828	37.6666	2.5423341449E-03
107	1.6279	0.1828	39.2846	2.5359685920E-03
108	1.6279	0.1828	38.8504	2.5348768910E-03
109	1.6279	0.1828	38.8750	2.5348701377E-03
110	1.6279	0.1828	38.8857	2.5348695637E-03
111	1.6279	0.1828	38.8847	2.5348695571E-03
112	1.6279	0.1828	38.8845	2.5348695572E-03
113	1.6279	0.1828	38.8848	2.5348695573E-03
114	1.6279	0.1828	38.8847	2.5348695571E-03
115	2.6279	0.1828	38.8847	2.4989602424E-03
116	4.2459	0.1828	38.8847	2.5897389595E-03
117	2.6279	0.1828	38.8847	2.4989602424E-03
118	3.2459	0.1828	38.8847	2.5186616524E-03
119	2.2459	0.1828	38.8847	2.5004100904E-03
120	2.4901	0.1828	38.8847	2.4980684794E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
121	2.4820	0.1828	38.8847	2.4980630755E-03
122	2.4776	0.1828	38.8847	2.4980623866E-03
123	2.4779	0.1828	38.8847	2.4980623832E-03
124	2.4780	0.1828	38.8847	2.4980623835E-03
125	2.4778	0.1828	38.8847	2.4980623835E-03
126	2.4779	0.1828	38.8847	2.4980623832E-03
127	3.0489	0.1997	40.6228	2.4964918218E-03
128	3.9728	0.2271	43.4353	2.7802620009E-03
129	3.0489	0.1997	40.6228	2.4964918218E-03
130	3.4018	0.2102	41.6971	2.5534377096E-03
131	2.8308	0.1932	39.9589	2.4868286289E-03
132	2.8318	0.1933	39.9619	2.4868379987E-03
133	2.8020	0.1924	39.8711	2.4866773430E-03
134	2.6782	0.1887	39.4943	2.4884507144E-03
135	2.7977	0.1923	39.8580	2.4866743616E-03
136	2.7973	0.1922	39.8569	2.4866743382E-03
137	2.7972	0.1922	39.8568	2.4866743378E-03
138	2.7972	0.1922	39.8567	2.4866743380E-03
139	4.5377	0.2187	41.1671	2.5148991722E-03
140	2.7972	0.1922	39.8568	2.4866743378E-03
141	4.5377	0.2187	41.1671	2.5148991722E-03
142	-0.0188	0.1495	37.7366	2.7354882640E-03
143	2.7972	0.1922	39.8568	2.4866743378E-03
144	1.7216	0.1759	39.0469	2.5320255011E-03
145	3.4620	0.2023	40.3573	2.4818592965E-03
146	3.3101	0.2000	40.2429	2.4815464085E-03
147	3.3293	0.2003	40.2573	2.4815397301E-03
148	3.3293	0.2003	40.2574	2.4815397300E-03
149	3.3294	0.2003	40.2574	2.4815397301E-03
150	3.3293	0.2003	40.2574	2.4815397300E-03
151	3.3293	0.2003	41.2574	2.5446963759E-03
152	3.3293	0.2003	38.6393	2.4475497040E-03
153	3.3293	0.2003	38.7959	2.4486454035E-03
154	3.3293	0.2003	36.0213	2.4581482106E-03
155	3.3293	0.2003	38.6393	2.4475497040E-03
156	3.3293	0.2003	37.6393	2.4466179584E-03
157	3.3293	0.2003	37.0213	2.4496552794E-03
158	3.3293	0.2003	38.0213	2.4459364818E-03
159	3.3293	0.2003	38.0333	2.4459331412E-03
160	3.3293	0.2003	38.0637	2.4459300779E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
161	3.3293	0.2003	38.0606	2.4459300349E-03
162	3.3293	0.2003	38.0604	2.4459300349E-03
163	3.3293	0.2003	38.0601	2.4459300353E-03
164	3.3293	0.2003	38.0604	2.4459300349E-03
165	4.3293	0.2003	38.0604	2.4766715824E-03
166	1.7113	0.2003	38.0604	2.5750660787E-03
167	3.3293	0.2003	38.0604	2.4459300349E-03
168	2.7113	0.2003	38.0604	2.4588096061E-03
169	3.7113	0.2003	38.0604	2.4515335223E-03
170	3.3138	0.2003	38.0604	2.4458937943E-03
171	3.2841	0.2003	38.0604	2.4458700864E-03
172	3.2871	0.2003	38.0604	2.4458697622E-03
173	3.2872	0.2003	38.0604	2.4458697619E-03
174	3.2872	0.2003	38.0604	2.4458697619E-03
175	3.2872	0.2003	38.0604	2.4458697619E-03
176	3.2872	0.2003	38.0604	2.4458697619E-03
177	3.2872	0.2003	38.0604	2.4458697619E-03
178	3.8193	0.2084	38.4610	2.4279589409E-03
179	4.6802	0.2215	39.1092	2.4119227773E-03
180	5.1127	0.2280	39.4347	2.4101935285E-03
181	5.8124	0.2386	39.9615	2.4168052683E-03
182	5.1127	0.2280	39.4347	2.4101935285E-03
183	5.3799	0.2321	39.6360	2.4111972159E-03
184	4.9475	0.2255	39.3104	2.4103706821E-03
185	5.0781	0.2275	39.4087	2.4101814879E-03
186	5.0794	0.2275	39.4097	2.4101814687E-03
187	5.0796	0.2275	39.4098	2.4101814688E-03
188	5.0792	0.2275	39.4095	2.4101814693E-03
189	7.3615	0.2628	38.9626	2.3702375521E-03
190	5.0794	0.2275	39.4097	2.4101814687E-03
191	7.3615	0.2628	38.9626	2.3702375521E-03
192	11.0541	0.3199	38.2392	2.3405515908E-03
193	11.7456	0.3306	38.1037	2.3383304534E-03
194	12.8645	0.3479	37.8845	2.3368032599E-03
195	12.9740	0.3496	37.8631	2.3368475184E-03
196	12.8645	0.3479	37.8845	2.3368032599E-03
197	12.4371	0.3413	37.9683	2.3369491668E-03
198	12.7013	0.3453	37.9165	2.3367945140E-03
199	12.7650	0.3463	37.9040	2.3367898385E-03
200	12.7619	0.3463	37.9046	2.3367898231E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
201	12.7611	0.3463	37.9048	2.3367898225E-03
202	12.7604	0.3463	37.9049	2.3367898229E-03
203	12.7611	0.3463	37.9048	2.3367898225E-03
204	12.7611	0.3463	38.9048	2.3492313452E-03
205	12.7611	0.3463	36.2867	2.3412819384E-03
206	12.7611	0.3463	37.9048	2.3367898225E-03
207	12.7611	0.3463	37.2867	2.3362439559E-03
208	12.7611	0.3463	36.9048	2.3374872230E-03
209	12.7611	0.3463	37.5228	2.3360215372E-03
210	12.7611	0.3463	37.5034	2.3360217522E-03
211	12.7611	0.3463	37.5144	2.3360212078E-03
212	12.7611	0.3463	37.5143	2.3360212078E-03
213	12.7611	0.3463	37.5144	2.3360212079E-03
214	12.7611	0.3463	37.5144	2.3360212078E-03
215	13.7611	0.3463	37.5144	2.3934791111E-03
216	11.1431	0.3463	37.5144	2.2667730135E-03
217	8.1307	0.3463	37.5144	2.2788642317E-03
218	11.1431	0.3463	37.5144	2.2667730135E-03
219	9.9925	0.3463	37.5144	2.2420058123E-03
220	9.2813	0.3463	37.5144	2.2417083202E-03
221	9.6184	0.3463	37.5144	2.2401036890E-03
222	9.6218	0.3463	37.5144	2.2401044353E-03
223	9.6125	0.3463	37.5144	2.2401031287E-03
224	9.6121	0.3463	37.5144	2.2401031265E-03
225	9.6118	0.3463	37.5144	2.2401031280E-03
226	9.6121	0.3463	37.5144	2.2401031265E-03
227	17.2938	0.4650	36.0095	6.3784569894E+03
228	-2.8173	0.1541	39.9493	8.9416846765E-02
229	9.6121	0.3463	37.5144	2.2401031265E-03
230	4.8645	0.2729	38.4444	2.3622939895E-03
231	12.5462	0.3916	36.9396	2.2339119895E-03
232	11.4222	0.3743	37.1598	2.2326107322E-03
233	11.6636	0.3780	37.1125	2.2325073180E-03
234	11.6621	0.3780	37.1128	2.2325073446E-03
235	11.6680	0.3781	37.1116	2.2325072843E-03
236	11.6682	0.3781	37.1116	2.2325072844E-03
237	11.6678	0.3781	37.1116	2.2325072845E-03
238	18.2566	0.5286	34.8135	6.4133288587E+03
239	11.6680	0.3781	37.1116	2.2325072843E-03
240	11.6680	0.3781	38.1116	2.2212383732E-03
241	11.6680	0.3781	39.7296	2.2319481078E-03
242	11.6680	0.3781	38.1116	2.2212383732E-03
243	11.6680	0.3781	38.7296	2.2194543153E-03
244	11.6680	0.3781	39.1116	2.2213944051E-03
245	11.6680	0.3781	38.4936	2.2195215358E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
246	11.6680	0.3781	38.6325	2.2193783676E-03
247	11.6680	0.3781	38.6302	2.2193783847E-03
248	11.6680	0.3781	38.6318	2.2193783637E-03
249	11.6680	0.3781	38.6319	2.2193783638E-03
250	11.6680	0.3781	38.6316	2.2193783639E-03
251	11.6680	0.3781	38.6318	2.2193783637E-03
252	12.6680	0.3781	38.6318	2.2471672838E-03
253	10.0499	0.3781	38.6318	2.2159923077E-03
254	10.7524	0.3781	38.6318	2.2096105533E-03
255	10.7524	0.3781	38.6318	2.2096105533E-03
256	11.1021	0.3781	38.6318	2.2112282011E-03
257	10.4841	0.3781	38.6318	2.2104324842E-03
258	10.7413	0.3781	38.6318	2.2096077567E-03
259	10.7370	0.3781	38.6318	2.2096074951E-03
260	10.7368	0.3781	38.6318	2.2096074943E-03
261	10.7367	0.3781	38.6318	2.2096074945E-03
262	10.7369	0.3781	38.6318	2.2096074945E-03
263	10.7368	0.3781	38.6318	2.2096074943E-03
264	18.4185	0.4968	37.1269	6.4148791749E+03
265	-1.6926	0.1859	41.0668	1.6674147865E-02
266	10.7368	0.3781	38.6318	2.2096074943E-03
267	5.9892	0.3047	39.5619	2.3192739107E-03
268	13.6709	0.4234	38.0570	2.2078668303E-03
269	12.3051	0.4023	38.3245	2.2032533983E-03
270	12.3210	0.4025	38.3214	2.2032442198E-03
271	12.4115	0.4039	38.3037	2.2032172341E-03
272	12.8926	0.4114	38.2095	2.2037862372E-03
273	12.4232	0.4041	38.3014	2.2032168801E-03
274	12.4230	0.4041	38.3014	2.2032168802E-03
275	12.4233	0.4041	38.3014	2.2032168801E-03
276	13.1783	0.4302	39.4912	2.1895319659E-03
277	12.4232	0.4041	38.3014	2.2032168801E-03
278	13.1783	0.4302	39.4912	2.1895319659E-03
279	14.4002	0.4724	41.4164	2.2918833147E-03
280	13.1783	0.4302	39.4912	2.1895319659E-03
281	13.6450	0.4463	40.2266	2.1984657568E-03
282	12.8899	0.4202	39.0368	2.1919748756E-03
283	13.1499	0.4292	39.4465	2.1895687904E-03
284	13.1876	0.4305	39.5059	2.1895309916E-03
285	13.1846	0.4304	39.5012	2.1895306989E-03
286	13.1845	0.4304	39.5010	2.1895306990E-03
287	13.1847	0.4304	39.5013	2.1895306991E-03
288	13.1846	0.4304	39.5012	2.1895306989E-03
289	13.1846	0.4304	40.5012	2.2154291684E-03
290	13.1846	0.4304	37.8831	2.2028173305E-03

