A SINGLE INPUT, SINGLE OUTPUT TIME-VARYING DIGITAL CONTROLLER FOR MULTIVARIABLE SYSTEMS

BY

VINDA P. KOTWAL

Submitted in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in the

Electrical Engineering

Program

Robert H Foulkes 12-2-92

Advisor Date

Sally M. Hotchkiss December 3, 1992

Dean of the Graduate School Date

YOUNGSTOWN STATE UNIVERSITY

DECEMBER, 1992

Y-10-4
3

ABSTRACT

A SINGLE-INPUT, SINGLE-OUTPUT, TIME-VARYING DIGITAL CONTROLLER DESIGN FOR MULTIVARIABLE SYSTEMS

Vinda P.Kotwal

Master of Science in Electrical Engineering

Youngstown State University,1992.

In this thesis, a step-varying design technique is applied to a MIMO continuous time-invariant system. A general method of updating one input at a time and measuring one output at that time, is presented to convert MIMO continuous time-invariant plant into SISO, periodic, step-varying system. A step-varying controller consisting of a step-varying state feedback design and a step-varying estimator design is developed. This step-varying design is applied in real-time for an application example and real-time results are compared with those of simulation results.

TO MY AND PRAKASH'S PARENTS,

Who sacrificed so much in life
to give me an excellent education
and made it possible for me to
continue my studies in U.S.

ACKNOWLEDGEMENTS

I have had the privilege of associating myself with my advisor Dr. Robert H. Foulkes, Jr., who has been a constant source of encouragement throughout my graduate program. I would like to thank him for enduring my persistent doubts and for his able guidance without which this thesis would never have been completed.

I would also like to offer my sincere thanks to Dr. Salvatore R. Pansino, and Dr. Jalal Jalali for their support and for serving on my thesis committee.

The faculty and staff of Electrical Engineering Department, especially Mrs. Anna Mae Serrecchio, have been indispensable by providing the necessary facilities for my thesis.

TABLE OF CONTENTS

PAGE

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Background

Classical or conventional methods for design and analysis of control systems are simple and require only a reasonable number of computations. But they are applicable only to linear time-invariant systems having a single input and a single output. They are based on the input-output relationship of the system, that is, a transfer function. Modern control systems may have many inputs and many outputs and may be time-varying and/or nonlinear. State space methods for analysis and design of control systems are better suited for such types of systems [1].

The state space method is based on the description of system equations in terms of n first-order differential equations (for continuous-time system) or difference equations (for discrete-time system), which may be combined into a first-order vector-matrix differential or difference equation. The use of vector-matrix notation greatly simplifies the mathematical representation of systems of equations. Design in state space can be carried out for a class of inputs, instead of a specific input function such as the impulse function, step function or sinusoidal function. It is also easy to see the response of the system to initial conditions using the state space method.

In the state space method of design, there are three types of variables that are involved in the modeling of dynamic systems: input variables, output variables, and state variables. For a linear continuous time-invariant plant, the state space model may be written as

$$\dot{x}(t) = A\,x(t) + B\,u(t) \;.........\textit{State Equation}$$

$$y(t) = C\,x(t) + D\,u(t) \;........\textit{Output Equation} \qquad (1.1)$$

where

x(t)	= n-vector	(state vector)
y(t)	= m-vector	(output vector)
u(t)	= r-vector	(input vector)
A	= n x n matrix	(state matrix)
B	= n x r matrix	(input matrix)
C	= m x n matrix	(output matrix)
D	= m X r matrix	(direct transmission matrix)

The state space model or any other mathematical model for the plant is usually obtained using physical laws governing the system.

When a digital controller is used to control a continuous-time plant, a discrete-time model of the plant must be obtained and then the appropriate design techniques must be applied in order to get an acceptable response of the controlled system. One discrete-time model for the above continuous-time plant has the following form:

$$x((k+1)T) = \Phi\, x(kT) + \Gamma\, u(kT)$$

$$y(kT) = C\, x(kT) + D\, u(kT)$$

where

$$\Phi = e^{AT} \quad , \quad \Gamma = \int_{0}^{T} e^{A\lambda}\, d\lambda\, B \quad , \quad \lambda = T - t$$

The above discrete-time model is obtained by assuming that all the components of the input u(t) are sampled and held constant over the interval between any two consecutive sampling instants. It is also assumed that all the inputs and outputs are sampled simultaneously, and that the sampling is periodic. Since the coefficient matrices in the above equations are constants, i.e., not functions of the time step k, the system is called a discrete time-invariant system or step-invariant system.

Once the mathematical model of the system is determined, a designer can apply different design techniques to get an acceptable response of the controlled system. If there is a requirement that one or more outputs of the system follow a specific reference input or inputs, then the control system design is called a tracking system design or servo system design.

In a tracking system design, the objective is for the system outputs to track (follow) an equal number of reference input signals. Regulator design is a special case of tracking in which the desired tracking value is zero. In that case, the objective is to bring the system tracking outputs near zero in an acceptable manner, often in the presence of disturbances. Thus, in tracking system design, a designer must design a controller not only to have specific tracking outputs but also to have an acceptable transient response in presence of disturbances or initial plant state [2].

The tracking system design may be carried out in two steps. In the first step, the initial plant state or disturbance effect is forced to zero, and, in the second step, the tracking outputs are forced to track reference inputs. The first step is referred to as regulator design, and the response of the controlled system is called a zero-input response, since the reference inputs are zero. The second step is called tracking system design and the response of the controlled system is called a zero-state response, since plant state is zero.

To obtain an acceptable zero-input response, a pole placement design technique or optimal control method can be used. In these methods, all state variables are fed back to form the control signal; such designs are called full state variable feedback (SVFB) designs. In practical control systems, however, measurement of all state variables may not be possible. Then, in order to implement a design based on state feedback, it becomes necessary to estimate the state variables by using a state observer.

Once the zero-input design is done, tracking system design is carried out using different methods, such as integral control, response model design, reference model design, or ideal tracking system design [2].

1.2 Objective

The purpose of this thesis is to design a digital controller for a multivariable analog system. We could use a discrete step-invariant model for a multiple-input multiple-output plant (MIMO) by sampling all the outputs simultaneously and updating all the inputs simultaneously at the same instant. This simultaneous sampling requires as many A/D converters as the number of outputs, as many D/A converters as the number of inputs, and synchronization of all sampling. Because of hardware limitation in the control laboratory, an attempt has been made to sample only one output at a time and update only one input at the same time. This specific way of sampling results in a single-input, single-output (SISO) discrete time-varying system. In this case the coefficient matrices are function of the time step k, so the system is also called a step-varying system.

The pole-placement method of design for obtaining desirable performance cannot be used for a step-varying system. Gene Hostetter has introduced step-varying controller techniques that can be used for discrete step-varying systems [2]. These techniques require a modest addition in knowledge and effort. The implementation of step-varying control requires a bit more complexity than the step-invariant control, a small price to pay for the ability to control step-varying plants. The additional design freedom can be used to improve controller performance for step-invariant plants.

The objective of this thesis is to apply Hostetter's design technique to the SISO step-varying model of a MIMO analog system, and to implement the design in hardware in the control laboratory.

1.3 Overview

In chapter two, basic ideas about step-varying systems and step-varying controllers are explained in detail. Application of the design technique for a specific MIMO system is carried out in chapter 3. The appropriate hardware and software for real time implementation of control system are also explained in chapter 3. Chapter 4 contains results and conclusions.

CHAPTER II

STEP-VARYING CONTROLLER THEORY

2.1 Introduction

In this chapter, we present a procedure for designing a SISO digital controller for a MIMO plant. When implemented on a single-processor computer, such a design allows the processor to handle the I/O and to evaluate the control algorithm in a simple, efficient manner. Using a SISO design necessitates multiplexing input-output samples, and hence results in a time- (or step-) varying system model of the MIMO plant. In section 2.2 some aspects of step-varying systems are presented. In section 2.3, a technique for modeling a time-invariant MIMO plant as a step-varying SISO discrete system is developed. Finally, the design of a Step-varying controller is covered in section 2.4.

2.2 Step-Varying Systems

In this section, some general results about step-varying discrete systems are introduced. The presentation here follows closely that in Hostetter [2]. Consider a discrete step-varying system having a state space model of the form,

$$x(k+1) = \Phi(k)\ x(k) + \Gamma(k)\ u(k) \quad \textit{State Equation} \quad \ldots.(2.1)$$

$$y(k) = C(k)\ x(k) + D(k)\ u(k) \quad \textit{Output Equation} \quad \ldots.(2.2)$$

The coefficient matrices each may vary with step as indicated by $\Phi(k)$, $\Gamma(k)$, $C(k)$, $D(k)$. The vector signal delay diagram for the system is given in Figure 2.1.

For step-invariant systems, the coefficient matrices Φ, Γ, C, D are constant. Some of the ideas and results for step-invariant systems have counterparts for step-varying systems; the Z-Transform, however, is of no help in finding the response of a step-varying system. Also matrix eigenvalues do not have same significance for step-varying systems as they do for step-invariant ones.

The response of a discrete step-varying system can be calculated recursively from the state equation (2.1), the initial state x(0), and the input u(k) as follows.

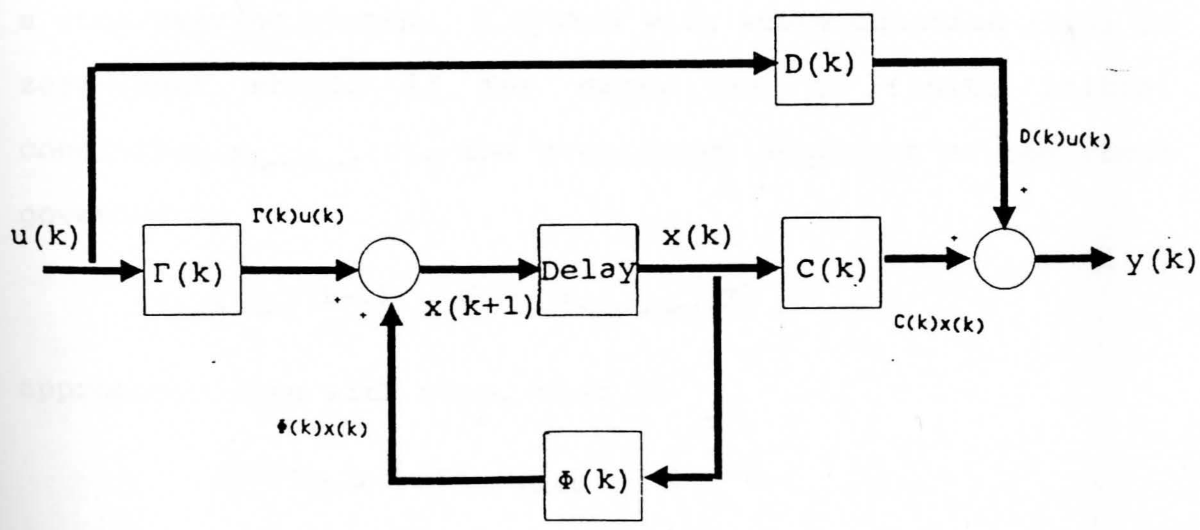

Figure 2.1: Block diagram for a Step-Varying System.

$$x(1) = \Phi(0)\ x(0) + \Gamma(0)\ u(0)$$

$$x(2) = \Phi(1)\ x(1) + \Gamma(1)\ u(1)$$

$$= \Phi(1)\ \Phi(0)\ x(0) + \Phi(1)\ \Gamma(0)\ u(0) + \Gamma(1)\ u(1)$$

$$\vdots$$

$$\begin{aligned} x(k) = {} & \Phi(k-1)\ \Phi(k-2) \ldots \Phi(0)\ x(0) + \Phi(k-1) \ldots \Phi(1)\ \Gamma(0)\ u(0) \\ & + \Phi(k-1) \ldots \Phi(2)\ \Gamma(1)\ u(1) + \ldots\ldots \\ & + \Phi(k-1)\ \Gamma(k-2)\ u(k-2) + \Gamma(k-1)\ u(k-1) \end{aligned}$$

$$= \prod_{i=0}^{k-1} \Phi(i)\ x(0) + \sum_{i=0}^{k-1} \prod_{j=i+1}^{k-1} \Phi(j)\ \Gamma(i)\ u(i)$$

$$\vdots$$

2.2.1 Stability of a Step-Varying System

This section presents a basic concept of stability for a step-varying system. A system with state equation (2.1) is zero-input stable if for every set of finite initial conditions $x_{zero\text{-}input}(0)$, the zero-input component of the state governed by

$$x_{zero\text{-}input}(k+1) = \Phi(k)\ x_{zero\text{-}input}(K)$$

approaches zero with step, that is

$$limit_{(k \to \infty)} \parallel x_{zero\text{-}input}\ (k) \parallel = 0,$$

where the double bars indicate the Euclidean vector norm.