TABLE X (Continued)

No	Kp	Ki	Kd	ISE
291	13.1846	0.4304	39.5012	2.1895306989E-03
292	13.1846	0.4304	38.8831	2.1894125471E-03
293	13.1846	0.4304	38.5012	2.1929495441E-03
294	13.1846	0.4304	39.1192	2.1884788064E-03
295	13.1846	0.4304	39.1833	2.1884178594E-03
296	13.1846	0.4304	39.1987	2.1884164433E-03
297	13.1846	0.4304	39.1953	2.1884163083E-03
298	13.1846	0.4304	39.1952	2.1884163082E-03
299	13.1846	0.4304	39.1953	2.1884163082E-03
300	13.1846	0.4304	39.1953	2.1884163082E-03
301	14.1846	0.4304	39.1953	2.2119652300E-03
302	11.5666	0.4304	39.1953	2.2039639831E-03
303	13.1846	0.4304	39.1953	2.1884163082E-03
304	12.5666	0.4304	39.1953	2.1862464481E-03
305	12.1846	0.4304	39.1953	2.1896096948E-03
306	12.8027	0.4304	39.1953	2.1860382523E-03
307	12.7230	0.4304	39.1953	2.1859595392E-03
308	12.7215	0.4304	39.1953	2.1859594873E-03
309	12.7207	0.4304	39.1953	2.1859594803E-03
310	12.7208	0.4304	39.1953	2.1859594804E-03
311	12.7207	0.4304	39.1953	2.1859594804E-03
312	12.7207	0.4304	39.1953	2.1859594803E-03
313	13.4822	0.4567	40.3950	2.1941422164E-03
314	11.4886	0.3879	37.2540	2.2229281268E-03
315	12.7207	0.4304	39.1953	2.1859594803E-03
316	12.2501	0.4142	38.4538	2.1960285901E-03
317	13.0116	0.4405	39.6535	2.1842592693E-03
318	13.0093	0.4404	39.6499	2.1842543740E-03
319	12.9720	0.4391	39.5911	2.1842205025E-03
320	12.8760	0.4358	39.4399	2.1845035067E-03
321	12.9759	0.4392	39.5972	2.1842201013E-03
322	12.9756	0.4392	39.5968	2.1842200998E-03
323	12.9756	0.4392	39.5969	2.1842200998E-03
324	12.9756	0.4392	39.5968	2.1842200999E-03

APPENDIX M

PID CONTROL

```
(* This is a program for the dynamic response of the PID FEEDBACK        *)
(* system while distributed parameter line modelling by explicit         *)
(* method of characteristics being used.                                  *)

PROGRAM DISTRIBUED_PARA_LINE_MODELLING_SYSTEM;

CONST
{ constants for simulation }
  n=6;
  h=1.0e-4;
  final_time=1.0;
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Va=5; { total volume of oil, in^3 }
  Kqi=0.0012; { valve current - spool displacement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

{ constant for simulating line }
  m = 2; { total divided node number nn= 2*m + 1 }
  dx = 5; { divided line length }
  d = 0.25; { inner tube diameter }
  nn = 5; { total divided node number }
  mu = 2.8e-6;

{ PID controller gain }
  Kpro=12.9756;
  Kint=0.4392;
  Kdrv=39.5968;

TYPE
  glnarray=array[1..n] of real;
  glmarray=array[1..nn] of real;

VAR
  i,j,k,it,data_no                :integer;
  hh,dt,
  Ff,time,Ap,Vp,Xp,Mt,
  QL1,QL2,QL3,QL4,Vt,Xv,Kc,
  current,output_time,
  R,L,Co,
  Xmin,Xmax,Atube,tube_volume,
  Ms,Cn,Cn_1,Cn_2,
  err,area,forv                   :real;
  y,dydx,yout                     :glnarray;
  Qa,Qb,Pa,Pb                     :glmarray;
```

```
  tout,cout,rout                   :array [1..200] of real;
  f                                :text;

{----------------------------------------------------------------------------}
 procedure PID;
{----------------------------------------------------------------------------}
 const
   Rn=0.8267;

 var
   dMs:real;

 begin
   dMs:=Kpro*(Cn_1-Cn)+Kint*(Rn-Cn)+Kdrv*(2*Cn_1-Cn_2-Cn);
   current:=Ms+dMs;
   if current>20 then current:=20;
   if current<-20 then current:=-20;
   Ms:=current;
   Cn_2:=Cn_1;
   Cn_1:=Cn;
 end;


{----------------------------------------------------------------------------}
 procedure LineA;
{----------------------------------------------------------------------------}
var
  i,j : integer;

begin

  Pa[1] := y[1];
  Pa[2*m+1]:=y[2];

  for i:=1 to m do
    begin
      j:=i*2;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
             (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
             (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
             /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  Qa[1]:=((Co*L-R*dx/4)*Qa[2]+Pa[1]-Pa[2])/(Co*L+R*dx/4);
  Qa[2*m+1]:=((Co*L-R*dx/4)*Qa[2*m]-Pa[2*m+1]+Pa[2*m])/(Co*L+R*dx/4);
  QL1:=-Qa[1];
```

```
  QL2:=Qa[2*m+1];
end;

{-----------------------------------------------------------------------------}
 procedure LineB;
{-----------------------------------------------------------------------------}
var
  i,j : integer;

begin

  Pb[1] := y[3];
  Pb[2*m+1]:=y[4];

  for i:=1 to m do
    begin
      j:=i*2;
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
             (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
            (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
            /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  Qb[1]:=((Co*L-R*dx/4)*Qb[2]+Pb[1]-Pb[2])/(Co*L+R*dx/4);
  Qb[2*m+1]:=((Co*L-R*dx/4)*Qb[2*m]-Pb[2*m+1]+Pb[2*m])/(Co*L+R*dx/4);
  QL3:=-Qb[1];
  QL4:=Qb[2*m+1];
end;

{-----------------------------------------------------------------------------}
 Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{-----------------------------------------------------------------------------}
 var
   P1,P2,P3,P4,P5,P6,V7,X8,S5,
   Cd1,Cd2,Qd1,Qd2,Qd3,Qd4,
   change_time,CC1,CC2,CC3,Qc1,Qc2,FC3,
   FM1,
   DQ2,DQ3,DQ4,DQ5,DF7,DX8                          : real;

 begin
   P1:=Ps; P2:=Y[1]; P3:=Y[2]; P4:=y[3]; P5:=y[4]; P6:=0;
   V7:=Y[5]; X8:=Y[6];

  { Power source }

  { Supply Line }
```

```
{ Signal }

{ Servo Valve }
  Xv:= current*Kqi;
  Cd1:=Cd; Cd2:=Cd;

  if current=0 then
    begin
      Qd1:=0;
      Qd2:=0;
      Qd3:=0;
      Qd4:=0;
    end
  else
    begin
      if current >0 then
        begin
          if P1 < P2 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P2-P1)/ro)
                     else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P2)/ro);
          if P5 < P6 then Qd3:=Cd2*Xv*Wd*sqrt(2*(P6-P5)/ro)
                     else Qd3:=-Cd2*Xv*Wd*sqrt(2*(P5-P6)/ro);
          if  (time < 0.002) then
            begin
              Qd1:=Qd1*(time)/0.002;
              Qd3:=Qd3*(time)/0.002;
            end;
          Qd2:=-Qd1;
          Qd4:=-Qd3;
        end
      else
        begin
          if P1 < P5 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P5-P1)/ro)
                     else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P5)/ro);
          if P2 < P6 then Qd2:=Cd2*Xv*Wd*sqrt(2*(P6-P2)/ro)
                     else Qd2:=-Cd2*Xv*Wd*sqrt(2*(P2-P6)/ro);
          if (time < 2.002) then
            begin
              Qd1:=Qd1*(time-0.2)/0.002;
              Qd2:=Qd2*(time-0.2)/0.002;
            end;
          Qd3:=-Qd1;
          Qd4:=-Qd2;
        end;
    end;

{ Cylinder }
  CC1:=0.0; CC2:=0.0; Xmax:=10.0; Xmin:=0.0;
  Qc1:=-(Ap*V7+CC1*(P3-P4)+CC2*P3);
  Qc2:=Ap*V7+CC1*(P3-P4)-CC2*P4;
  FC3:=Ap*(P3-P4);

{ Load }
  FM1:=-Ff;
```

```
  { Drain Line }

  DQ2:=Qd2+QL1;
  DQ3:=QL2+Qc1;
  DQ4:=Qc2+QL3;
  DQ5:=QL4+Qd3;
  DF7:=FC3+FM1;
  if (DF7>=0) and (X8>=Xmax) then DF7:=0.0;
  if (DF7<=0) and (X8<=Xmin) then DF7:=0.0;
  DX8:=V7;

  DYDX[1]:=(BETA/0.5)*DQ2;
  DYDX[2]:=(BETA/((Va/2-0.5)+X8*Ap))*DQ3;
  DYDX[3]:=(BETA/((Va/2-0.5)-X8*Ap))*DQ4;
  DYDX[4]:=(BETA/0.5)*DQ5;
  DYDX[5]:=DF7/Mt;
  DYDX[6]:=DX8;

END; { OF DERIVS }


{---------------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
               VAR yout: glnarray);
{---------------------------------------------------------------------------}

VAR
   i          : integer;
   xh,h6      : real;
   dym,dyt,yt: glnarray;

BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;

   if (yout[5]>0) and (yout[6]>=Xmax) then
```

```
      begin
        yout[5]:=0.0;
        yout[6]:=Xmax;
      end;
    if (yout[5]<0) and (yout[6]<=Xmin) then
      begin
        yout[5]:=0.0;
        yout[6]:=Xmin;
      end;
    for j:=1 to 4 do
      if yout[j]<0 then yout[j]:=0;

END;

{----------------------------------------------------------------------------}
 Procedure initialize_parameters;
{----------------------------------------------------------------------------}
 var
   Kq,wh,dh : real;

 begin
   time:=0.0;
   it:=0;
   data_no:=0;
   dt:=h;

   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
   Kq:=Cd*Wd*sqrt(Ps/ro); { flow gain, in^4/sec }
   Ff:=Mw*fc; { friction force, lbf }
   Atube:=pi*d*d/4;
   tube_volume:=Atube*m*dx;
   Vt:=Va+tube_volume*2;
   R:=128*mu/(pi*d*d*d*d);
   L:=ro/Atube;
   Co:=dx/(dt/2);

   for i:=1 to nn do
     begin
       Qa[i]:=0;
       Qb[i]:=0;
       Pa[i]:=0;
       Pb[i]:=0;
     end;
   for i:=1 to n do
     begin
       y[i]:=0;
       DYDX[I]:=0.0;
     end;
   Cn:=0;
   Cn_1:=0;
   Cn_2:=0;
   Ms:=0;
   area:=0;
   forv:=0;
```

```
 end;

{------------------------------ MAIN PROGRAM ------------------------------}

BEGIN
  assign(f,'d:PID.out');
  rewrite(f);
  initialize_parameters;
  repeat
    it:=it+1;
    PID;
    LineA;LineB;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
    Cn:=y[5];
    writeln(current:6:2,' ',time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4
 ',
                         y[4]:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
    if it>=10 then
      begin
        data_no:=data_no+1;
        tout[data_no]:=time;
        cout[data_no]:=current;
        rout[data_no]:=y[5];
        it:=0;
      end;
   err:=y[5]-0.8267;
   area:=area+(err*err+forv)*h/2;
   forv:=err*err;
   until (time>final_time) or (data_no>=100);
   for i:=1 to 200 do
     begin
       writeln(f,tout[i]:8:4,' ',cout[i]:10:4,' ',rout[i]:10:4);
       writeln(tout[i]:8:4,' ',rout[i]:10:4);
     end;
   writeln(area); readln;
   close(f);
end.
```

APPENDIX N

OPTIMIZATION FOR PID

```
PROGRAM DYNOPT; { Dynamic Simulation using POWELL Optimizing Method }
                { June 7, 1989 }

uses glvar{,grafpak,quickplt,dos,crt},dyna,optimum;

var
  i,j:integer;

begin
  assign(fout,'d:opt2.out');
  rewrite(fout);
  p[1]:=0.1; p[2]:=0.13; p[3]:=0;  { p[1]=Kp, p[2]=Ki, p[3]=Kd }
  ftol:=0.1;

  for i:=1 to LL do
    begin
      for j:=1 to LL do
        XI[i,j]:=0;
      XI[i,i]:=1
    end;

  powell(p,xi,LL,np,ftol,iter,fret);

  close(fout);
end.
```

```
UNIT OPTIMUM;

interface
uses glvar,dyna{,grafpak,quickplt,dos,crt};

PROCEDURE powell(VAR p: gllarray; VAR xi: glnpbynp; n,np: integer;
       ftol: real; VAR iter: integer; VAR fret: real);

implementation

PROCEDURE powell(VAR p: gllarray; VAR xi: glnpbynp; n,np: integer;
       ftol: real; VAR iter: integer; VAR fret: real);

LABEL 1,99;
CONST
   itmax=100;
VAR
   j,ibig,i: integer;
   t,fptt,fp,del: real;
   pt,ptt,xit: gllarray;

FUNCTION fidim(x: real): real;
var
  j:integer;
  xt:gllarray;

begin
  for j:=1 to ncom do
      xt[j] := pcom[j]+x*xicom[j];
  fidim := FNC(xt)
end;


FUNCTION func(x: real): real;
begin
  func := fidim(x)
end;

PROCEDURE mnbrak(VAR ax,bx,cx,fa,fb,fc: real);
(* Programs using routine MNBRAK must supply an external
function func(x:real):real for which a minimum is to be found *)
LABEL 1;
CONST
   gold=1.618034;
   glimit=100.0;
   tiny=1.0e-20;
VAR
   ulim,u,r,q,fu,dum: real;
FUNCTION max(a,b: real): real;
   BEGIN
      IF (a > b) THEN max := a ELSE max := b
   END;
FUNCTION sign(a,b: real): real;
   BEGIN
      IF (b > 0.0) THEN sign := abs(a) ELSE sign := -abs(a)
```

```
   END;
BEGIN
   fa := func(ax);
   fb := func(bx);
   IF (fb > fa) THEN BEGIN
      dum := ax;
      ax := bx;
      bx := dum;
      dum := fb;
      fb := fa;
      fa := dum
   END;
   cx := bx+gold*(bx-ax);
   fc := func(cx);
1:    IF (fb >= fc) THEN BEGIN
      r := (bx-ax)*(fb-fc);
      q := (bx-cx)*(fb-fa);
      u := bx-((bx-cx)*q-(bx-ax)*r)/
         (2.0*sign(max(abs(q-r),tiny),q-r));
      ulim := bx+glimit*(cx-bx);
      IF ((bx-u)*(u-cx) > 0.0) THEN BEGIN
         fu := func(u);
         IF (fu < fc) THEN BEGIN
            ax := bx;
            fa := fb;
            bx := u;
            fb := fu;
            GOTO 1 END
         ELSE IF (fu > fb) THEN BEGIN
            cx := u;
            fc := fu;
            GOTO 1
         END;
         u := cx+gold*(cx-bx);
         fu := func(u)
      END ELSE IF  ((cx-u)*(u-ulim) > 0.0) THEN BEGIN
         fu := func(u);
         IF (fu < fc) THEN BEGIN
            bx := cx;
            cx := u;
            u := cx+gold*(cx-bx);
            fb := fc;
            fc := fu;
            fu := func(u)
         END
      END ELSE IF  ((u-ulim)*(ulim-cx) >= 0.0) THEN BEGIN
         u := ulim;
         fu := func(u)
      END ELSE BEGIN
         u := cx+gold*(cx-bx);
         fu := func(u)
      END;
      ax := bx;
      bx := cx;
      cx := u;
```

```
      fa := fb;
      fb := fc;
      fc := fu;
      GOTO 1
   END
END;

FUNCTION brent(ax,bx,cx,tol: real; VAR xmin: real): real;
(* Programs using routine BRENT must supply an external function
func(x:real):real whose minimum is to be found. *)
LABEL 1,2,3;
CONST
   itmax=100;
   cgold=0.3819660;
   zeps=1.0e-10;
VAR
   a,b,d,e,etemp: real;
   fu,fv,fw,fx: real;
   iter: integer;
   p,q,r,tol1,tol2: real;
   u,v,w,x,xm: real;
FUNCTION sign(a,b: real): real;
   BEGIN
      IF (b > 0.0) THEN sign := abs(a) ELSE sign := -abs(a)
   END;
BEGIN
   IF ax < cx THEN a := ax ELSE a := cx;
   IF ax > cx THEN b := ax ELSE b := cx;
   v := bx;
   w := v;
   x := v;
   e := 0.0;
   fx := func(x);
   fv := fx;
   fw := fx;
   FOR iter := 1 to itmax DO BEGIN
      xm := 0.5*(a+b);
      tol1 := tol*abs(x)+zeps;
      tol2 := 2.0*tol1;
      IF (abs(x-xm) <= (tol2-0.5*(b-a))) THEN GOTO 3;
      IF (abs(e) > tol1) THEN BEGIN
         r := (x-w)*(fx-fv);
         q := (x-v)*(fx-fw);
         p := (x-v)*q-(x-w)*r;
         q := 2.0*(q-r);
         IF (q > 0.0) THEN  p := -p;
         q := abs(q);
         etemp := e;
         e := d;
         IF((abs(p) >= abs(0.5*q*etemp)) OR (p <= q*(a-x))
            OR (p >= q*(b-x))) THEN GOTO 1;
         d := p/q;
         u := x+d;
         IF (((u-a) < tol2) OR ((b-u) < tol2)) THEN d := sign(tol1,xm-x);
         GOTO 2
```

```
      END;
1:      IF (x >= xm)  THEN e := a-x ELSE e := b-x;
      d := cgold*e;
2:      IF (abs(d) >= tol1)  THEN u := x+d ELSE u := x+sign(tol1,d);
      fu := func(u);
      IF (fu <= fx)  THEN BEGIN
         IF (u >= x)  THEN a := x ELSE b := x;
         v := w;
         fv := fw;
         w := x;
         fw := fx;
         x := u;
         fx := fu
      END ELSE BEGIN
         IF (u < x)  THEN a := u ELSE b := u;
         IF ((fu <= fw) OR (w = x))  THEN BEGIN
            v := w;
            fv := fw;
            w := u;
            fw := fu
         END ELSE IF ((fu <= fv) OR (v = x) OR (v = 2)) THEN BEGIN
            v := u;
            fv := fu
         END
      END
   END;
   writeln('pause in routine BRENT - too many iterations');
3:   xmin := x;
   brent := fx
END;


PROCEDURE linmin(VAR p,xi: gllarray; n: integer; VAR fret: real);
(* Programs using routine LINMIN must define the type
TYPE
   glnarray = ARRAY [1..n] OF real;
They must also declare the variables
VAR
   ncom: integer;
   pcom,xicom: glnarray;
in the main routine. Also the function FUNC referenced by BRENT
and MNBRAK must be set to return the function F1DIM. *)
CONST
   tol=1.0e-4;
VAR
   j: integer;
   xx,xmin,fx,fb,fa,bx,ax: real;
BEGIN
   ncom := n;
   FOR j := 1 to n DO BEGIN
      pcom[j] := p[j];
      xicom[j] := xi[j]
   END;
   ax := 0.0;
   xx := 1.0;
```

```
   bx := 2.0;
   mnbrak(ax,xx,bx,fa,fx,fb);
   fret := brent(ax,xx,bx,tol,xmin);
   FOR j := 1 to n DO BEGIN
      xi[j] := xmin*xi[j];
      p[j] := p[j]+xi[j]
   END
END;


{---------------------------- MAIN OF POWELL -----------------------------}
BEGIN
   fret := fnc(p);
   FOR j := 1 to n DO BEGIN
      pt[j] := p[j]
   END;
   iter := 0;
1:   iter := iter+1;
   fp := fret;
   ibig := 0;
   del := 0.0;
   FOR i := 1 to n DO BEGIN
      FOR j := 1 to n DO BEGIN
         xit[j] := xi[j,i]
      END;
      linmin(p,xit,n,fret);
      IF (abs(fp-fret) > del) THEN BEGIN
         del := abs(fp-fret);
         ibig := i
      END
   END;
   IF (2.0*abs(fp-fret) <= ftol*(abs(fp)+abs(fret))) THEN GOTO 99;
   IF (iter = itmax) THEN BEGIN
      writeln('pause in routine POWELL');
      writeln('too many interations'); readln
   END;
   FOR j := 1 to n DO BEGIN
      ptt[j] := 2.0*p[j]-pt[j];
      xit[j] := p[j]-pt[j];
      pt[j] := p[j]
   END;
   fptt := fnc(ptt);
   IF (fptt >= fp) THEN GOTO 1;
   t := 2.0*(fp-2.0*fret+fptt)*sqr(fp-fret-del)-del*sqr(fp-fptt);
   IF (t >= 0.0) THEN GOTO 1;
   linmin(p,xit,n,fret);
   FOR j := 1 to n DO BEGIN
      xi[j,ibig] := xit[j]
   END;
   GOTO 1;
99:   END;

begin
end.
```

```
UNIT GLVAR;
interface

const
  LL=3;
  np=3;
  ndim=3;

type
  gllarray=array[1..LL] of real;
  glndim=array[1..ndim] of real;
  glnpbynp=array[1..np,1..np] of real;

var
  ncom,iter:integer;
  p,pcom,xicom:gllarray;
  xi:glnpbynp;
  ftol,fret:real;
  fout:text;

implementation
begin
end.
```

```
UNIT DYNA;

interface
uses glvar;

function FNC(P:gllarray):real;

implementation
function FNC(P:gllarray):real;

CONST
{ constants for simulation }
  n=6;
  h=2.5e-4;
  final_time=10;
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.5e5; { bulk modulus, psi }
  Cd=0.61; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Dc=1.82; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Va=5; { total volume of oil, in^3 }
  Kqi=0.0012; { valve current - spool displacement ratio, in/mA }