The system is zero-state stable if and only if for zero initial conditions and every bounded input

$\| u(k) \| < \delta$; $k = 0,1,2.....$

the zero-state component of the state, governed by

$$x_{zero\text{-}state}(k+1) = \Phi(k)\, x_{zero\text{-}state}(k) + \Gamma(k)\, u(k)$$

$$x_{zero\text{-}state}(0) = 0$$

is bounded, that is

$$\| x_{zero\text{-}state}(k) \| < \rho\,, \quad k = 0,1,2....$$

A linear discrete time system is stable if it is both zero-input and zero-state stable. Having defined stability, it may not be an easy matter to determine whether a given step-varying system is stable. But, whether the system is stable or not, there is the possibility of using feedback to obtain an acceptable response.

2.2.2 Step-Varying Feedback Control

Hostetter [2] has developed a design procedure for using step-varying feedback control to shape the zero-input response of a step-varying plant. Since the design developed in this paper is an extension of Hostetter's approach, the basics of Hostetter's approach are reviewed here.

Consider a single-input, n^{th}-order step-varying plant with following state equation,

$$x(k+1) = \Phi(k)\, x(k) + \Gamma(k)\, u(k)$$

for which we need to have an acceptable zero-input response.

The SVFB (state variable feedback) control law is given by

$$u(k) = -F(k)\ x(k)$$

where F(k) is a step-varying feedback gain matrix. The feedback system is

$$x(k+1) = [\Phi(k) - \Gamma(k)\ F(k)]\ x(k)$$

Then zero-input response of the feedback system is

$$x_{zero\text{-}input}(k) = A(k-1)\ A(k-2) \ldots . A(0)\ x(0)$$

$$\textit{where} \quad A(i) = \Phi(i) - \Gamma(i)\ F(i)$$

To have the zero-input response of the feedback system decay to zero in n steps and beyond, it is required that

$$A(n-1)\ A(n-2)\ \ldots .. A(0) = 0$$

This condition can be met if

$$j'_1\ A(n-1)\ A(n-2) \ldots .. A(0) = 0$$

$$j'_2\ A(n-1)\ A(n-2) \ldots .. A(0) = 0$$

$$\vdots$$

$$j'_n\ A(n-1)\ A(n-2) \ldots .. A(0) = 0 \qquad \ldots \ldots ..(2.3)$$

where $j_1, j_2, \ldots j_n$ are any n linearly independent n-vectors. These are called basis vectors because they span the n dimensional state space. Relation (2.3) is in turn obtained if

$$j'_1 \, A(n-1) = 0$$

$$j'_2 \, A(n-1) \, A(n-2) = 0$$

$$\vdots$$

$$j'_n \, A(n-1) \, A(n-2) \ldots\ldots A(0) = 0 \qquad \ldots\ldots(2.4)$$

The first equation in relation (2.4) is in terms of feedback gain F(n-1),

$$i.\,e \qquad j'_1 \, A(n-1) = j'_1 \, [\Phi(n-1) - \Gamma(n-1) F(n-1)] = 0$$

$$\therefore \quad F(n-1) = \frac{j'_1 \, [\Phi(n-1)]}{j'_1 \, \Gamma(n-1)}$$

Similarly the second equation of relation (2.4) gives

$$F(n-2) = \frac{j'_2 \, A(n-1) \, \Phi(n-2)}{j'_2 \, A(n-1) \, \Gamma(n-2)}$$

In general, for k^{th} step

$$F(k) = \frac{j'_{n-k} \, A(n-1) \, \ldots . A(k+1) \, \Phi(k)}{j'_{n-k} \, A(n-1) \ldots . A(k+1) \, \Gamma(k)}$$

These computations proceed backward in step.

After n steps, the zero-input response will ideally been driven to zero. However, if there are inaccuracies in the equations or disturbances acting on the plant, it is required to drive the response towards zero on succeeding steps. This can be done by computing gain matrices beyond n steps by repeating the algorithm for every group of n steps. Thus

$$F(2n-1) = \frac{j'_1\,\Phi(2n-1)}{j'_1\,\Gamma(2n-1)},$$

$$F(2n-2) = \frac{J'_2\,A(2n-1)\;\Phi(2n-2)}{j'_2\,A(2n-1)\;\Gamma(2n-2)},$$

and so on.

The feedback system designed above is called a deadbeat design since the zero input response is driven to zero in n steps.

2.2.3 Step-Varying Observer

The design method presented above utilizes the feedback of all the state variables. In many practical cases, only a few state variables of a system are measurable and the rest are not measurable. Hence it is necessary to estimate the state variables that are not directly measurable. A state observer, also called as a state estimator, is a subsystem in the control system that performs an estimation of the state variables based on the measurements of the outputs and the control variables. A full-order state observation means that we observe (estimate) all n state variables regardless of whether some statse are available for direct measurement. This concept of observer of step-invariant plants is the same for step-varying ones. However, the eigenvalue placement method does not hold for step-varying case because the observer error is not generally governed by powers of a matrix for step-varying systems. Also, in the step-invariant case, continuous convergence of the observer is guaranteed, but for step-varying observers we must specifically design continued convergence; it is not automatic. A step-varying observer may also be desirable for a step-invariant plant because it can give an error response with a relatively small transient response amplitude.

For the n^{th} order step-varying plant (relation 2.1), a full-order state observer is another n^{th} order, linear, step-varying system of the form

$$\hat{x}(k+1) = \Phi(k)\ \hat{x}(k) + \Gamma(k)\ u(k) + P(k)\ [y(k) - \hat{y}(k)]$$

$$\hat{y}(k) = C(k)\ \hat{x}(k)$$

The observer gain sequence P(k) should be such that the observer state $\hat{x}(k)$ converges to that of the plant state x(k) in adequate steps. The error between the plant state and observer state is governed by

$$x(k+1) - \hat{x}(k+1) = [\Phi(k) - P(k)C(k)]\ [x(k) - \hat{x}(k)] \quad \ldots(2.5)$$

$$= A(k)\ [x(k) - \hat{x}(k)]$$

where

$$A(k) = \Phi(k) - P(k)C(k)$$

Therefore, at the n^{th} step, the error is

$$x(n) - \hat{x}(n) = A(n-1)\ A(n-2)\ \ldots\ldots A(0)\ [x(0) - \hat{x}(0)]$$

If P(0), P(1),....P(n-1) are chosen so that

$$A(n-1)\ A(n-2)\ \ldots\ldots A(0) = 0 \quad \ldots\ldots\ldots(2.6)$$

then the observer state will be equal to the plant state at the n^{th} step and beyond.

The desired relation (2.6) is obtained if

$$A(n-1)\,A(n-2)\,\ldots.A(0)\,j_1 = 0$$

$$A(n-1)\,A(n-2)\ldots.A(0)\,j_2 = 0$$

$$\vdots$$

$$A(n-1)\,A(n-2)\ldots.A(0)\,j_n = 0 \qquad \ldots\ldots.(2.7)$$

where $j_1,\ldots.j_n$ are any n linearly independent n-vectors.

The relation (2.7) is in turn obtained, if

$$A(0)\,j_1 = 0$$

$$A(1)\,A(0)\,j_2 = 0$$

$$\vdots$$

$$A(n-1)\,A(n-2)\ldots.A(0)\,j_n = 0 \qquad \ldots\ldots.(2.8)$$

The first equation of relation (2.8) is

$$[\Phi(0) - P(0)C(0)]\,j_1 = 0$$

$$\therefore\ P(0) = \frac{\Phi(0)\,j_1}{C(0)\,j_1}$$

$$then\quad A(0) = \Phi(0) - P(0)C(0)$$

From the second equation,

$$[\Phi(1) - P(1)C(1)]\,A(0)\,j_2 = 0$$

$$\therefore\ P(1) = \frac{\Phi(1)\,A(0)\,j_2}{C(1)\,A(0)\,j_2}$$

$$\text{then} \quad A(1) = \Phi(1) - P(1)C(1)$$

Continuing in this fashion, at the k^{th} step,

$$P(k) = \frac{\Phi(k)\ A(k-1)\ A(k-2)\ldots A(0)\ j_{k+1}}{C(k)\ A(k-1)\ A(k-2)\ldots A(0)\ j_{k+1}}$$

The resulting observer is termed deadbeat since its error is zero in n steps.

In n steps, the observer error will ideally be zero; however, if there are inaccuracies we have to drive any remaining error towards zero on succeeding steps. This is achieved by using the same algorithm for succeeding groups of n steps as,

$$P(n) = \frac{\Phi(n)\ j_1}{C(n)\ j_1}$$

$$\text{then} \quad A(n) = \Phi(n) - P(n)C(n)$$

and so on.

2.3 Conversion of MIMO plant into SISO system

Our aim is to design a digital controller for the continuous-time plant. Therefore we need to develop a discrete-time model of a continuous-time plant. A commonly used, linear, continuous, time-invariant system has the state space model:

$$\dot{x}(t) = A\,x(t) + B\,u(t)$$
$$y(t) = C\,x(t)$$

Assume that the continuous-time system is controllable and observable. A discrete-time model for this continuous-time system is given as

$$x((k+1)T) = \Phi\, x(kT) + \Gamma\, u(kT) \qquad \text{.......(2.9)}$$

$$y(kT) = C\, x(kT) \qquad \text{.......(2.10)}$$

Where

$$\Phi = e^{AT} \qquad \Gamma = \int_0^T e^{AT} dt\ .B$$

T = Sampling Period

Here coefficient matrices Φ, Γ and C are constants for the system. Use of this model assumes that we measure all the outputs simultaneously as well as update all the inputs at the same instant and hold these inputs constant for each sampling period. We want a SISO system, because we want to measure only one output at a time and update one input at the same time as shown in figure (2.2). Then, with respect to a digital controller, the plant to be controlled is SISO step-varying.