  Wd=0.1; { orifice width in the valve }
  Ps=500; { supply pressure, psig }
  vis=2.8e-6; { absolute viscosity, lb-sec/in^2 }

{ constant for simulating line }
  m = 1; { total divided node number nn= 2*m + 1 }
  dx = 10; { divided line length }
  d = 0.25; { inner tube diameter }
  nn = 3; { total divided node number }
  mu = 2.8e-6;


TYPE
  glnarray=array[1..n] of real;
  glmarray=array[1..nn] of real;

VAR
  i,j,k,it,data_no                :integer;
  hh,dt,
  Ff,time,Ap,Vp,Xp,Mt,
  QL1,QL2,QL3,QL4,Vt,Xv,Kc,
  current,output_time,
  R,L,Co,
  Xmin,Xmax,Atube,tube_volume,
  Kpro,Kint,Kdrv,
  Ms,Cn,Cn_1,Cn_2,
  err,forv,area                   :real;
  y,dydx,yout                     :glnarray;
```

```
  Qa,Qb,Pa,Pb                       :glmarray;
  tout,cout,rout                    :array [1..200] of real;
  f                                 :text;

{-----------------------------------------------------------------------------}
 procedure PID;
{-----------------------------------------------------------------------------}
 const
   Rn=0.8267;

 var
   dMs:real;

 begin
   dMs:=Kpro*(Cn_1-Cn)+Kint*(Rn-Cn)+Kdrv*(2*Cn_1-Cn_2-Cn);
   current:=Ms+dMs;
   if current>20 then current:=20;
   if current<-20 then current:=-20;
   Ms:=current;
   Cn_2:=Cn_1;
   Cn_1:=Cn;
 end;


{-----------------------------------------------------------------------------}
 procedure LineA;
{-----------------------------------------------------------------------------}
var
  i,j : integer;

begin

  Pa[1] := y[1];
  Pa[2*m+1]:=y[2];

  for i:=1 to m do
    begin
      j:=i*2;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
              (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
      Pa[j]:=(Pa[j-1]+Pa[j+1])/2 +
             (Co*L-R*dx/4)*(Qa[j-1]-Qa[j+1])/2;
      Qa[j]:=((Co*L-R*dx/4)*(Qa[j-1]+Qa[j+1]) + (Pa[j-1]-Pa[j+1]))
             /(2*(Co*L+R*dx/4));
      if Pa[j]<0 then Pa[j]:=0;
    end;
  Qa[1]:=((Co*L-R*dx/4)*Qa[2]+Pa[1]-Pa[2])/(Co*L+R*dx/4);
  Qa[2*m+1]:=((Co*L-R*dx/4)*Qa[2*m]-Pa[2*m+1]+Pa[2*m])/(Co*L+R*dx/4);
```

```
  QL1:=-Qa[1];
  QL2:=Qa[2*m+1];
end;

{-----------------------------------------------------------------------------}
 procedure LineB;
{-----------------------------------------------------------------------------}
var
  i,j : integer;

begin

  Pb[1] := y[3];
  Pb[2*m+1]:=y[4];

  for i:=1 to m do
    begin
      j:=i*2;
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
              (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
              /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  for i:=1 to m-1 do
    begin
      j:=i*2+1;
      Pb[j]:=(Pb[j-1]+Pb[j+1])/2 +
             (Co*L-R*dx/4)*(Qb[j-1]-Qb[j+1])/2;
      Qb[j]:=((Co*L-R*dx/4)*(Qb[j-1]+Qb[j+1]) + (Pb[j-1]-Pb[j+1]))
             /(2*(Co*L+R*dx/4));
      if Pb[j]<0 then Pb[j]:=0;
    end;
  Qb[1]:=((Co*L-R*dx/4)*Qb[2]+Pb[1]-Pb[2])/(Co*L+R*dx/4);
  Qb[2*m+1]:=((Co*L-R*dx/4)*Qb[2*m]-Pb[2*m+1]+Pb[2*m])/(Co*L+R*dx/4);
  QL3:=-Qb[1];
  QL4:=Qb[2*m+1];
end;

{-----------------------------------------------------------------------------}
 Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{-----------------------------------------------------------------------------}
 var
   P1,P2,P3,P4,P5,P6,V7,X8,S5,
   Cd1,Cd2,Qd1,Qd2,Qd3,Qd4,
   change_time,CC1,CC2,CC3,Qc1,Qc2,FC3,
   FM1,
   DQ2,DQ3,DQ4,DQ5,DF7,DX8                             : real;

 begin
   P1:=Ps; P2:=Y[1]; P3:=Y[2]; P4:=y[3]; P5:=y[4]; P6:=0;
   V7:=Y[5]; X8:=Y[6];

  { Power source }
```

```
{ Supply Line }

{ Signal }

{ Servo Valve }
  Xv:= current*Kqi;
  Cd1:=Cd; Cd2:=Cd;

  if current=0 then
    begin
      Qd1:=0;
      Qd2:=0;
      Qd3:=0;
      Qd4:=0;
    end
  else
    begin
      if current >0 then
        begin
          if P1 < P2 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P2-P1)/ro)
                     else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P2)/ro);
          if P5 < P6 then Qd3:=Cd2*Xv*Wd*sqrt(2*(P6-P5)/ro)
                     else Qd3:=-Cd2*Xv*Wd*sqrt(2*(P5-P6)/ro);
          if  (time < 0.002) then
            begin
              Qd1:=Qd1*(time)/0.002;
              Qd3:=Qd3*(time)/0.002;
            end;
          Qd2:=-Qd1;
          Qd4:=-Qd3;
        end
      else
        begin
          if P1 < P5 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P5-P1)/ro)
                     else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P5)/ro);
          if P2 < P6 then Qd2:=Cd2*Xv*Wd*sqrt(2*(P6-P2)/ro)
                     else Qd2:=-Cd2*Xv*Wd*sqrt(2*(P2-P6)/ro);
          if (time < 2.002) then
            begin
              Qd1:=Qd1*(time-0.2)/0.002;
              Qd2:=Qd2*(time-0.2)/0.002;
            end;
          Qd3:=-Qd1;
          Qd4:=-Qd2;
        end;
    end;

{ Cylinder }
  CC1:=0.0; CC2:=0.0; Xmax:=10.0; Xmin:=0.0;
  Qc1:=-(Ap*V7+CC1*(P3-P4)+CC2*P3);
  Qc2:=Ap*V7+CC1*(P3-P4)-CC2*P4;
  FC3:=Ap*(P3-P4);

{ Load }
  FM1:=-Ff;
```

```
{ Drain Line }

DQ2:=Qd2+QL1;
DQ3:=QL2+Qc1;
DQ4:=Qc2+QL3;
DQ5:=QL4+Qd3;
DF7:=FC3+FM1;
if (DF7>=0) and (X8>=Xmax) then DF7:=0.0;
if (DF7<=0) and (X8<=Xmin) then DF7:=0.0;
DX8:=V7;

DYDX[1]:=(BETA/0.5)*DQ2;
DYDX[2]:=(BETA/((Va/2-0.5)+X8*Ap))*DQ3;
DYDX[3]:=(BETA/((Va/2-0.5)-X8*Ap))*DQ4;
DYDX[4]:=(BETA/0.5)*DQ5;
DYDX[5]:=DF7/Mt;
DYDX[6]:=DX8;

END; { OF DERIVS }


{-----------------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
               VAR yout: glnarray);
{-----------------------------------------------------------------------------}

VAR
   i,j       : integer;
   xh,h6     : real;
   dym,dyt,yt: glnarray;

BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;
```

```
   if (yout[5]>0) and (yout[6]>=Xmax) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmax;
     end;
   if (yout[5]<0) and (yout[6]<=Xmin) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmin;
     end;
   for j:=1 to 4 do
     if yout[j]<0 then yout[j]:=0;

END;

{----------------------------------------------------------------------------}
 Procedure initialize_parameters;
{----------------------------------------------------------------------------}
 var
   i:integer;
   Kq,wh,dh : real;

 begin
   time:=0.0;
   it:=0;
   data_no:=0;
   dt:=h;

   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
   Kq:=Cd*Wd*sqrt(Ps/ro); { flow gain, in^4/sec }
   Ff:=Mw*fc; { friction force, lbf }
   Atube:=pi*d*d/4;
   tube_volume:=Atube*m*dx;
   Vt:=Va+tube_volume*2;
   R:=128*mu/(pi*d*d*d*d);
   L:=ro/Atube;
   Co:=dx/(dt/2);

   for i:=1 to nn do
     begin
       Qa[i]:=0;
       Qb[i]:=0;
       Pa[i]:=0;
       Pb[i]:=0;
     end;
   for i:=1 to n do
     begin
       y[i]:=0;
       DYDX[I]:=0.0;
     end;
   Kpro:=P[1];
   Kint:=P[2];
   Kdrv:=P[3];
   Cn:=0;
```

```
   Cn_1:=0;
   Cn_2:=0;
   Ms:=0;
   area:=0;
   forv:=0;

 end;

{------------------------------ MAIN PROGRAM -------------------------------}

BEGIN

  initialize_parameters;
  writeln(iter:4,' ',Kpro:8:4,' ',Kint:10:4,' ',Kdrv:10:4);
  write(fout,Kpro:8:4,' ',Kint:10:4,' ',Kdrv:10:4);
  repeat
    it:=it+1;
    PID;
    LineA;LineB;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
    Cn:=y[6];
    writeln(current:5:1,' ',time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,'
 ',
                          y[4]:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
    err:=y[6]-1;
    area:=area+(err*err+forv)*h/2;
    forv:=err*err;

    if it>=10 then
      begin
        data_no:=data_no+1;
        it:=0;
      end;
   until (time>final_time) or (data_no>=400);
   FNC:=area;
   writeln;
   writeln('**** error area = ',area);
   writeln; writeln;writeln;
   write(fout,'  ',area);
   writeln(fout);

end;
begin
end.
```