The sampling mechanism used here is demonstrated by considering a simple three-input, two-output, 5th order system as:

$$x(k+1) = \Phi\, x(k) + [\gamma_1\ \gamma_2\ \gamma_3] \begin{bmatrix} u_1(k) \\ u_2(k) \\ u_3(k) \end{bmatrix}$$

where $\gamma_1, \gamma_2, \gamma_3$, *are* 5x1 *column vectors.*

$$y(k) = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} x(k)$$

where c_1, c_2 *are* (1x5) *row vectors*

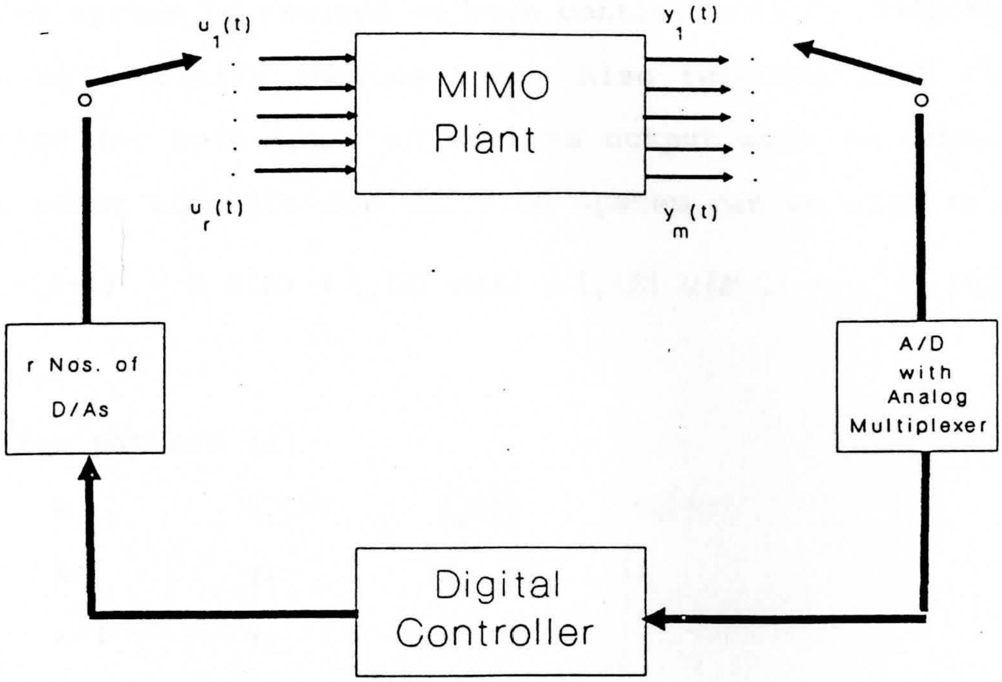

Figure 2.2: Sampling Mechanism for MIMO plant to convert it to SISO Step-Varying System.

To develop a SISO system we need to update only one input at a time and measure only one output at that time. The period of the SISO step-varying system can be made equal to the order (n=5) of the MIMO step-invariant system. This can be achieved if the input update pattern is repeated every five steps and the output measurement pattern is repeated every five steps. The input update pattern can be selected using controllability indices of each input. Also the output sampling pattern may be selected using observability indices of each output. This is evident from figure (2.3a) and figure (2.3b) which shows this sampling mechanism for three-input, two-output, fifth-order system for two different patterns. Note that the fifth order system is assumed to have controllability indices 2,2,1 and obervability indices 3,2. Also remember that sampling period for both input as well as output must be same. Then the state equation for the SISO system can be written as,

$$x(k+1) = \Phi\, x(k) + \xi_1(k)\, u(k) + \xi_2(k)\, u(k-1) + \xi_3(k)\, u(k-2)$$

where,

a) For pattern (a)

k	$\xi_1(k)$	$\xi_2(k)$	$\xi_3(k)$
k	γ_1	γ_3	γ_2
k+1	γ_1	γ_1	γ_3
k+2	γ_2	γ_1	γ_1
k+3	γ_2	γ_2	γ_1
k+4	γ_3	γ_2	γ_2

and so on.

b> For pattern (b)

k	$\xi_1(k)$	$\xi_2(k)$	$\xi_3(k)$
k	γ_1	γ_2	γ_1
k+1	γ_2	γ_1	γ_2
k+2	γ_3	γ_2	γ_1
k+3	γ_1	γ_3	γ_2
k+4	γ_2	γ_1	γ_3

and so on.

Note that ξ_i has the same dimensions (5x1) as that of γ_i (i=1,2,3). Here u(k) is the input which is updated at a particular instant. Also, for simplicity, the sampling period

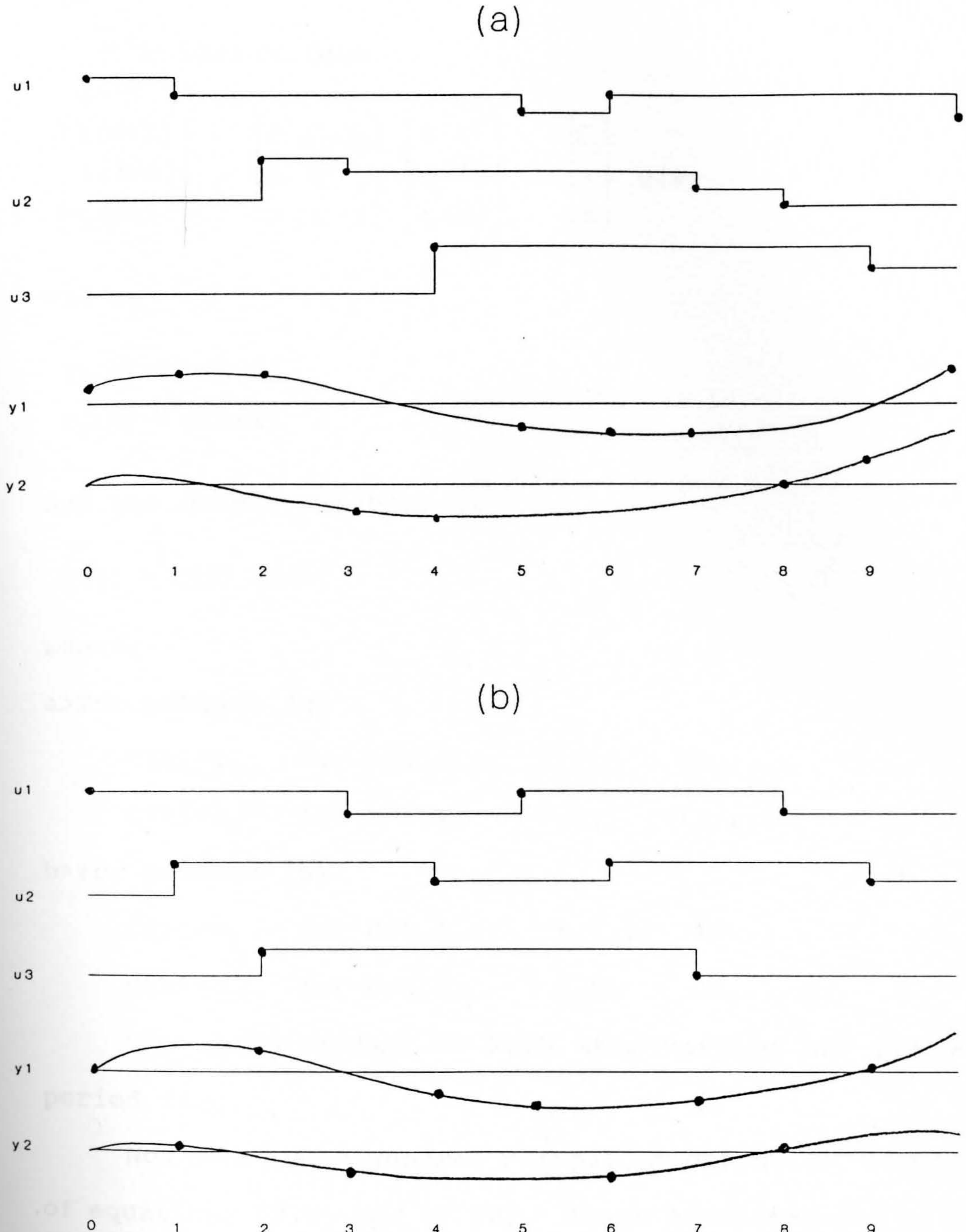

Figure 2.3: Sampling Mechanism for obtaining the period for SISO System equal to the order of the MIMO System. Two patterns for 3 inputs, 2 outputs, fifth order System are shown in (a) and (b).

T is omitted from the equation; thus x(k) means x(kT), and so on.

In matrix form,

$$\begin{bmatrix} x(k+1) \\ v_1(k+1) \\ v_2(k+1) \end{bmatrix} = \begin{bmatrix} \Phi & \xi_2 & \xi_3 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ v_1(k) \\ v_2(k) \end{bmatrix} + \begin{bmatrix} \xi_1 \\ 1 \\ 0 \end{bmatrix} u(k)$$

where,

$$v_1(k) = u(k-1)$$
$$v_2(k) = u(k-2)$$

And the output equation is,

$$y(k) = C(k)\; x(k)$$

where,

a>For pattern (a)

$C(k)=c_1$ for k=0,1,2, 5,6,7, 10,....

$C(k)=c_2$ for k=3,4, 8,9, 13,....

b>For pattern (b)

$C(k)=c_1$ for k=0,2,4, 5,7,9, 10,....

$C(k)=c_2$ for k=1,3, 6,8, 11,....

The above system is SISO step-varying and periodic of period five.

Now consider a general r-input, m-output, n^{th} order system of equations (2.9) and (2.10). These equations can be written as

$$x((k+1)T) = \Phi\, x(k) + [\gamma_1\ \gamma_2\ \ldots\ \gamma_r] \begin{bmatrix} u_1(kT) \\ u_2(kT) \\ \cdot \\ \cdot \\ u_r(kT) \end{bmatrix}$$

where γ_i*s are columns of* $\Gamma(k)$ $(i=1,2\ldots r)$

$$y(kT) = \begin{bmatrix} c_1 \\ c_2 \\ \cdot \\ \cdot \\ c_m \end{bmatrix} x(kT)$$

where c_i*s are rows of* $C(k)$ $(i=1,2\ldots m)$

Since we are updating the inputs one at a time, the state equation for r-input n^{th} order plant is,

$$x(k+1) = \Phi\, x(k) + \xi_1(k)\, u(k) + \xi_2(k)\, u(k-1) + \xi_3(k)\, u(k-2) + \ldots + \xi_{r-1}(k)\, u(k-(r-2)) + \xi_r(k)\, u(k-(r-1)) \quad (2.11)$$

where the values of ξ_1, ξ_2, ...ξ_r depend on the input update pattern and on the controllability indices of the MIMO plant.

The equation (2.11) can be written as,

$$x(k+1) = \Phi\, x(k) + \xi_1(k)\, u(k) + \xi_2(k)\, v_1(k) + \xi_3(k)\, v_2(k) + \ldots\ldots + \xi_{r-1}(k)\, v_{r-2}(k) + \xi_r(k)\, v_{r-1}(k)$$

where,

$v_1(k) = u(k-1)$ *i.e.* $v_1(k+1) = u(k)$

$v_2(k) = u(k-2)$ *i.e.* $v_2(k+1) = v_1(k)$

$v_3(k) = u(k-3)$ *i.e.* $v_3(k+1) = v_2(k)$

$\cdot$

$\cdot$

$$v_{r-2}(k) = u(k-(r-2)) \quad i.e. \quad v_{r-2}(k+1) = v_{r-3}(k)$$

$$v_{r-1}(k) = u(k-(r-1)) \quad i.e. \quad v_{r-1}(k+1) = v_{r-2}(k)$$

Therefore the state equation for the system in matrix form is

$$\begin{bmatrix} x(k+1) \\ v_1(k+1) \\ v_2(k+1) \\ \cdot \\ \cdot \\ v_{r-2}(k+1) \\ v_{r-1}(k+1) \end{bmatrix} = \begin{bmatrix} \Phi & \xi_2 & \xi_3 & \cdot & \cdot & \cdot & \xi_r \\ 0 & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 1 & 0 & \cdot & \cdot & \cdot & 0 \\ \cdot & & & & & & \\ \cdot & & & & & & \\ 0 & 0 & 0 & \cdot & \cdot & 1 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ v_1(k) \\ v_2(k) \\ \cdot \\ \cdot \\ v_{r-2}(k) \\ v_{r-1}(k) \end{bmatrix} + \begin{bmatrix} \xi_1 \\ 1 \\ 0 \\ \cdot \\ \cdot \\ 0 \end{bmatrix} u(k)$$

The output equation is

$$y(k) = C(k) \ x(k)$$

where C(k) depends on the output measurement pattern and the observability indices of the MIMO plant. The above system is SISO step-varying and periodic of period n.

2.4 Tracking System Controller Design

Tracking system design may have two steps:

a) zero-input response design (sometimes called regulator design), and

b) Zero-state response design (or steady-state response or tracking design).

In regulator design, the performance objective is to make the zero-input response of the system decay to zero in a desired manner whatever the initial conditions are. This is achieved using state variable feedback. In tracking system design, the objective is to make the tracking outputs track the reference inputs once the initial transients die out. Friedland [3] combined these two steps by defining error state variables and applying regulator design technique to these error state variables instead of state variables themselves. This reduces the need to design the two responses separately. Section 2.4.1 explains the concept of error state variable design. This idea is combined with Hostetter's idea of step-varying feedback in section 2.4.2.

2.4.1 Error state variable design

For a linear, continuous time-invariant system (relation 1.1), it is possible to completely specify the closed loop dynamic performance of the system by placing closed loop poles (eigenvalues) anywhere in the complex s-plane in principle. Practicability depends upon the particular plant, hardware available, cost, etc. The method used to achieve this is called pole-placement design [3].

In this method, a designer determines a gain matrix F in a linear feedback law,

$$u(t) = - F x(t)$$

which shapes the dynamic response of the process in the absence of reference inputs. The designer selects the feedback matrix F such that the eigenvalues of the closed loop dynamic matrix,

$$A_c = A - B\,F$$

are at desired locations.

In general the designer not only needs to achieve good zero-input response but also needs to design the control so that tracking outputs track the reference inputs. In order to achieve both the above objectives in one step, Friedland defines a error state vector as

$$e(t) = x(t) - x_r(t)$$

where x_r is reference input vector that is assumed to satisfy a differential equation,

$$\dot{x}_r(t) = A_r\, x_r(t)$$

called the reference model. The state equation becomes

$$\begin{aligned} \dot{e}(t) &= \dot{x}(t) - \dot{x}_r(t) \\ &= A\,e(t) + B\,u(t) + (A - A_r)\,x_r(t) \end{aligned}$$

and the control becomes

$$u(t) = -F\,e(t) - G\,x_r(t)\,.$$

The closed-loop system is shown in figure (2.4).
When the above control law is used in the process, the closed loop system becomes,

$$\dot{e} = (A - BF)\,e(t) + (A - A_r - BG)\,x_r(t)$$

Then the design objective is to choose the feedback gain matrix F so that the closed loop system is asymptotically stable and to choose the feedforward gain matrix G so that a

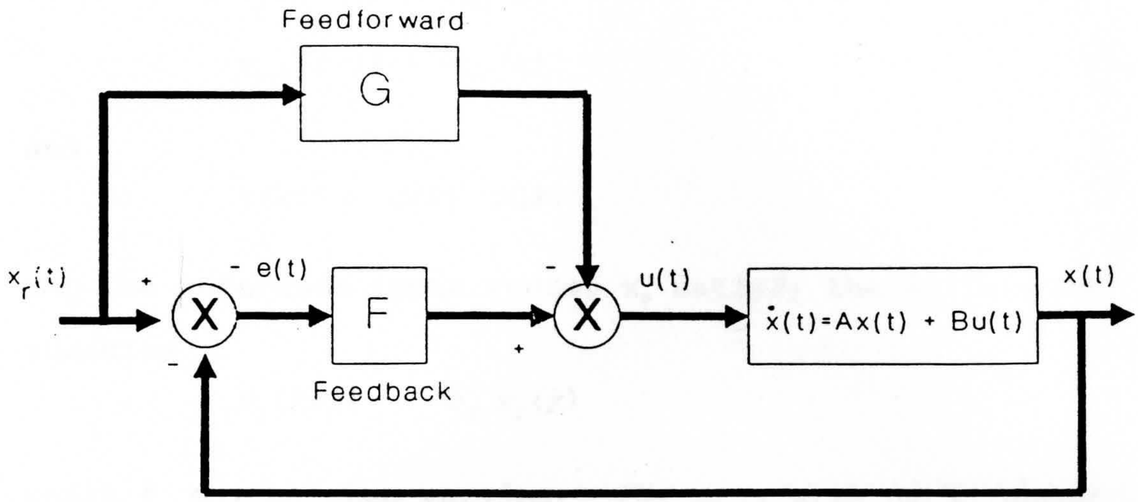

Figure 2.4: Block diagram for the System with Control Gains designed for Error State Variables.

linear combination of the error variables is zero under steady state conditions.

2.4.2 Step-Varying Error Design

From section 2.3, the state-variable model for the SISO step-varying system is

$$x(k+1) = \Phi\ x(k) + \xi_1(k)\ u(k) + \xi_2(k)\ v_1(k) + \xi_3(k)\ v_2(k)$$
$$+ \dots\dots\dots + \xi_{r-1}(k)\ v_{r-2}(k) + \xi_r(k)\ v_{r-1}(k)$$

where,

$$v_1(k+1) = u(k)$$

$$v_2(k+1) = v_1(k)$$

$$\vdots$$

$$v_{r-2}(k+1) = v_{r-3}(k)$$

$$v_{r-1}(k+1) = v_{r-2}(k)$$

and

$$y(k) = C(k)\ x(k)$$

Let the reference input vector x_r satisfy the difference equation

$$x_r(k+1) = \Phi_r\ x_r(k)$$

where Φ_r may be step-varying. The selection of Φ_r depends upon the specific reference input. The error vector is then given as

$$e(k) = x(k) - x_r(k)$$

$$\therefore\ e(k+1) = \Phi\ e(k) + \xi_1(k)\ u(k) + \xi_2(k)\ v_1(k) + \ldots + \xi_r(k)\ v_{r-1}(k) + [\ \Phi - \Phi_r\ ]\ x_r(k)$$

with

$$v_1(k+1) = u(k)$$

$$\vdots$$

$$v_{r-1}(k+1) = v_{r-2}(k)$$

Let the feedback law be

$$u(k) = -F(k)\ e(k) - l_1(k)\ v_1(k) - l_2(k)\ v_2(k) - \ldots \ldots - l_{r-1}(k)\ v_{r-1}(k) - G(k)\ x_r(k)$$

where

F(k) is a 1 x n feedback gain matrix,

G(k) is a 1 x n feedforward gain matrix, and

$l_1, l_2, .., l_{r-1}$ are scalar gains on the previous inputs.

In vector form, the closed-loop error state equations are

$$e(k+1) = A(k)\ e(k) + D(k)\ \overline{V}(k) + B(k)\ x_r(k)$$

$$\overline{V}(k+1) = -\ \overline{F}(k)\ e(k) - \overline{L}(k)\ \overline{V}(k) - \overline{G}(k)\ x_r(k),$$

and

$$x_r(k+1) = \Phi_r\ x_r(k).$$

where,

$$\overline{V}(k) = \begin{bmatrix} v_1(k) \\ v_2(k) \\ \cdot \\ \cdot \\ \cdot \\ v_{r-1}(k) \end{bmatrix}$$

$$A(k) = [\ \Phi - \xi_1(k)\ F(k)\ ]$$

$$D(k) = [\ (\ \xi_2(k) - \xi_1(k) l_1(k)\ )\ \ldots\ldots\ (\ \xi_r(k) - \xi_1(k) l_{r-1}(k)\ )\ ]$$

$$B(k) = [\ \Phi - \Phi_r - \xi_1(k)\ G(k)\ ]$$

$$\overline{L}(k) = \begin{bmatrix} l_1(k) & l_2(k) & . & . & . & l_{r-1}(k) \\ -1 & 0 & . & . & . & 0 \\ 0 & -1 & . & . & . & 0 \\ \cdot & & & & & \\ \cdot & & & & & \\ 0 & 0 & . & . & -1 & 0 \end{bmatrix}$$

$$\overline{G}(k) = \begin{bmatrix} G(k) \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

and

$$\overline{F}(k) = \begin{bmatrix} F(k) \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

To have the desired response, the error should be zero at step n and beyond, where n is the order of the MIMO discrete step-invariant system. At n^{th} step,

$$e(n) = A(n-1)\ e(n-1) + D(n-1)\ \overline{V}(n-1) + B(n-1)\ x_r(n-1)$$

Choose j_1 , $j_2, \ldots, j_n$ as the n basis vectors. To have error zero in all n directions,

$$\begin{aligned} j_1'\ e(n) &= 0 \\ &\vdots \\ j_n'\ e(n) &= 0 \end{aligned} \qquad \ldots\ldots.(2.12)$$

Consider first equation in the relation (2.12)

$$j_1'\ e(n) = j_1'\ [A(n-1)\ e(n-1) + D(n-1)\ \overline{V}(n-1) + B(n-1)\ x_r(n-1)\ ] = 0$$

$$\therefore \; j_1' A(n-1) = 0 \; , \quad \textit{which gives} \quad F(n-1) = \frac{j_1' \Phi}{j_1' \xi_1(n-1)}$$

$$\textit{Also} \quad j_1' D(n-1) = 0 \; , \quad \textit{which gives} \quad l_1(n-1) = \frac{j_1' \xi_2(n-1)}{j_1' \xi_1(n-1)}$$

$$l_2(n-1) = \frac{j_1' \xi_3(n-1)}{j_1' \xi_1(n-1)}$$

$$\vdots$$

and

$$l_{r-1}(n-1) = \frac{j_1' \xi_r(n-1)}{j_1' \xi_1(n-1)}$$

$$\textit{Finally} \quad j_1' B(n-1) = 0 \; , \quad \textit{which gives} \quad G(n-1) = \frac{j_1' (\Phi - \Phi_r)}{j_1' \xi_1(n-1)}$$

All the above gains can be used to calculate A(n-1), D(n-1), and B(n-1).

Considering second equation in relation (2.12), we have

$$\begin{aligned} j_2' e(n) = j_2' \, [&A(n-1)\, e(n-1) + D(n-1)\, \overline{V}(n-1) \\ &+ B(n-1)\, x_r(n-1)] = 0 \end{aligned}$$

Putting in the expressions for e(n-1), $\overline{v}(n-1)$, and $x_r(n-1)$ in terms of e(n-2), $\overline{v}(n-2)$, and $x_r(n-2)$ and simplifying, we get

$$\begin{aligned} j_2' e(n) = j_2' \, [&[A(n-1)\, A(n-2) - D(n-1)\, \overline{F}(n-2)]\, e(n-2) \\ &+ [A(n-1)\, D(n-2) - D(n-1)\, \overline{L}(n-2)]\, \overline{V}(n-2) \\ &+ [A(n-1)\, B(n-2) - D(n-1)\, \overline{G}(n-2) + B(n-1)\, \Phi_r]\, x_r(n-2) \end{aligned}$$

This gives

$$F(n-2) = \frac{j_2' \, A(n-1) \, \Phi}{j_2' \, [A(n-1) \, \xi_1(n-2) + d_1(n-1)]}$$

$$l_1(n-2) = \frac{j_2' \, [A(n-1) \, \xi_2(n-2) + d_2(n-1)]}{j_2' \, [A(n-1) \, \xi_1(n-1) + d_1(n-1)]}$$

$$l_2 \, (n-2) = \frac{j_2' \, [A(n-1) \, \xi_3(n-2) + d_3(n-1)]}{j_2' \, [A(n-1) \, \xi_1(n-1) + d_1(n-1)]}$$

$$\vdots$$

$$l_{r-1}(n-2) = \frac{j_2' \, [A(n-1) \, \xi_r(n-2) \,]}{j_2' \, [A(n-1) \, \xi_1(n-1) + d_1(n-1)]}$$

and

$$G(n-2) = \frac{j_2' \, [A(n-1) \, \Phi - A(n-1) \, \Phi_r + B(n-1) \, \Phi_r]}{j_2' \, [A(n-1) \, \xi_1(n-2) + d_1(n-1)]}$$

where $d_i(k)$ is the i^{th} column of $D(k)$.