APPENDIX O

EXAMPLE OF OPTIMIZATION FOR PID (POSITION CONTROL)

TABLE XI

OPTIMIZATION FOR PID GAINS
(POSITION CONTROL)

No.	Kp	Ki	Kd	ISE
1	0.1000	0.1300	0.0000	2.3921481237E-01
2	0.1000	0.1300	0.0000	2.3921481237E-01
3	1.1000	0.1300	0.0000	2.3032893085E-01
4	2.7180	0.1300	0.0000	2.0613377125E-01
5	5.3361	0.1300	0.0000	1.8447601815E-01
6	6.6497	0.1300	0.0000	1.9479066569E-01
7	5.3361	0.1300	0.0000	1.8447601815E-01
8	4.3361	0.1300	0.0000	1.9720800755E-01
9	5.8378	0.1300	0.0000	1.8020660096E-01
10	6.1479	0.1300	0.0000	1.7873028994E-01
11	6.5084	0.1300	0.0000	1.9365491206E-01
12	6.2856	0.1300	0.0000	1.7717334867E-01
13	6.3707	0.1300	0.0000	1.7726541953E-01
14	6.3185	0.1300	0.0000	1.7724708470E-01
15	6.2330	0.1300	0.0000	1.7840192881E-01
16	6.2655	0.1300	0.0000	1.7697046433E-01
17	6.2531	0.1300	0.0000	1.7685735438E-01
18	6.2454	0.1300	0.0000	1.7837295368E-01
19	6.2579	0.1300	0.0000	1.7691051593E-01
20	6.2502	0.1300	0.0000	1.7689208633E-01
21	6.2537	0.1300	0.0000	1.7686411136E-01
22	6.2525	0.1300	0.0000	1.7687769050E-01
23	6.2531	0.1300	0.0000	1.7685735438E-01
24	6.2531	1.1300	0.0000	1.3306133489E-01
25	6.2531	2.7480	0.0000	1.2793530209E-01
26	6.2531	2.0411	0.0000	1.2853225706E-01
27	6.2531	5.3661	0.0000	1.2721316421E-01
28	6.2531	4.2590	0.0000	1.2692498245E-01
29	6.2531	4.2590	0.0000	1.2692498245E-01
30	6.2531	3.6819	0.0000	1.2759996152E-01
31	6.2531	4.6819	0.0000	1.2708643365E-01
32	6.2531	4.3474	0.0000	1.2603868696E-01
33	6.2531	4.4643	0.0000	1.2543904913E-01
34	6.2531	4.5474	0.0000	1.2785310968E-01
35	6.2531	4.5132	0.0000	1.2673466573E-01
36	6.2531	4.4197	0.0000	1.2765172035E-01
37	6.2531	4.4830	0.0000	1.2705934842E-01
38	6.2531	4.4473	0.0000	1.2629185289E-01
39	6.2531	4.4715	0.0000	1.2822948524E-01
40	6.2531	4.4578	0.0000	1.2606320834E-01

TABLE XI(Continued)

No.	Kp	Ki	Kd	ISE
41	6.2531	4.4671	0.0000	1.2552048532E-01
42	6.2531	4.4618	0.0000	1.2687821879E-01
43	6.2531	4.4654	0.0000	1.2572390090E-01
44	6.2531	4.4634	0.0000	1.2677564784E-01
45	6.2531	4.4648	0.0000	1.2543722401E-01
46	6.2531	4.4648	0.0000	1.2543722401E-01
47	6.2531	4.4648	1.0000	1.2543508733E-01
48	6.2531	4.4648	2.6180	1.2535346074E-01
49	6.2531	4.4648	5.2361	1.2745273845E-01
50	6.2531	4.4648	2.6180	1.2535346074E-01
51	6.2531	4.4648	3.6180	1.2759041824E-01
52	6.2531	4.4648	2.0000	1.2535055706E-01
53	6.2531	4.4648	2.3073	1.2535242803E-01
54	6.2531	4.4648	1.6180	1.2534635647E-01
55	6.2531	4.4648	1.3820	1.2545765151E-01
56	6.2531	4.4648	1.7639	1.2534549887E-01
57	6.2531	4.4648	1.7321	1.2534512913E-01
58	6.2531	4.4648	1.7101	1.2534487481E-01
59	6.2531	4.4648	1.6750	1.2534446817E-01
60	6.2531	4.4648	1.6532	1.2534421746E-01
61	6.2531	4.4648	1.6398	1.2534659618E-01
62	6.2531	4.4648	1.6615	1.2534431316E-01
63	6.2531	4.4648	1.6481	1.2534415835E-01
64	6.2531	4.4648	1.6449	1.2534412183E-01
65	6.2531	4.4648	1.6430	1.2534409927E-01
66	6.2531	4.4648	1.6417	1.2534661781E-01
67	6.2531	4.4648	1.6437	1.2534410789E-01
68	6.2531	4.4648	1.6425	1.2534409394E-01
69	6.2531	4.4648	1.6422	1.2534409065E-01
70	6.2531	4.4648	1.6420	1.2534408861E-01
71	12.4062	8.7995	3.2841	1.2696201234E-01
72	6.2531	4.4648	1.6420	1.2534408861E-01
73	12.4062	8.7995	3.2841	1.2696201234E-01
74	-3.7028	-2.5490	-1.0148	9.9987499777E-01
75	6.2531	4.4648	1.6420	1.2534408861E-01
76	2.4503	1.7857	0.6272	1.3254375128E-01
77	8.6034	6.1205	2.2692	1.2566934460E-01
78	7.2187	5.1450	1.8997	1.2622038971E-01
79	4.8006	3.4415	1.2544	1.2683264421E-01
80	5.6983	4.0739	1.4940	1.2617246714E-01

TABLE XI (Continued)

No.	Kp	Ki	Kd	ISE
81	6.6219	4.7246	1.7404	1.2700849641E-01
82	6.0412	4.3155	1.5855	1.2653558005E-01
83	6.3940	4.5640	1.6796	1.2645865033E-01
84	6.1722	4.4077	1.6204	1.2612558763E-01
85	6.3069	4.5027	1.6564	1.2559380981E-01
86	6.2222	4.4430	1.6338	1.2752201608E-01
87	6.2737	4.4793	1.6475	1.2639011289E-01
88	6.2413	4.4565	1.6389	1.2667861029E-01
89	6.2610	4.4703	1.6441	1.2595330441E-01
90	6.2486	4.4616	1.6408	1.2776606980E-01
91	6.2561	4.4669	1.6428	1.2557926407E-01
92	6.2514	4.4636	1.6416	1.2661383000E-01
93	6.2543	4.4656	1.6423	1.2663394534E-01
94	6.2525	4.4643	1.6419	1.2534776537E-01
95	6.2529	4.4646	1.6420	1.2534498998E-01
96	6.2536	4.4651	1.6421	1.2663961375E-01
97	6.2533	4.4649	1.6421	1.2767304499E-01
98	6.2530	4.4647	1.6420	1.2534441598E-01
99	6.2532	4.4648	1.6420	1.2534639201E-01
100	6.2531	4.4647	1.6420	1.2534421142E-01
101	6.2531	4.4648	1.6420	1.2534653225E-01
102	6.2531	4.4648	1.6420	1.2534413520E-01
103	6.2531	4.4648	1.6420	1.2534658690E-01
104	6.2531	4.4648	1.6420	1.2534410636E-01
105	6.2531	4.4648	1.6420	1.2534660793E-01
106	6.2531	4.4648	1.6420	1.2534409538E-01
107	6.2531	4.4648	1.6420	1.2534661599E-01
108	6.2531	4.4648	1.6420	1.2534409120E-01
109	6.2531	4.4648	1.6420	1.2534408652E-01
110	6.2531	4.4648	1.6420	1.2534661789E-01
111	6.2531	4.4648	1.6420	1.2534408732E-01
112	6.2531	4.4648	1.6420	1.2534661862E-01
113	6.2531	4.4648	1.6420	1.2534408683E-01
114	6.2531	4.4648	1.6420	1.2534408634E-01
115	6.2531	4.4648	1.6420	1.2534661879E-01
116	6.2531	4.4648	1.6420	1.2534408641E-01
117	6.2531	4.4648	1.6420	1.2534661885E-01
118	6.2531	4.4648	1.6420	1.2534408636E-01
119	6.2531	4.4648	1.6420	1.2534408632E-01
120	6.2531	4.4648	1.6420	1.2534408631E-01

TABLE XI(Continued)

No.	Kp	Ki	Kd	ISE
121	6.2531	4.4648	1.6420	1.2534408630E-01
122	6.2531	4.4648	1.6420	1.2534408630E-01
123	6.2531	4.4648	1.6420	1.2534661886E-01
124	6.2531	4.4648	1.6420	1.2534408630E-01
125	7.2531	4.4648	1.6420	1.2763986545E-01
126	4.6351	4.4648	1.6420	1.2712267319E-01
127	6.2531	4.4648	1.6420	1.2534408630E-01
128	5.6351	4.4648	1.6420	1.2758676646E-01
129	6.6351	4.4648	1.6420	1.2967715046E-01
130	6.0653	4.4648	1.6420	1.2727304730E-01
131	6.3990	4.4648	1.6420	1.2647331186E-01
132	6.2544	4.4648	1.6420	1.2534648752E-01
133	6.2294	4.4648	1.6420	1.2534700105E-01
134	6.2441	4.4648	1.6420	1.2534517377E-01
135	6.2489	4.4648	1.6420	1.2534459074E-01
136	6.2515	4.4648	1.6420	1.2534427831E-01
137	6.2525	4.4648	1.6420	1.2534415954E-01
138	6.2536	4.4648	1.6420	1.2534656864E-01
139	6.2529	4.4648	1.6420	1.2534411426E-01
140	6.2533	4.4648	1.6420	1.2534659967E-01
141	6.2530	4.4648	1.6420	1.2534409698E-01
142	6.2532	4.4648	1.6420	1.2534661153E-01
143	6.2531	4.4648	1.6420	1.2534409038E-01
144	6.2531	4.4648	1.6420	1.2534661606E-01
145	6.2531	4.4648	1.6420	1.2534408786E-01
146	6.2531	4.4648	1.6420	1.2534661778E-01
147	6.2531	4.4648	1.6420	1.2534408689E-01
148	6.2531	4.4648	1.6420	1.2534661845E-01
149	6.2531	4.4648	1.6420	1.2534408653E-01
150	6.2531	4.4648	1.6420	1.2534661871E-01
151	6.2531	4.4648	1.6420	1.2534408639E-01
152	6.2531	4.4648	1.6420	1.2534661880E-01
153	6.2531	4.4648	1.6420	1.2534408633E-01
154	6.2531	4.4648	1.6420	1.2534661884E-01
155	6.2531	4.4648	1.6420	1.2534408632E-01
156	6.2531	4.4648	1.6420	1.2534661885E-01
157	6.2531	4.4648	1.6420	1.2534408630E-01
158	6.2531	4.4648	1.6420	1.2534408630E-01
159	6.2531	4.4648	1.6420	1.2534408630E-01
160	6.2531	4.4648	1.6420	1.2534408630E-01

TABLE XI (Continued)

No.	Kp	Ki	Kd	ISE
161	6.2531	4.4648	1.6420	1.2534408630E-01
162	6.2531	4.4648	1.6420	1.2534408630E-01
163	6.2531	4.4648	1.6420	1.2534408630E-01
164	6.2531	4.4648	1.6420	1.2534408630E-01
165	6.2531	4.4648	1.6420	1.2534408630E-01
166	6.2531	4.4648	1.6420	1.2534408630E-01
167	6.2531	4.4648	1.6420	1.2534408630E-01
168	6.2531	4.4648	1.6420	1.2534408630E-01
169	6.2531	4.4648	1.6420	1.2534408630E-01
170	6.2531	4.4648	1.6420	1.2534408630E-01
171	6.2531	5.4648	1.6420	1.2671688208E-01
172	6.2531	2.8467	1.6420	1.2760549403E-01
173	6.2531	4.4648	1.6420	1.2534408630E-01
174	6.2531	3.8467	1.6420	1.2646685820E-01
175	6.2531	4.8467	1.6420	1.2734546951E-01
176	6.2531	4.2845	1.6420	1.2614911689E-01
177	6.2531	4.6107	1.6420	1.2616560224E-01
178	6.2531	4.4468	1.6420	1.2606888153E-01
179	6.2531	4.5205	1.6420	1.2587292426E-01
180	6.2531	4.4856	1.6420	1.2662122367E-01
181	6.2531	4.4727	1.6420	1.2746390710E-01
182	6.2531	4.4579	1.6420	1.2610271501E-01
183	6.2531	4.4678	1.6420	1.2560373150E-01
184	6.2531	4.4621	1.6420	1.2709861670E-01
185	6.2531	4.4659	1.6420	1.2655891865E-01
186	6.2531	4.4638	1.6420	1.2752791758E-01
187	6.2531	4.4652	1.6420	1.2663824048E-01
188	6.2531	4.4644	1.6420	1.2534680710E-01
189	6.2531	4.4647	1.6420	1.2534443165E-01
190	6.2531	4.4649	1.6420	1.2664094845E-01
191	6.2531	4.4648	1.6420	1.2534630224E-01
192	6.2531	4.4647	1.6420	1.2534420546E-01
193	6.2531	4.4648	1.6420	1.2534649541E-01
194	6.2531	4.4648	1.6420	1.2534657133E-01
195	6.2531	4.4648	1.6420	1.2534413151E-01
196	6.2531	4.4648	1.6420	1.2534660065E-01
197	6.2531	4.4648	1.6420	1.2534410353E-01
198	6.2531	4.4648	1.6420	1.2534661190E-01
199	6.2531	4.4648	1.6420	1.2534409288E-01
200	6.2531	4.4648	1.6420	1.2534661620E-01

TABLE XI (Continued)

No.	Kp	Ki	Kd	ISE
201	6.2531	4.4648	1.6420	1.2534408881E-01
202	6.2531	4.4648	1.6420	1.2534661784E-01
203	6.2531	4.4648	1.6420	1.2534408726E-01
204	6.2531	4.4648	1.6420	1.2534661847E-01
205	6.2531	4.4648	1.6420	1.2534408667E-01
206	6.2531	4.4648	1.6420	1.2534661871E-01
207	6.2531	4.4648	1.6420	1.2534408644E-01
208	6.2531	4.4648	1.6420	1.2534661880E-01
209	6.2531	4.4648	1.6420	1.2534408636E-01
210	6.2531	4.4648	1.6420	1.2534661884E-01
211	6.2531	4.4648	1.6420	1.2534408632E-01
212	6.2531	4.4648	1.6420	1.2534408630E-01
213	6.2531	4.4648	1.6420	1.2534661885E-01
214	6.2531	4.4648	1.6420	1.2534408630E-01
215	6.2531	4.4648	1.6420	1.2534661885E-01
216	6.2531	4.4648	1.6420	1.2534408630E-01
217	6.2531	4.4648	1.6420	1.2534661885E-01
218	6.2531	4.4648	1.6420	1.2534661885E-01
219	6.2531	4.4648	1.6420	1.2534408630E-01
220	6.2531	4.4648	1.6420	1.2534661674E-01
221	6.2531	4.4648	1.6420	1.2534409003E-01
222	6.2531	4.4648	1.6420	1.2534408630E-01
223	6.2531	4.4648	1.6420	1.2534408772E-01
224	6.2531	4.4648	1.6420	1.2534661805E-01
225	6.2531	4.4648	1.6420	1.2534408702E-01
226	6.2531	4.4648	1.6420	1.2534661854E-01
227	6.2531	4.4648	1.6420	1.2534408657E-01
228	6.2531	4.4648	1.6420	1.2534661873E-01
229	6.2531	4.4648	1.6420	1.2534408640E-01
230	6.2531	4.4648	1.6420	1.2534661881E-01
231	6.2531	4.4648	1.6420	1.2534408634E-01
232	6.2531	4.4648	1.6420	1.2534661884E-01
233	6.2531	4.4648	1.6420	1.2534408631E-01
234	6.2531	4.4648	1.6420	1.2534661885E-01
235	6.2531	4.4648	1.6420	1.2534408630E-01
236	6.2531	4.4648	1.6420	1.2534661885E-01
237	6.2531	4.4648	1.6420	1.2534661886E-01
238	6.2531	4.4648	1.6420	1.2534661885E-01
239	6.2531	4.4648	1.6420	1.2534661885E-01
240	6.2531	4.4648	1.6420	1.2534661885E-01

TABLE XI(Continued)

No.	Kp	Ki	Kd	ISE
241	6.2531	4.4648	1.6420	1.2534408630E-01
242	6.2531	4.4648	1.6420	1.2534408630E-01
243	6.2531	4.4648	1.6420	1.2534661885E-01
244	6.2531	4.4648	1.6420	1.2534408630E-01
245	6.2531	4.4648	1.6420	1.2534408630E-01
246	6.2531	4.4648	1.6420	1.2534661885E-01
247	6.2531	4.4648	1.6420	1.2534408630E-01
248	6.2531	4.4648	1.6420	1.2534661885E-01
249	6.2531	4.4648	1.6420	1.2534408630E-01
250	6.2531	4.4648	1.6420	1.2534408630E-01
251	6.2531	4.4648	1.6420	1.2534408630E-01
252	6.2531	4.4648	1.6420	1.2534408630E-01
253	6.2531	4.4648	1.6420	1.2534408630E-01
254	6.2531	4.4648	1.6420	1.2534408630E-01
255	6.2531	4.4648	1.6420	1.2534408630E-01
256	6.2531	4.4648	1.6420	1.2534408630E-01
257	6.2531	4.4648	1.6420	1.2534408630E-01
258	6.2531	4.4648	1.6420	1.2534408630E-01
259	6.2531	4.4648	1.6420	1.2534408630E-01
260	6.2531	4.4648	1.6420	1.2534408630E-01