In this way all the gains in step (n-2) are known. Proceeding in a similar way, expressions for the gain matrices for all the n steps are calculated. As noted earlier, the resulting design is deadbeat. After n steps, the error response will ideally remain zero. However, if there are model inaccuracies, disturbance inputs or sensor errors, it is necessary to drive the response towards zero on succeeding steps. This can be done by computing the gain matrices beyond n steps by using the same algorithm for every group of n steps as explained in section (2.2.4). The gain matrices obtained for the SISO step-varying system are also periodic. The period for gain matrices depends on the period of the SISO step-varying system. The SISO system period is the order of the MIMO system. Therefore, the period for the gain matrices is the minimum multiple of SISO system period n and the MIMO system period n, that is, the period for the gain matrices is also n. Therefore, the period for the gain matrices for the example given in the section (2.3) is the minimum multiple of 5 and 5, i.e., 5. This shows that it is preferable to have the SISO step-varying system period to be the order of the MIMO discrete step-invariant system. An iterative method for calculating the gain matrices is given in appendix A.

2.4.3 Observer Design

The state equation for the SISO step-varying system is,

$$\begin{aligned} x(k+1) &= \Phi\, x(k) + \xi_1(k)\, u(k) + \xi_2(k)\, v_1(k) + \xi_3(k)\, v_2(k) \\ &\quad + \dots\dots\dots + \xi_{r-1}(k)\, v_{r-2}(k) + \xi_r(k)\, v_{r-1}(k) \end{aligned}$$

Let the observer state equation be,

$$\begin{aligned} \hat{x}(k+1) &= \Phi\, \hat{x}(k) + \xi_1(k)\, u(k) + \xi_2(k)\, v_1(k) + \xi_3(k)\, v_2(k) \\ &\quad + \dots\dots\dots + \xi_r(k)\, v_{r-1}(k) + P(k)\, [y(k) - \hat{y}(k)] \end{aligned}$$

$$\text{where } \hat{y}(k) = C(k)\, \hat{x}(k)$$

Then the error between the system state equation and the observer state equation is,

$$x(k+1) - \hat{x}(k+1) = [\Phi - P(k)\, C(k)]\, [x(k) - \hat{x}(k)]$$

The above equation is exactly same as the observer error equation (2.5) given in section (2.2.3). Therefore expressions for the observer gain matrices in section (2.2.3) can be used to calculate observer gain matrices for the SISO step-varying system.

CHAPTER III

Application Design

3.1 Introduction

In this chapter, the state variable design technique discussed in chapter 2 is applied to a simple two-car train system. A mathematical model of the plant is briefly derived in section 3.2. Also this multivariable plant is converted into the SISO step-varying discrete-time system, and a step-varying controller is designed using the techniques explained in chapter 2. For the step-varying controller as well as the observer design, four sets of basis vectors are considered. Finally, section 3.3 gives a brief idea about the hardware and the software used for the real-time implementation of the design.

3.2. Two-Car Train Plant

In this section a mathematical model for the two-car train plant is developed. An idealized two-car train consisting of a pair of masses coupled by a spring (shown in figure 3.1), is the plant for which a controller is to be designed. The wheels of each car are independently driven by a direct current electric motor.

The parameters of the plant are as follow:

M ....Mass of each car

K ....Spring constant of the spring between two masses

k ....Motor torque constant

k_b ....Back emf constant

T ....Motor torque

f1....Linear force applied to the car 1

f2....Linear force applied to the car 2

r ....Ratio of the motor torque to the linear force=T/f

u_1....Input voltage to motor 1

u_2....Input voltage to motor 2

u_b....Back e.m.f. of a d.c. motor

i ....Armature current of a motor

R ....Resistance of a motor armature

ω ....Angular velocity of a motor.

Also signals of the plant are,

x_1 ...Displacement of car 1

x_3 ...Displacement of car 2

x_2 ...Velocity of car 1

x_4 ...Velocity of car 2

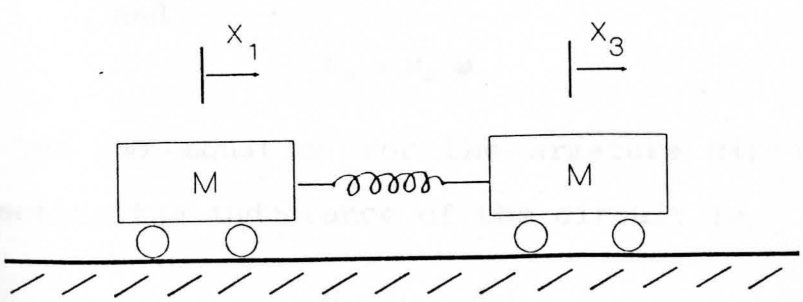

Figure 3.1: Two-Car Train Plant.

To derive the mathematical model for the plant, consider the mechanical part of the plant first. Applying Newton's second law of motion to the two masses we have

$$M \ddot{x}_1 + K (x_1 - x_3) = f_1$$

$$M \ddot{x}_3 + K (x_3 - x_1) = f_2$$

......(3.1)

Here a dot is used to indicate differentiation with respect to time.

Now consider an electrical system of the plant. An electrical motor is a device that converts electrical energy into mechanical energy. Here each electrical motor is assumed to be a separately excited, armature controlled, fixed-field direct current motor. The torque (T) developed by a motor is proportional to the armature current (i) of the motor since the field current is held constant. Also an induced back e.m.f.(u_b) is proportional to the angular velocity (ω) of a motor. Therefore,

$$T = k\, i$$

and

$$u_b = k_b\, \omega$$

The emf equation for the armature circuit of a motor neglecting the inductance of the circuit is,

$$u - u_b = R\, i$$

Putting the expressions for torque and back e.m.f. in the above equation and simplifying we get,

$$T = k/R\ [u - k_b\ \omega]$$

If the ratio of a motor torque to linear force applied to a car is r (=T/f), then a linear force applied by each motor is given as

$$f = (k/r\ R)\ [u - k_b\ \omega]$$

Also linear velocity $\dot{x} = r\ \omega$. Therefore final expression for a linear force is

$$f = (k/r\ R)\ [u - k_b\ \dot{x}/r]$$

Therefore for motor 1 and motor 2

$$f_1 = (k/r\ R)\ [u_1 - k_b\ \dot{x}_1/r]$$

$$f_2 = (k/r\ R)\ [u_2 - k_b\ \dot{x}_3/r] \qquad \ldots\ldots(3.2)$$

Using the relations 3.1 and 3.2, differential equations for car 1 and car 2 are

$$M\ddot{x}_1 + K\ (x_1 - x_3) = (k/r\ R)\ [u_1 - k_b\ \dot{x}_1/r]$$

$$M\ddot{x}_3 + K\ (x_3 - x_1) = (k/r\ R)\ [u_2 - k_b\ \dot{x}_3/r]$$

Rearranging these equations properly to get mathematical model in terms of first-order differential equations for the given plant, we have

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \ddot{x}_1 = (-K/M)\ x_1 + (K/M)\ x_3 + (k/MR_1\ r)\ u_1 - (kk_b/MR_1\ r^2)\ x_2$$

$\dot{x}_3 = x_4$

$\dot{x}_4 = \ddot{x}_3 = (K/M)\ x_1 - (K/M)\ x_3 + (k/MR_2\ r)\ u_2 - (kk_b/MR_2\ r^2)\ x_4$

Considering the displacements of the two cars as output variables, the state space model for the plant is,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -K/M & -kk_b/MR_1\ r^2 & K/M & 0 \\ 0 & 0 & 0 & 1 \\ K/M & 0 & -K/M & -kk_b/MR_2 r^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 \\ k/MR_1\ r & 0 \\ 0 & 0 \\ 0 & k/MR_2\ r \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

.......STATE EQUATION

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

.......OUTPUT EQUATION

Numerical data for the plant is as follows.

$M = 1\ Kg \qquad R = 100\Omega \qquad r = 4\ cm$

$K = 20\ \ N/m \qquad k = k_b = 2\ \ volts\text{-sec}$

Therefore the numerical state space model of the plant is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -20 & -25 & 20 & 0 \\ 0 & 0 & 0 & 1 \\ 20 & 0 & -20 & -25 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ .5 & 0 \\ 0 & 0 \\ 0 & .5 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

This can be written in compact form as

$$x(t) = A\,x(t) + B\,u(t)$$

$$y(t) = C\,x(t)$$

Analog computer simulation of the plant

In this section the two-car train plant is simulated on an analog computer. An analog computer solves a problem or provides responses of a plant by solving an analogous problem patched on the computer. The mathematical model of an analog computer programmed to simulate a specific physical system is identical to the mathematical model of the system. The input and output voltages (analog computer variables) are analogous to the corresponding physical variables of the system. Because of limitations of the analog computer or its associated input/output equipment, it is usually necessary to change the scale of the computer variables, thus forcing the values of a computer variable to differ from the corresponding problem variable values. More details about simulation of mathematical model of a plant on an analog computer can be found in the GP-6 analog computer manual [4].

To simulate the given plant on an analog computer, consider the differential equations of the plant

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = 20\ (x_3 - x_1) - 25\ x_2 + 0.5\ u_1$$

$$\dot{x}_3 = x_4$$

$$\dot{x}_4 = 20\ (x_1 - x_3) - 25\ x_4 + 0.5\ u_2$$

The analog computer simulation diagram is shown on next page (Figure 3.2). Because of limited resistances and capacitances available on the analog computer (comdyna GP-6), there is need to scale the inputs by a factor of 10 .
The system model with scaled inputs is given below.

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -20 & -25 & 20 & 0 \\ 0 & 0 & 0 & 1 \\ 20 & 0 & -20 & -25 \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0 \\ 5 & 0 \\ 0 & 0 \\ 0 & 5 \end{bmatrix} u(t)$$

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

The open-loop poles for the plant are 0.0, -1.718, -23.28, -25, which show that the plant is marginally stable. The zero-input response of the plant to the initial condition x(0)=[.5 0 0 0]' is shown in figure 3.3. Both the tracking outputs have smooth transients but they do not go to zero. The settling time of each response is 2.5 seconds which may not be desirable for some applications. This shows the need for a controller to the plant.