APPENDIX P

VSS CONTROL

```
PROGRAM DLINE;

CONST
{ constants for simulation }
  n=6;
  h=1.0e-4;
  final_time=1.0;
  gravity=386; { gravity accelration, in/sec^2 }

{ constants in the system }
  beta=1.0e5; { bulk modulus, psi }
  Cd=6.1; { discharge coefficient }
  ro=0.78e-4; { density of oil, lb-sec^2/in^4 }
  Kce=0.004; { total flow-pressure coefficient, in^3/sec-psi }
  Dc=1.92; { inner diameter of the cylinder, in }
  Dr=1.378; { diameter of the ram, in }
  Mw=100; { total weight of piston & load reffered to piston, lbf }
  fc=0.1; { friction coefficient }
  Vt=20; { total volume of oil, in^3 }
{  Xv=0.005; }{ valve displacement, in }
  Wd=0.005; { orifice width in the valve }
  Ps=1000; { supply pressure, psig }

  Rs=0.38; { required speed of piston, in/sec }

{ constant for simulating line }
  m =  2;
  dx = 100.0;
  mu = 8.3e-6;
  r = 0.125;

TYPE
  glnarray=array[1..n] of real;
  glmarray=array[1..m] of real;

VAR
  i,j,it                          :integer;
  hh,dt,
  X1,X2,fX1,Xv,Us,
  Ff,time,Ap,Vp,Xp,Mt,
  QL1,QL2,QL3,QL4,
  output_time,data_no,
  Xmin,Xmax,Atube,tube_volume     :real;
  y,dydx,yout                     :glnarray;
  Ua,Ub,Pa,Pb                     :glmarray;
  f                               :text;

{--------------------------------------------------------------------------}
 procedure LineA;
{--------------------------------------------------------------------------}
var
  i           : integer;
  co1,co2     : real;
```

```
begin
{    dt:=hh;        }
    co1 := beta*dt/dx;
    co2 := dt/(ro*dx);

    Pa[1] := y[1];
    Pa[m] := y[2];
{
    Pa[1] := -co1*(Ua[2]-Ua[1]) + Pa[1];
    Pa[m] := -co1*(Ua[m]-Ua[m-1]) + Pa[m];
}
    for i:=2 to m-1 do
      Pa[i] := -0.5*co1*(Ua[i+1]-Ua[i-1]) + Pa[i];

    Ua[1] := -co2*(Pa[2]-Pa[1]) + Ua[1] -32*mu*Ua[1]/(4*r*r) {2.0*mu*Ua[1]/(ro*r
*r)};
    Ua[m] := -co2*(Pa[m]-Pa[m-1]) + Ua[m] -32*mu*Ua[m]/(4*r*r) {2.0*mu*Ua[m]/(ro
*r*r)};
    for i:=2 to m-1 do
      Ua[i] := -0.5*co2*(Pa[i+1]-Pa[i-1]) + Ua[i] -32*mu*Ua[i]/(4*r*r) {2.0*mu*U
a[i]/(ro*r*r)};

    QL1:=-Atube*Ua[1];
    QL2:= Atube*Ua[m];

end;

{-----------------------------------------------------------------------------}
 procedure LineB;
{-----------------------------------------------------------------------------}
var
  i            : integer;
  co1,co2      : real;

begin
{    dt:=hh;        }
    co1 := beta*dt/dx;
    co2 := dt/(ro*dx);

    Pb[1] := y[3];
    Pb[m] := y[4];
{
    Pb[1] := -co1*(Ub[2]-Ub[1]) + Pb[1];
    Pb[m] := -co1*(Ub[m]-Ub[m-1]) + Pb[m];
}
    for i:=2 to m-1 do
      Pb[i] := -0.5*co1*(Ub[i+1]-Ub[i-1]) + Pb[i];

    Ub[1] := -co2*(Pb[2]-Pb[1]) + Ub[1] -32*mu*Ub[1]/(4*r*r) {2.0*mu*Ub[1]/(ro*r
*r)};
    Ub[m] := -co2*(Pb[m]-Pb[m-1]) + Ub[m] -32*mu*Ub[m]/(4*r*r) {2.0*mu*Ub[m]/(ro
*r*r)};
    for i:=2 to m-1 do
      Ub[i] := -0.5*co2*(Pb[i+1]-Pb[i-1]) + Ub[i] -32*mu*Ub[i]/(4*r*r) {2.0*mu*U
b[i]/(ro*r*r)};

    QL3:=-Atube*Ub[1];
    QL4:= Atube*Ub[m];

end;
```

```
{---------------------------------------------------------------------------}
Procedure VSS_controller;
{---------------------------------------------------------------------------}
const
  alpha1=1.5;
  beta1=-1.5;
  alpha2=0.0005;
  beta2=-0.0005;
  C1=2;
  Pg=0.0;

var
  sign:integer;
  ks1,ks2,thigma:real;

begin
  thigma:=X2+C1*X1;
  if thigma>0 then sign:=1;
  if thigma<0 then sign:=-1;
  if X1*thigma>0 then ks1:=alpha1;
  if X1*thigma<0 then ks1:=beta1;
  if X2*thigma>0 then ks2:=alpha2;
  if X2*thigma<0 then ks2:=beta2;
  Xv:=ks1*X1+ks2*X2+Pg*sign;
end;

{---------------------------------------------------------------------------}
Procedure derivs(time:real; var y:glnarray; VAR dydx:glnarray);
{---------------------------------------------------------------------------}
var
  P1,P2,P3,P4,P5,P6,V7,X8,S5,
  rate_current,sir,Cd1,Cd2,Qd1,Qd2,Qd3,Qd4,
  change_time,CC1,CC2,CC3,Qc1,Qc2,FC3,
  Bb,Ks,FM1,
  DQ2,DQ3,DQ4,DQ5,DF7,DX8                            : real;

begin
  P1:=Ps; P2:=Y[1]; P3:=Y[2]; P4:=y[3]; P5:=y[4]; P6:=0;
  V7:=Y[5]; X8:=Y[6];

 { Power source }

 { Supply Line }
   LineA;

 { Signal }
   s5:=10.0;


 { Servo Valve }
   rate_current:=10.0;
   sir:=s5/rate_current;
   if sir>1 then sir:=1;
   if sir<-1 then sir:=-1;
   Cd1:=abs(sir)*Cd; Cd2:=Cd1;
```

```
  if sir=0 then
    begin
      Qd1:=0;
      Qd2:=0;
      Qd3:=0;
      Qd4:=0;
    end
  else
    begin
      if sir >0 then
        begin
          if P1 < P2 then Qd1:=Cd1*Xv*Wd*sqrt(2*(P2-P1)/ro)
                     else Qd1:=-Cd1*Xv*Wd*sqrt(2*(P1-P2)/ro);
          if P5 < P6 then Qd3:=Cd2*Xv*Wd*sqrt(2*(P6-P5)/ro)
                     else Qd3:=-Cd2*Xv*Wd*sqrt(2*(P5-P6)/ro);
          if  (time < 0.002) then
            begin
              Qd1:=Qd1*(time)/0.002;
              Qd3:=Qd3*(time)/0.002;
            end;
          Qd2:=-Qd1;
          Qd4:=-Qd3;
        end
      else
        begin
          if P1 < P5 then Qd1:=Cd1*sqrt(P5-P1)
                     else Qd1:=-Cd1*sqrt(P1-P5);
          if P2 < P6 then Qd2:=Cd2*sqrt(P6-P2)
                     else Qd2:=-Cd2*sqrt(P2-P6);
          if (time < 2.002) then
            begin
              Qd1:=Qd1*(time-0.2)/0.002;
              Qd2:=Qd2*(time-0.2)/0.002;
            end;
          Qd3:=-Qd1;
          Qd4:=-Qd2;
        end;
    end;

{ Cylinder }
  CC1:=0.003; CC2:=0.001; Xmax:=10.0; Xmin:=0.0;
  Qc1:=-(Ap*V7+CC1*(P3-P4)+CC2*P3);
  Qc2:=Ap*V7+CC1*(P3-P4)-CC2*P4;
  FC3:=Ap*(P3-P4);

{ Load }
  FM1:=-Ff;

{ Drain Line }
  LineB;

DQ2:=Qd2+QL1;
DQ3:=QL2+Qc1;
DQ4:=Qc2+QL3;
```

```
  DQ5:=QL4+Qd3;
  DF7:=FC3+FM1;
  if (DF7>=0) and (X8>=Xmax) then DF7:=0.0;
  if (DF7<=0) and (X8<=Xmin) then DF7:=0.0;
  DX8:=V7;

  DYDX[1]:=(BETA/0.5)*DQ2;
  DYDX[2]:=(BETA/((Vt/2-0.5-tube_volume)+X8*Ap))*DQ3;
  DYDX[3]:=(BETA/((Vt/2-0.5-tube_volume)+X8*Ap))*DQ4;
  DYDX[4]:=(BETA/0.5)*DQ5;
  DYDX[5]:=DF7/Mt;
  DYDX[6]:=DX8;

END; { OF DERIVS }


{---------------------------------------------------------------------------}
 PROCEDURE rk4(y,dydx: glnarray; n: integer; time,h: real;
               VAR yout: glnarray);
{---------------------------------------------------------------------------}

VAR
   i         : integer;
   xh,h6     : real;
   dym,dyt,yt: glnarray;

BEGIN
   hh := h*0.5;
   h6 := h/6.0;
   xh := time+hh;
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dydx[i]
   END;
   derivs(xh,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+hh*dyt[i]
   END;
   derivs(xh,yt,dym);
   FOR i := 1 to n DO BEGIN
      yt[i] := y[i]+h*dym[i];
      dym[i] := dyt[i]+dym[i]
   END;
   hh:=h;
   derivs(time+h,yt,dyt);
   FOR i := 1 to n DO BEGIN
      yout[i] := y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i])
   END;

   if (yout[5]>0) and (yout[6]>=Xmax) then
     begin
       yout[5]:=0.0;
       yout[6]:=Xmax;
     end;
   if (yout[5]<0) and (yout[6]<=Xmin) then
```

```
    begin
      yout[5]:=0.0;
      yout[6]:=Xmin;
    end;
  for j:=1 to 4 do
    if yout[j]<0 then yout[j]:=0;

END;

{-----------------------------------------------------------------------------}
 Procedure initialize_parameters;
{-----------------------------------------------------------------------------}
 var
   Kq,wh,dh : real;

 begin
   time:=0.0;
   it:=0;
   data_no:=0;
   dt:=h;

   Ap:=pi*(Dc*Dc-Dr*Dr)/4; { piston area, in^2 }
   Mt:=Mw/gravity; { total mass, lb }
   Kq:=Cd*Wd*sqrt(Ps/ro); { flow gain, in^4/sec }
   wh:=sqrt(4*beta*Ap*Ap/(Vt*Mt)); { hydraulic natural frequency, rad/sec }
   dh:=(Kce/Ap)*sqrt(beta*Mt/Vt); { damping ratio }
   Ff:=Mw*fc; { friction force, lbf }
   Atube:=pi*r*r;
   tube_volume:=Atube*(m-1)*dx;

   for i:=1 to m do
     begin
       Ua[i]:=0;
       Ub[i]:=0;
       Pa[i]:=0;
       Pb[i]:=0;
     end;
   for i:=1 to n do
     begin
       y[i]:=0;
       DYDX[I]:=0.0;
     end;
   fX1:=0;
   X1:=0.0000001;
   X2:=0.0000001;
   writeln(f,time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,' ',
                        y[4]:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
 end;

{------------------------------ MAIN PROGRAM ------------------------------}

BEGIN
  assign(f,'d:VDLINE.out');
  rewrite(f);
  initialize_parameters;
```

```
  repeat
    it:=it+1;
    VSS_controller;
    derivs(time,y,dydx);
    rk4(y,dydx,n,time,h,yout);
    for i:=1 to n do
      y[i]:=yout[i];
    time:=time+h;
    Vp:=y[5];
    Xp:=y[6];
    if time<0.01 then Us:=time*Rs/0.01
                 else Us:=Rs;
    X1:=Us-Vp;
    X2:=(X1-fX1)/h;
    fX1:=X1;
    writeln(time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,' ',
                         y[4]:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
    if it>=10 then
      begin
        writeln(f,time:6:4,' ',y[1]:10:4,' ',y[2]:10:4,' ',y[3]:10:4,' ',
                             y[4]:10:4,' ',y[5]:10:4,' ',y[6]:10:4);
        data_no:=data_no+1;
        it:=0;
      end;
   until (time>final_time) or (data_no>=300);

   close(f);
end.
```

APPENDIX Q

PID CONTROL PROGRAM FOR EXPERIMENT

```
PROGRAM SERVO_LINE_CONTROL;

const
   Offset = 2.6;
   Kp = 0.2259;
   Ki = 10.7263;
   Kd = 0;

type
   IOARRAY=Array[1..10,1..10] of Real;
   SYARRAY=Array[1..10] of Real;
   glns2= array [0..2000] of integer;
   glns5= array [0..500] of integer;
   outarray = array [0..1000] of real;


var
   BASE,TRIG,TVALUE,SAMPLE_RATE,
   DANUM,UPMODE,COUNT,BVALUE,
   HOUR,MINUTE,SECOND,TENTHS,
   secSTART,secEND,tenthsSTART,tenthsEND,
   CH1,GAIN1,FUNI1,NS1,NCHAN1,
   CH2,GAIN2,FUNI2,NS2,NCHAN2,
   CH3,GAIN3,FUNI3,NS3,NCHAN3,
   CH4,GAIN4,FUNI4,NS4,NCHAN4
                        : integer;
   current,setvolt,disp,pressure,
   time,vel,totsec,timestep,
   dMs,Ms,Cn,Cn_1,Cn_2,Rn
                        : real;
   LVDT,PRESS           : outarray;
   DAT1,DAT2,DAT6,DAT7  : glns5;
   DAT3,DAT4            : glns2;
   i,j,up_bound,
   low_bound,ctlsgl
                        : integer;   { for control signal from ADALAB }
   continue,st          : char;
   filename             : string[10];
   fo                   : text;


procedure INIT(var BASE: integer); EXTERNAL 'ADAPTP.COM';
procedure TRIGGER( var TRIG,VALUE: integer); EXTERNAL INIT[48];
procedure FASTAD( var CHAN,GAIN,FUNI,NS,NCHAN,DATA: integer);EXTERNAL INIT[42];
procedure DAOUT( var DANUM,UPMODE,COUNT: integer); EXTERNAL INIT[33];
procedure CLOCKIN( var HOUR,MINUTE,SECOND,TENTHS: integer); EXTERNAL INIT[66];
procedure CLOCKOUT( var HOUR,MINUTE,SECOND,TENTHS: integer); EXTERNAL INIT[69];


procedure SYSTEM_INIT;

begin
    ClrScr;
    BASE:=768;
    INIT(BASE);
```

```
    SAMPLE_RATE:=16000 ; { per second }
    TRIG:=1;
    TVALUE:=ROUND(1193210.0/SAMPLE_RATE - 4) div 2;
    { >>>>  D/A channel A <<<< }
    DANUM:=0; { bipolar at A }
    UPMODE:=1;
    COUNT:=0;  { digital voltage output }
    up_bound:=2047;
    low_bound:=-2048;
    DAOUT(DANUM,UPMODE,COUNT);


    { >>>>  channal #1  <<<< }
    CH1:=1;   { displacement by LVDT }
    GAIN1:=1;
    FUNI1:=1;
    NS1:=10;
    NCHAN1:=1;

    { >>>>  channal #2  <<<< }
    CH2:=2;   { pressure }
    GAIN2:=1;
    FUNI2:=1;
    NS2:=10;
    NCHAN2:=1;


end; { of initiation }

procedure SET_CLOCK;
begin
  hour:=0;
  minute:=0;
  second:=0;
  tenths:=0;
  CLOCKOUT(hour,minute,second,tenths);
end;


procedure PID;

begin
  dMs:=Kp*(Cn_1-Cn)+Ki*(Rn-Cn)+Kd*(2*Cn_1-Cn-2-Cn);
  current:=-(Ms+dMs);
  if current>7.5 then current:=7.5;
  if current<-12.5 then current:=-12.5;
  Ms:=current;
  Cn_2:=Cn_1;
  Cn_1:=Cn;
  setvolt:=current/4;
  if setvolt>=0 then ctlsgl:=round(setvolt*up_bound/5)
              else ctlsgl:=-round(setvolt*low_bound/5);
  DAOUT(DANUM,UPMODE,ctlsgl);
end;
```

```
procedure CO_PID;

begin
  dMs:=100*(Rn-Cn);
  current:=-(Ms+dMs);
  if current>7.5 then current:=7.5;
  if current<-12.5 then current:=-12.5;
  Ms:=current;
  Cn_2:=Cn_1;
  Cn_1:=Cn;
  setvolt:=current/4;
  if setvolt>=0 then ctlsgl:=round(setvolt*up_bound/5)
                 else ctlsgl:=-round(setvolt*low_bound/5);
  DAOUT(DANUM,UPMODE,ctlsgl);
end;


procedure GET_DISP( N_SAMPLES       : integer;
                        DAT            : glns5;
                    var DISP        : real);

var
     i               : integer;

begin
      disp:=0;
      for i:=0 to (N_SAMPLES - 1) do begin
      disp:=disp+dat[i];
      end;
      disp:=disp/n_samples;

end; { of get disp }


procedure TURN_OFF_SYSTEM;
begin
  setvolt:=-Offset/4;
  ctlsgl:=round(setvolt*up_bound/5);
  DAOUT(DANUM,UPMODE,ctlsgl);
end;

{-------------------------------- MAIN PROGRAM -------------------------------}


begin
   SYSTEM_INIT;
   TRIGGER(TRIG,TVALUE);
   repeat
      clrscr;
      SET_CLOCK;
      FASTAD(CH1,GAIN1,FUNI1,NS1,NCHAN1,DAT1[0]);
      GET_DISP(NS1,DAT1,DISP);
      Cn:=disp/409.5; Cn_1:=Cn; Cn_2:=Cn; Ms:=Offset;
      Rn:=Cn+2.0; st:='y';
```

```
      for i:=1 to 500 do
        begin
         { if (Cn<Rn) and (st='y') then PID
          else
            begin
              st:='n';
              CO_PID;
            end; } PID;
          FASTAD(CH1,GAIN1,FUNI1,NS1,NCHAN1,DAT1[0]);
          GET_DISP( NS1,DAT1,disp);
          Cn:=disp/409.5;
        end;
      writeln('   >>> END OF TESTING !!! ');

      write(' CONTINUE ? <y/n> '); readln(continue);
      TURN_OFF_SYSTEM;
      close(fo);
    until continue='n';

END.
```

APPENDIX R

VSS CONTROL PROGRAM FOR EXPERIMENT

```
PROGRAM SERVO_LINE_CONTROL;

const
   Offset = 2.4;
   alpa1=100;
   beta1=-160;
   alpa2=10;
   beta2=-20;
   c1=28;
   c2=6;

type
   IOARRAY=Array[1..10,1..10] of Real;
   SYARRAY=Array[1..10] of Real;
   glns2= array [0..2000] of integer;
   glns5= array [0..500] of integer;
   outarray = array [0..1000] of real;


var
   BASE,TRIG,TVALUE,SAMPLE_RATE,
   DANUM,UPMODE,COUNT,BVALUE,
   HOUR,MINUTE,SECOND,TENTHS,
   secSTART,secEND,tenthsSTART,tenthsEND,
   CH1,GAIN1,FUNI1,NS1,NCHAN1,
   CH2,GAIN2,FUNI2,NS2,NCHAN2,
   CH3,GAIN3,FUNI3,NS3,NCHAN3,
   CH4,GAIN4,FUNI4,NS4,NCHAN4
                           : integer;
   current,setvolt,disp,pressure,
   time,vel,totsec,timestep,
   X1,X2,X3,fX1,fX2,U,R,
   thigma,ftime,ntime,
   ks1,ks2,dt
                           : real;
   LVDT,PRESS              : outarray;
   DAT1,DAT2,DAT6,DAT7     : glns5;
   DAT3,DAT4               : glns2;
   i,j,up_bound,
   low_bound,ctlsgl
                           : integer;   { for control signal from ADALAB }
   continue                : char;
   filename                : string[10];
   fo                      : text;


procedure INIT(var BASE: integer); EXTERNAL 'ADAPTP.COM';
procedure TRIGGER( var TRIG,VALUE: integer); EXTERNAL INIT[48];
procedure FASTAD( var CHAN,GAIN,FUNI,NS,NCHAN,DATA: integer);EXTERNAL INIT[42];
procedure DAOUT( var DANUM,UPMODE,COUNT: integer); EXTERNAL INIT[33];
procedure CLOCKIN( var HOUR,MINUTE,SECOND,TENTHS: integer); EXTERNAL INIT[66];
procedure CLOCKOUT( var HOUR,MINUTE,SECOND,TENTHS: integer); EXTERNAL INIT[69];


procedure SYSTEM_INIT;
```

```
begin
    ClrScr;
    BASE:=768;
    INIT(BASE);

    SAMPLE_RATE:=16000 ; { per second }
    TRIG:=1;
    TVALUE:=ROUND(1193210.0/SAMPLE_RATE - 4) div 2;
    { >>>>  D/A channel A <<<< }
    DANUM:=0; { bipolar at A }
    UPMODE:=1;
    COUNT:=0;  { digital voltage output }
    up_bound:=2047;
    low_bound:=-2048;
    DAOUT(DANUM,UPMODE,COUNT);


    { >>>>  channal #1  <<<< }
    CH1:=1;   { displacement by LVDT }
    GAIN1:=1;
    FUNI1:=1;
    NS1:=10;
    NCHAN1:=1;

    { >>>>  channal #2  <<<< }
    CH2:=2;   { pressure }
    GAIN2:=1;
    FUNI2:=1;
    NS2:=10;
    NCHAN2:=1;


end; { of initiation }

procedure SET_CLOCK;
begin
  hour:=0;
  minute:=0;
  second:=0;
  tenths:=0;
  CLOCKOUT(hour,minute,second,tenths);
end;


procedure VSS;

begin
  thigma:=C1*X1+C2*X2+X3;
  if X1*thigma>0 then ks1:=alpa1;
  if X1*thigma<0 then ks1:=beta1;
  if X2*thigma>0 then ks2:=alpa2;
  if X2*thigma<0 then ks2:=beta2;
  current:=ks1*X1+ks2*X2;
  if current>15 then current:=10;
```

```
  if current<-15 then current:=-10;
  setvolt:=-(current+Offset)/4;
  ctlsgl:=round(setvolt*up_bound/5);
{
  if ctlsgl>up_bound then ctlsgl:=up_bound;
  if ctlsgl<low_bound then ctlsgl:=low_bound;
}
 DAOUT(DANUM,UPMODE,ctlsgl);
end;


procedure GET_DISP( N_SAMPLES       : integer;
                        DAT             : glns5;
                    var DISP        : real);

var
     i                  : integer;

begin
      disp:=0;
      for i:=0 to (N_SAMPLES - 1) do begin
      disp:=disp+dat[i];
      end;
      disp:=disp/n_samples;

end; { of get disp }


procedure TURN_OFF_SYSTEM;
begin
  setvolt:=-0/4;
  ctlsgl:=round(setvolt*up_bound/5);
  DAOUT(DANUM,UPMODE,ctlsgl);
end;

{-------------------------------- MAIN PROGRAM -------------------------------}


begin
   SYSTEM_INIT;
   TRIGGER(TRIG,TVALUE);
   repeat
      clrscr;
      SET_CLOCK;
      FASTAD(CH1,GAIN1,FUNI1,NS1,NCHAN1,DAT1[0]);
      GET_DISP(NS1,DAT1,DISP);
      R:=disp/409.5;
      U:=R+1.5;
      X1:=U-R; X2:=0; X3:=0; fX1:=0; fX2:=0;
      CLOCKIN(hour,minute,second,tenths);
      ftime:=second+tenths/10;
      for i:=1 to 300 do
        begin
          VSS;
          FASTAD(CH1,GAIN1,FUNI1,NS1,NCHAN1,DAT1[0]);
```

```
          GET_DISP( NS1,DAT1,disp);
          R:=disp/409.5;
          dt:=0.026;
          X1:=U-R; X2:=(X1-fX1)/dt; X3:=(X2-fX2)/dt;
          fX1:=X1; fX2:=X2; ftime:=ntime;
        end;
      writeln('   >>> END OF TESTING !!! ');

      write(' CONTINUE ? <y/n> '); readln(continue);
      TURN_OFF_SYSTEM;
      close(fo);
    until continue='n';

END.
```

VITA

Jaecheon Lee

Candidate for the Degree of

Master of Science

Thesis: AN INVESTIGATION INTO THE COMPUTER CONTROL OF TRANSMISSION LINE DYNAMICS FOR A SERVO - CONTROLLED HYDRAULIC SYSTEM

Major Field: Mechanical Engineering

Biographical:

Personal Data: Born in Seoul, Republic of Korea, November 23, 1957, the son of Bo-Hyung and Keun-Young Lee.

Education: Graduated from Jungang High School, Seoul, Korea, in February 1976; received Bachelor of Science Degree in Agricultural Engineering from Seoul National University at Seoul in February, 1980; received Master of Science Degree in Production Engineering from Korea Advanced Institute of Science and Technology at Seoul in February, 1985; completed requirements for the Master of Science Degree at Oklahoma State University in December, 1989.

Professional Experience: Lieutenant (Platoon Leader), Korean Army, March, 1980, to June, 1982; Research Engineer (Assistant Section Chief), Technical Research Institute, Hyundai Precision and Industry Inc., LTD, Yongin-Kun, Kyungki-do, Korea, March, 1985, to July, 1988.