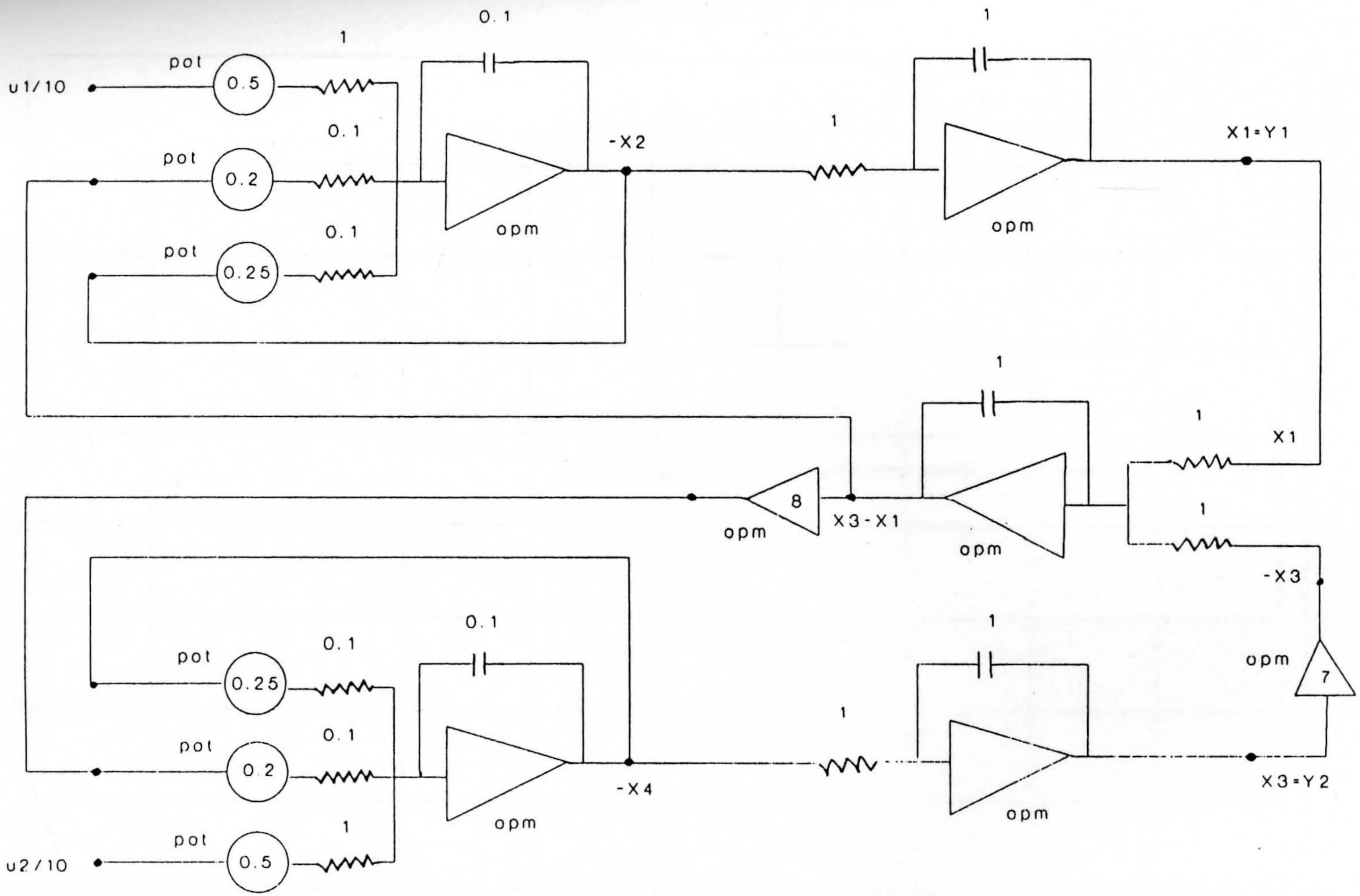

Figure 3.2: Analog Computer Simulation of the Plant.

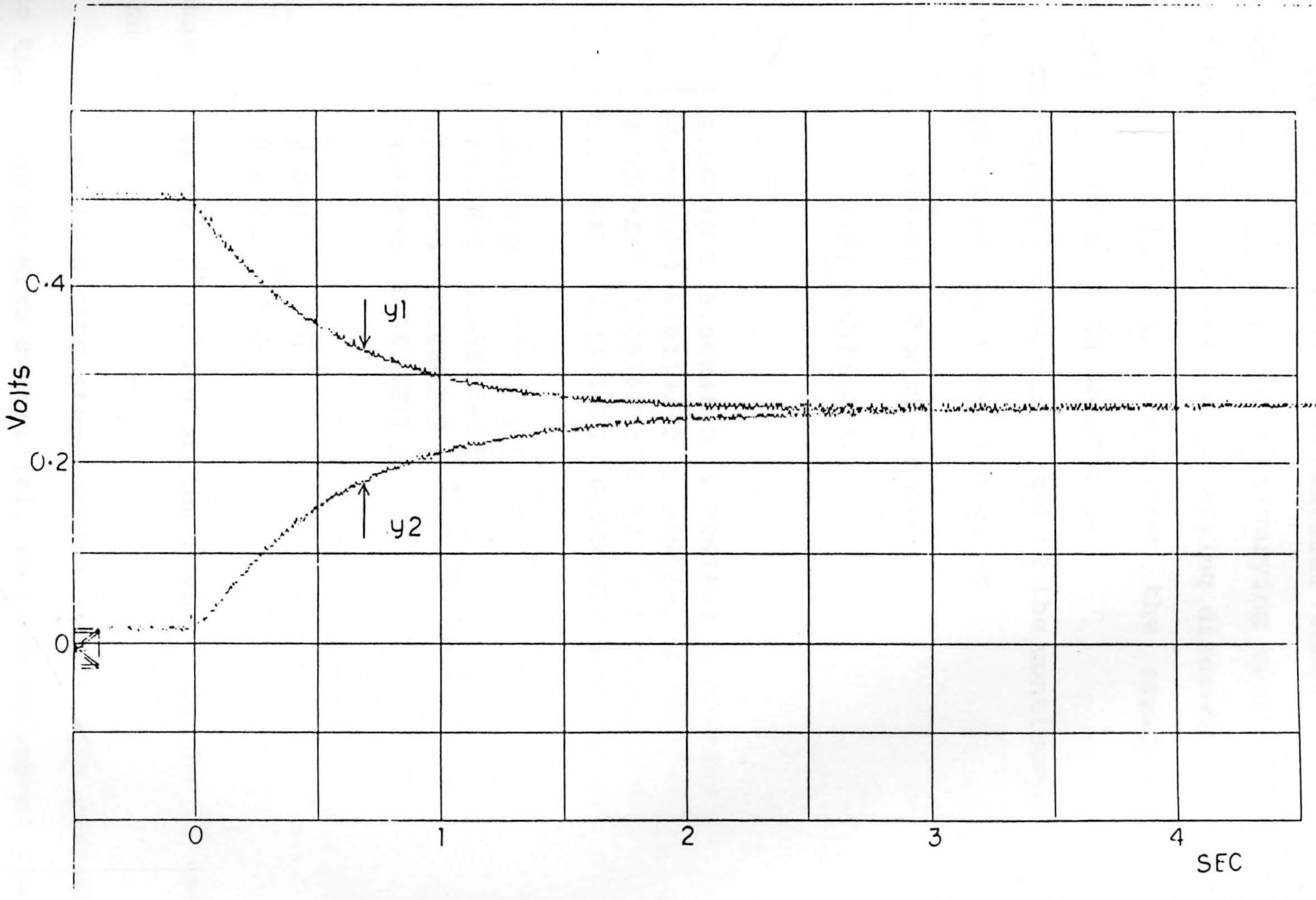

Figure 3.3: Real-Time Response of the Plant to x(0)=[0.5 0 0 0]'.

Discrete-time model of the plant

The scaled analog mathematical model of the plant is known now. To apply the step-varying controller design we need to obtain the SISO step-varying discrete-time model for the same plant. This requires the application of the technique discussed in chapter 2.

The MIMO discrete-time model of the continuous-time plant with sampling period T as .1 sec, is

$$x(k+1) = \Phi\, x(k) + \Gamma\, u(k)$$

$$y(k) = C(k)\; x(k)$$

where

$$\Phi = \begin{bmatrix} 9.5073E\text{-}1 & 3.5624E\text{-}2 & 4.9263E\text{-}2 & 1.0916E\text{-}3 \\ -6.9066E\text{-}1 & 6.0113E\text{-}2 & 6.9066E\text{-}1 & 2.1971E\text{-}1 \\ 4.9263E\text{-}2 & 1.0916E\text{-}3 & 9.5073E\text{-}1 & 3.5624E\text{-}2 \\ 6.9066E\text{-}1 & 2.1971E\text{-}2 & -6.9066E\text{-}1 & 6.0113E\text{-}2 \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} 1.2486E\text{-}2 & 1.7035E\text{-}4 \\ 1.7812E\text{-}1 & 5.4584E\text{-}3 \\ 1.7035E\text{-}4 & 1.2486E\text{-}2 \\ 5.4684E\text{-}3 & 1.7812E\text{-}1 \end{bmatrix} = [\ \gamma_1 \ \ \gamma_2\ ]$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

where γ_i is the i^{th} column of the Γ and c_i is the i^{th} row of the C.

To obtain a SISO step-varying system with period equal to the order of MIMO system (4), we need to repeat the input update pattern and output sampling pattern every 4 steps.

To choose these patterns, consider the controllability and observability indices of the MIMO discrete-time system. The controllability indices for inputs u_1 and u_2 of the MIMO discrete-time system are 2 each. Also the observability indices for outputs y_1 and y_2 of the MIMO discrete-time system are 2 each. Therefore we can update inputs u_1 and u_2 twice in 4 steps and measure outputs y_1 and y_2 twice in 4 steps. The sampling mechanism utilized in this thesis for the given system is shown in figure 3.4. Then the SISO discrete-time model for the continuous-time two-car train plant in state space form is,

$$x(k+1) = \Phi\, x(k) + \xi_1(k)\, u(k) + \xi_2(k)\, v(k)$$

$$v(k+1) = u(k)$$

$$y(k) = C(k)\, x(k)$$

Where v is previous input u. Also,

$$\begin{aligned} \xi_1(k) &= \gamma_1 , \quad \textit{for } k \textit{ even} \\ &= \gamma_2 , \quad \textit{for } k \textit{ odd} \end{aligned}$$

$$\begin{aligned} \xi_2(k) &= \gamma_2 , \quad \textit{for } k \textit{ even} \\ &= \gamma_1 , \quad \textit{for } k \textit{ odd} \end{aligned}$$

$$\begin{aligned} C(k) &= c_1 , \quad \textit{for } k \textit{ even} \\ &= c_2 , \quad \textit{for } k \textit{ odd} \end{aligned}$$

The discrete-time mathematical model for our continuous time-invariant two-car train plant in matrix form is given below.

$$\begin{bmatrix} x(k+1) \\ v(k+1) \end{bmatrix} = \begin{bmatrix} \Phi & \xi_2(k) \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ v(k) \end{bmatrix} + \begin{bmatrix} \xi_1(k) \\ 1 \end{bmatrix} u(k)$$

.......STATE EQUATION

$$y(k) = [\ C(k) \ \ 0\ ] \begin{bmatrix} x(k) \\ v(k) \end{bmatrix}$$

.... OUTPUT EQUATION

Note that it is SISO 5^{th}-order step-varying system of period two. Note that, because of the particular input update pattern and output measurement pattern, and values of controllability and observability indices, we have period of SISO system two instead of four.

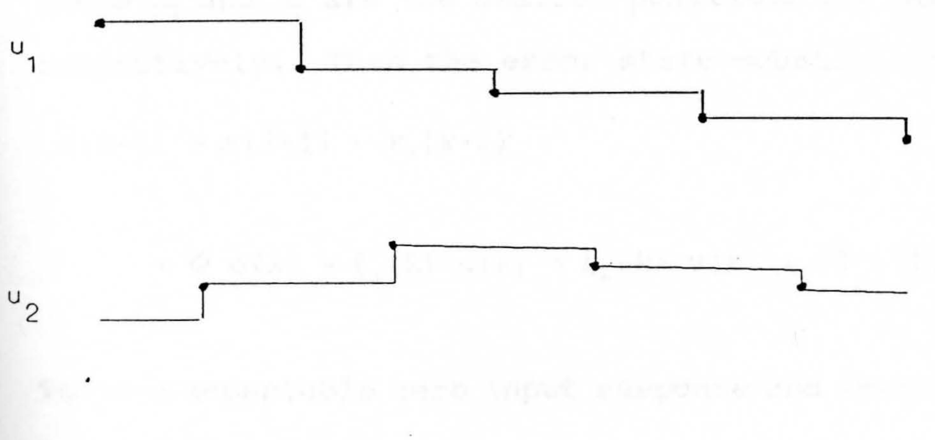

Figure 3.4: Sampling Mechanism for the Plant.

3.2.1 Controller Design

Once the SISO step-varying model of the plant is obtained, the next step is to design a step-varying controller for this system. In this section, the error step-varying controller design is carried out for the SISO step-varying system.

The tracking outputs are the positions of the two cars; therefore, the reference input model chosen for the plant is

$$x_r(k+1) = I_4 \, x_r(k)$$

$$with \quad x_r(0) = [\, R_1 \;\; 0 \;\; R_2 \;\; 0 \,]'$$

where R_1 and R_2 are the desired positions for car 1 and car 2, respectively. Then the error state equation is

$$e(k+1) = x(k+1) - x_r(k+1)$$

$$= \Phi \, e(k) + \xi_1(k) \, u(k) + \xi_2(k) \, v(k) + [\Phi - \Phi_r] \, x_r(k)$$

To have acceptable zero input response and zero-state response let the control law be

$$u(k) = -F(k) \, e(k) - L(k) \, v(k) - G(k) \, x_r(k)$$

where

F(k) is a 1x4 feedback control matrix,

G(k) is a 1x4 feedforward gain matrix, and

L(k) is a scalar gain on the previous input.

Then, in vector form, the error state equations are

$$\therefore \quad e(k+1) = A(k)\ e(k) + D(k)\ v(k) + B(k)\ x_r(k)$$

$$v(k+1) = -\ F(k)\ e(k) - L(k)\ v(k) - G(k)\ x_r(k)$$

$$x_r(k+1) = I_4\ x_r(k)$$

where

$$A(k) = \Phi - \xi_1(k)\ F(k)$$

$$D(k) = \xi_2(k) - \xi_1(k)\ L(k)$$

$$B(k) = \Phi - I_4 - \xi_1(k)\ G(k)$$

Since e(k) is of 4x1 dimension, choose j_1, j_2, j_3, j_4 as 4x1 basis vectors. To have the error after four steps to be zero in the j_1-direction,

$$j_1'\ e(4) = j_1'\ [\ A(3)\ e(3) + D(3)\ v(3) + b(3)\ x_r(3)] = 0$$

The above equation gives

$$F(3) = \frac{j_1'\ \Phi}{j_1'\ \xi_1(3)}$$

$$L(3) = \frac{j_1'\ \xi_2(3)}{j_1'\ \xi_1(3)}$$

$$G(3) = \frac{j_1'\ [\Phi - I_4]}{j_1'\ \xi_1(3)}$$

This can be used to calculate A(3), D(3) and B(3). To have zero error in the j_2-direction,

$$j_2' \, e(4) = j_2'[\, [A(3)\, A(2) - D(3)\, F(2)]\, e(2) + [A(3) D(2) - D(3)\, L(2)]\, v(2) + [A(3)\, B(2) - D(3)\, G(2) + B(3)\, I_4]\, x_r(2)\,] = 0$$

This gives,

$$F(2) = \frac{j_2' \, A(3)\, \Phi}{j_2' \, [A(3)\, \xi_1(2) + D(3)]}$$

$$L(2) = \frac{j_2' \, A(3)\, \xi_2(2)}{j_2' \, [A(3)\, \xi_1(2) + D(3)]}$$

$$G(2) = \frac{j_2' \, [A(3)\, \Phi - A(3)\, I_4 + B(3)\, I_4]}{j_2' \, [A(3)\, \xi_1(2) + D(3)]}$$

Similarly, by forcing the error to go to zero in the j_3 and j_4 directions, we get the gain matrices for steps 1 and 0 as

$$F(1) = \frac{j_3' \, [A(3)\, A(2)\, \Phi - D(3)\, F(2)\, \Phi\,]}{j_3'[A(3) A(2) \xi_1(1) - D(3) F(2) \xi_1(1) + A(3) D(2) - D(3) L(2)]}$$

$$L(1) = \frac{j_3' \, [A(3)\, A(2)\, \xi_2(1) - D(3)\, F(2)\, \xi_2(1)\,]}{j_3'[A(3) A(2) \xi_1(1) - D(3) F(2) \xi_1(1) + A(3) D(2) - D(3) L(2)]}$$

$$G(1) = \frac{j_3'[A(3) A(2)\, \Phi - A(3) A(2) I_4 - D(3) F(2)\, \Phi + D(3) F(2) I_4 + A(3)\, B(2)\, I_4 - D(3)\, G(2)\, I_4 + B(3)\, I_4^2]}{j_3'[A(3) A(2) \xi_1(1) - D(3) F(2) \xi_1(1) + A(3) D(2) - D(3) L(2)]}$$

$$F(0) = \frac{j_4 \, [A(3)\, A(2)\, A(1) - D(3)\, F(2)\, A(1) - A(3)\, D(2)\, F(1) + D(3)\, L(2)\, F(1)]\, \Phi}{J_4' \, DENOM}$$

$$L(0) = \frac{j_4' \, [A(3)\,A(2)\,A(1) - D(3)\,F(2)\,A(1) - A(3)\,D(2)\,F(1) + D(3)\,L(2)\,F(1)]\,\xi_2(0)}{J_4' \; DENOM}$$

$$G(0) = \frac{j_4' [\,[A(3)A(2)A(1) - D(3)F(2)A(1) - A(3)D(2)F(1) + D(3)L(2)F(1)]\,(\Phi - I_4) + [A(3)A(2)B(1) - D(3)F(2)B(1) - A(3)D(2)G(1) + D(3)L(2)G(1) + A(3)B(2)I_4 - D(3)G(2)I_4 + B(3)I_4^2]\,I_4]}{j_4' \; DENOM}$$

where

$$DENOM = j_4' [\,[A(3)A(2)A(1) - D(3)F(2)A(1) - A(3)D(2)F(1) + D(3)L(2)F(1)]\,\xi_1(0) + A(3)A(2)D(1) - D(3)F(2)D(1) - A(3)D(2)L(1) + D(3)L(2)L(1)]$$

The calculations for the control gains are backward in step.

To have continued feedback after the fourth step, applying the same technique for next four steps we get,

$$F(7) = \frac{j_1' \, \Phi}{j_1' \, \xi_1(7)}$$

but

$$\xi_1(7) = \xi_1(3)$$

$$\therefore \; F(7) = F(3)$$

similarly,

$$L(7) = L(3) \quad , \quad G(7) = G(3)$$

Also,

$$F(6) = \frac{j_2' \, A(7) \, \Phi}{j_2' \, [A(7) \, \xi_1(6) + D(7)]}$$

but

$$A(3) = A(7) \quad , \quad \xi_1(2) = \xi_1(6) \quad , \quad D(3) = D(7)$$

$$\therefore \quad F(6) = F(2)$$

We can also show that F(5)=F(1) and F(4)=F(0). Because ξ_1, ξ_2 are periodic of period two and the error state vector is of dimensions 4x1, the period for the gain matrices F(k), G(k), L(k) is four, which is the minimum multiple of SISO system period (2) and the order of the MIMO system (4). That is,

F(3)=F(7)=F(11)=.....

F(2)=F(6)=F(10)=.....

F(1)=F(5)=F(9)=.....

F(0)=F(4)=F(8)=.....

And the same observations hold for G(k) and L(k).

The four sets of basis vectors which are considered for finding the control gain matrices are given below.

1) Unit coordinate vectors:

$$j_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad j_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad j_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad j_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

2) Eigenvectors of the discrete MIMO model of the plant:

$$j_1 = \begin{bmatrix} 7.071068E\text{-}1 \\ -1.965913E\text{-}15 \\ 7.071068E\text{-}1 \\ 2.126375Ee\text{-}15 \end{bmatrix}, \quad j_2 = \begin{bmatrix} 3.557045E\text{-}1 \\ -6.111255E\text{-}1 \\ -3.557045E\text{-}1 \\ 6.111255E\text{-}1 \end{bmatrix},$$

$$j_3 = \begin{bmatrix} 3.034351E\text{-}2 \\ -7.064554E\text{-}1 \\ -3.034351E\text{-}2 \\ 7.064554E\text{-}1 \end{bmatrix}, \qquad j_4 = \begin{bmatrix} -2.826167E\text{-}2 \\ 7.065418E\text{-}1 \\ -2.826167E\text{-}2 \\ 7.065418E\text{-}1 \end{bmatrix}.$$

3) First four independent column vectors of controllability matrix of the discrete MIMO model of the plant. Hereafter, these vectors are referred to as the controllability vectors:

$$j_1 = \begin{bmatrix} 1.248633E\text{-}2 \\ 1.781246E\text{-}1 \\ 1.703513E\text{-}4 \\ 5.45840Ee\text{-}3 \end{bmatrix}, \qquad j_2 = \begin{bmatrix} 1.703513E\text{-}4 \\ 5.458400E\text{-}3 \\ 1.248633E\text{-}2 \\ 1.781246E\text{-}1 \end{bmatrix},$$

$$j_3 = \begin{bmatrix} 1.823123E\text{-}2 \\ 2.321339E\text{-}3 \\ 1.165995E\text{-}3 \\ 1.274807E\text{-}2 \end{bmatrix}, \qquad j_4 = \begin{bmatrix} 1.165995E\text{-}2 \\ 1.274807E\text{-}2 \\ 1.823123E\text{-}2 \\ 2.321339E\text{-}3 \end{bmatrix}.$$

4) First four independent row vectors of observability matrix of the discrete MIMO model of the plant. These vectors are called as observability vectors:

$$j_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad j_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix},$$

$$j_3 = \begin{bmatrix} 9.507361E\text{-}1 \\ 3.562492E\text{-}2 \\ 4.926391E\text{-}2 \\ 1.901680E\text{-}3 \end{bmatrix}, \qquad j_4 = \begin{bmatrix} 4.926391E\text{-}2 \\ 1.091680E\text{-}3 \\ 9.507361E\text{-}1 \\ 3.563492E\text{-}2 \end{bmatrix}.$$

To find the numerical values of gain matrices for each set of basis vectors, Program CC macros have been developed [APPENDIX B]. The control gain matrices for the above sets of basis vectors for the dead beat control are given below.

1. Control gain matrices when the basis vectors are unit coordinate vectors:

$F(3) = [5.582713E{+}3 \quad 2.091892E{+}2 \quad 2.892772E{+}2 \quad 6.410335E{+}0]$

$F(2) = [3.600130E{+}1 \quad 1.463846E{+}0 \quad 3.879445E{+}0 \quad 1.122515E{-}1]$

$F(1) = [6.262594E{+}2 \quad 2.573601E{+}1 \quad 9.006203E{+}1 \quad 2.837284E{+}0]$

$F(0) = [2.267406E{+}1 \quad 9.336732E{-}1 \quad 4.316701E{+}0 \quad 1.457069E{-}1]$

$L(3) = 73.28244 \qquad L(2) = 3.062769E{-}2$

$L(1) = 13.21242 \qquad L(0) = 5.508251E{-}2$

$G(3) = [-2.892772E{+}2 \quad 2.091892E{+}2 \quad 2.892772E{+}2 \quad 6.410335E{+}0]$

$G(2) = [-3.879445E{+}0 \quad 2.708201E{+}0 \quad 3.879445E{+}0 \quad 1.122515E{-}1]$

$G(1) = [-4.889294E{+}1 \quad 1.409373E{+}2 \quad 4.889294E{+}1 \quad 2.837284E{+}0]$

$G(0) = [-3.782079E{+}0 \quad 5.447524E{+}0 \quad 3.782079E{+}0 \quad 7.013680E{-}2]$

2. Control gain matrices when the basis vectors are eigen vectors:

$F(3) = [7.904951E{+}1 \quad 2.902429E{+}0 \quad 7.904951E{+}1 \quad 2.902429E{+}0]$

$F(2) = [2.009368E{+}1 \quad 8.056389E{-}1 \quad 2.627241E{+}1 \quad 1.036508E{+}0]$

$F(1) = [-6.82132E{+}0 \quad -3.13805E{-}1 \quad 2.256269E{+}1 \quad 9.4311E{-}1]$

$F(0) = [2.267406E{+}1 \quad 9.336732E{-}1 \quad 4.316701E{+}0 \quad 1.457069E{-}1]$

$L(3) = 1 \qquad L(2) = 0.4855799$

$L(1) = -0.1817109 \qquad L(0) = 5.508251E{-}2$

$G(3) = [3.34153E{-}13 \quad 2.90242E{+}0 \quad -3.14987E{-}13 \quad 2.902429E{+}0]$

$G(2) = [-2.057808E{+}0 \quad -9.66644E{-}1 \quad 2.057808E{+}0 \quad 2.80879E{+}0]$

$G(1) = [4.729583E{+}0 \quad -2.275324E{+}0 \quad -4.729583E{+}0 \quad 2.90463E{+}0]$

$G(0) = [-3.782079E{+}0 \quad 5.447524E{+}0 \quad 3.782079E{+}0 \quad 7.013680E{-}2]$

3. Control gain matrices when the basis vectors are controllability vectors:

$F(3) = [-5.51036E{+}1 \quad 5.784962E{+}0 \quad 6.159895E{+}1 \quad 2.18698E{+}0]$

$F(2) = [-3.33592E{+}0 \quad -9.911961E{-}2 \quad 3.703149E{+}0 \quad 1.49598E{-}1]$

$F(1) = [-5.74848E{+}3 \quad -2.36569E{+}2 \quad -5.89615E{+}2 \quad -1.62358E{+}1]$

$F(0) = [2.26740E{+}1 \quad 9.336732E{-}1 \quad 4.316701E{+}0 \quad 1.45706E{-}1]$

$L(3) = 16.3757 \qquad L(2) = 6.973661E{-}2$

$L(1) = -121.6584 \qquad L(0) = 5.508251E{-}2$

$G(3) = [-6.15115E{+}1 \quad -8.56267E{+}1 \quad 6.15115E{+}1 \quad -6.14206E{-}1]$

$G(2) = [-3.72144E{+}0 \quad -5.593035E{+}0 \quad 3.72144E{+}0 \quad 3.17050E{-}1]$

$G(1) = [4.91053E{+}2 \quad -1.301127E{+}3 \quad -4.910536E{+}2 \quad 3.51541E{+}0]$

$G(0) = [-3.78207E{+}0 \quad 5.447524E{+}0 \quad 3.782079E{+}0 \quad 7.01368E{-}2]$

4. Control gain matrices when the basis vectors are observability vectors:

$F(3) = [5.582713E{+}3 \quad 2.091892E{+}2 \quad 2.892772E{+}2 \quad 6.410335E{+}0]$

$F(2) = [2.870587E{+}1 \quad 1.174457E{-}0 \quad 3.437476E{+}0 \quad 1.026141E{-}1]$

$F(1) = [2.237071E{+}2 \quad 9.171907E{+}0 \quad 4.714172E{+}1 \quad 1.632851E{+}0]$

$F(0) = [2.267406E{+}1 \quad 9.336732E{-}1 \quad 4.316701E{+}0 \quad 1.457069E{-}1]$

$L(3) = 73.28244 \qquad L(2) = .0298802$

$L(1) = 4.695595 \qquad L(0) = 5.508251E{-}2$

$G(3) = [-2.892772E{+}2 \quad 2.091892E{+}2 \quad 2.892772E{+}2 \quad 6.410335E{-}0]$

$G(2) = [-3.882167E{+}0 \quad 1.174457E{+}0 \quad 3.882167E{+}0 \quad 1.026141E{-}1]$

$G(1) = [-1.479639E{+}1 \quad 4.987375E{+}1 \quad 1.479639E{+}1 \quad 2.880107E{+}0]$

$G(0) = [-3.782079E{+}0 \quad 5.447524E{+}0 \quad 3.782079E{+}0 \quad 7.013680E{-}2]$

An interesting point is that the gain matrices at step 0 are the same for all four sets.

3.2.2 Observer Design

In this section observer design for the SISO step-varying system is carried out. The state equation for the SISO discrete plant is,

$$x(k+1) = \Phi\, x(k) + \xi_1(k)\, u(k) + \xi_2(k)\, v(k)$$

$$v(k+1) = u(k)$$

Since v is the previous input there is no need to estimate it. Therefore the observer state equation has the form given below,

$$\hat{x}(k+1) = \Phi \hat{x}(k) + \xi_1(k)\, u(k) + \xi_2(k)\, v(k) + P(k)\, [y(k) - \hat{y}(k)]$$

where

$$\hat{y}(k) = C(k)\, \hat{x}(k)$$

$$\begin{aligned} \therefore\ e(k+1) &= x(k+1) - \hat{x}(k+1) \\ &= A(k)\, e(k) \end{aligned}$$

where

$$A(k) = \Phi - P(k)\, C(k)$$

Then applying the technique of section (2.4.3), we have

$$P(0) = \frac{\Phi\, j_1}{C(0)\, j_1}$$

$$P(1) = \frac{\Phi\, A(0)\, j_2}{C(1)\, A(0)\, j_2}$$

$$P(2) = \frac{\Phi\, A(1)\, A(0)\, j_3}{C(2)\, A(1)\, A(0)\, j_3}$$

$$P(3) = \frac{\Phi\ A(2)\ A(1)\ A(0)\ j_4}{C(3)\ A(2)\ A(1)\ A(0)\ j_4}$$

The calculations for the observer gain matrices are forward in step. To have continued observations, the same method is applied to calculate the next four observer gain matrices as follows:

$$P(4) = \frac{\Phi\ j_1}{C(4)\ j_1}$$

but

$$C(4) = C(0)$$

$$\therefore\ P(4) = P(0)$$

$$P(5) = \frac{\Phi\ A(4)\ j_2}{C(5)\ A(4)\ j_2}$$

but

$$C(5) = C(1) \quad , \quad A(4) = A(0)$$

$$\therefore\ P(5) = P(1)$$

Similarly we can show that P(6)=P(2), P(7)=P(3). Again since C(k) is periodic of period two and state vector x(k) is 4x1, the period for the observer gain matrices is four (minimum multiple of the SISO period (2) and order of the MIMO system (4)). That is,

P(0)=P(4)=P(8)=....

P(1)=P(5)=P(9)=....

P(2)=P(6)=P(10)=....

P(3)=P(7)=P(11)=....

Here the same four sets of basis vectors for the observer gain calculations are used as were used for the control gain calculations. To calculate the numerical values for the observer gain matrices for the different sets of basis vectors Program CC macros have been developed [APPENDIX C]. The observer gain matrices for the above sets of basis vectors for a deadbeat observer design are as follows:

1. The observer gain matrices when the basis vectors are unit co-ordinate vectors:

$$L(0) = \begin{bmatrix} 9.507361E-1 \\ -6.906648E-1 \\ 4.926391E-2 \\ 6.906648E-1 \end{bmatrix} \qquad L(1) = \begin{bmatrix} 3.305839E+1 \\ -1.809555E+1 \\ 3.335496E+0 \\ 2.426764E+1 \end{bmatrix}$$

$$L(2) = \begin{bmatrix} 9.348520E-1 \\ -6.578112E-1 \\ 1.451216E-1 \\ 6.731942E-1 \end{bmatrix} \qquad L(3) = \begin{bmatrix} 1.424716E-1 \\ 6.747090E-1 \\ 9.346653E-1 \\ -6.598071E-1 \end{bmatrix}$$

2. The observer gain matrices when the basis vectors are eigen vectors:

$$L(0) = \begin{bmatrix} 1 \\ 1.102756E-17 \\ 1 \\ 1.248038E-16 \end{bmatrix} \qquad L(1) = \begin{bmatrix} 1.578584E-1 \\ 6.614367E-1 \\ 9.278345E-1 \\ -6.614367E-1 \end{bmatrix}$$

$$L(2) = \begin{bmatrix} 9.288806E-1 \\ -7.216324E-1 \\ 6.038664E-2 \\ 7.216324E-1 \end{bmatrix} \qquad L(3) = \begin{bmatrix} 1.424716E-1 \\ 6.747090E-1 \\ 9.346653E-1 \\ -6.598071E-1 \end{bmatrix}$$

3. The observer gain matrices when the basis vectors are

observability vectors:

$$L(0) = \begin{bmatrix} 9.507361E\text{-}1 \\ -6.906648E\text{-}1 \\ 4.926391E\text{-}2 \\ 6.906648E\text{-}1 \end{bmatrix} \qquad L(1) = \begin{bmatrix} 1.236146E\text{-}1 \\ 6.825847E\text{-}1 \\ 9.282020E\text{-}1 \\ -6.825847E\text{-}1 \end{bmatrix}$$

$$L(2) = \begin{bmatrix} 9.350390E\text{-}1 \\ -6.558127E\text{-}1 \\ 1.477750E\text{-}1 \\ 6.716774E\text{-}1 \end{bmatrix} \qquad L(3) = \begin{bmatrix} 1.424716E\text{-}1 \\ 6.747090E\text{-}1 \\ 9.346653E\text{-}1 \\ -6.598071E\text{-}1 \end{bmatrix}$$

4. The observer gain matrices when the basis vectors are controllability vectors:

$$L(0) = \begin{bmatrix} 1.460095E\text{-}0 \\ 1.859104E\text{-}1 \\ 9.338170E\text{-}2 \\ 1.020962E\text{-}0 \end{bmatrix} \qquad L(1) = \begin{bmatrix} 1.221390E\text{-}1 \\ 7.004413E\text{-}1 \\ 9.581788E\text{-}1 \\ -6.334594E\text{-}1 \end{bmatrix}$$

$$L(2) = \begin{bmatrix} 9.349149E\text{-}1 \\ -6.571389E\text{-}1 \\ 1.460141E\text{-}1 \\ 6.726840E\text{-}1 \end{bmatrix} \qquad L(3) = \begin{bmatrix} 1.424716E\text{-}1 \\ 6.747090E\text{-}1 \\ 9.346653E\text{-}1 \\ -6.598071E\text{-}1 \end{bmatrix}$$

The interesting point is the observer gain matrix at the fourth step is the same for all the four sets.

3.2.3 Simulation of the Design

In the design of the step-varying controller, the control and observer gain matrices depend on the set of basis vectors used to calculate those matrices. We have considered 4 sets of basis vectors for comparison purposes. To simulate the response of the controlled system for different sets of control and observer gain matrices, a simulation program is written in C language [APPENDIX D].

The following equations are used in the simulation program to find control inputs and tracking outputs at each step.

$$u(k) = -F(k)\,\hat{x}(k) - L(k)\,v(k) - G(k)\,x_r(k)$$

$$y(k) = C(k)\,x(k) \qquad \hat{y}(k) = C(k)\,x(k)$$

$$x(k+1) = \Phi(k)\,x(k) + \xi_1(k)\,u(k) + \xi_2(k)\,v(k)$$

$$\hat{x}(k+1) = \Phi(k)\,\hat{x}(k) + \xi_1(k)\,u(k) + \xi_2(k)\,v(k) + P(k)\,[y(k) - \hat{y}(k)]$$

Since there are four sets of the basis vectors for calculations of the control as well as observer gain matrices, there are sixteen possible combinations of gains to compare.

The results for output and control input responses from the simulation program are plotted and compared. The simulation responses for all the above combinations are carried out and compared. Using a simulation program, we get values of outputs and control inputs only at the sampling steps (time steps). In simulation plots, straight lines are drawn between the sampling values, in order to read the values easily. Also, in the following discussion you will encounter terms like "eigen controller", "coordinate observer", etc. The term "eigen controller" means the controller for which the eigenvectors are used as the basis vectors to calculate the control gain matrices. Also the term "coordinate observer" means the observer for which coordinate vectors are used as the basis vectors to calculate the observer gain matrices.

From these responses it is clear that eigen controller is the best for the given system. Also, the coordinate observer is not desirable for the given system. A few of these results are shown in figures 3.5.1 to 3.8.2, which illustrate the preference for the eigencontroller and eigen observer.

Figures 3.5.1 and 3.5.2 show the zero state response of the controlled system without observer error for the four sets of control gain matrices obtained using the four sets of basis vectors. The desired reference state vector x_r is [.1 0 .2 0]'. Both the output and control input responses are shown.

When the observer error is zero, the observer state vector is exactly equal to the plant state vector at each step, then observer gains do not contribute in shaping the system response. Therefore there are only four responses to compare.

From the output responses (figure 3.5.1), it is clear that each set of control gain matrices gives a very good output response. Each response goes to the desired state in an acceptable manner and in four steps as we have designed (i.e., a deadbeat design). Also, all four output responses are quite similar. One reason might be that the control gain matrices are the same for each set of the basis vectors at the 0^{th} step. The plots of control inputs (figure 3.5.2) for all four sets of control gain matrices are also similar due to the same reasons. Also each control input for all the four sets is reasonable. This response does not help us in selecting

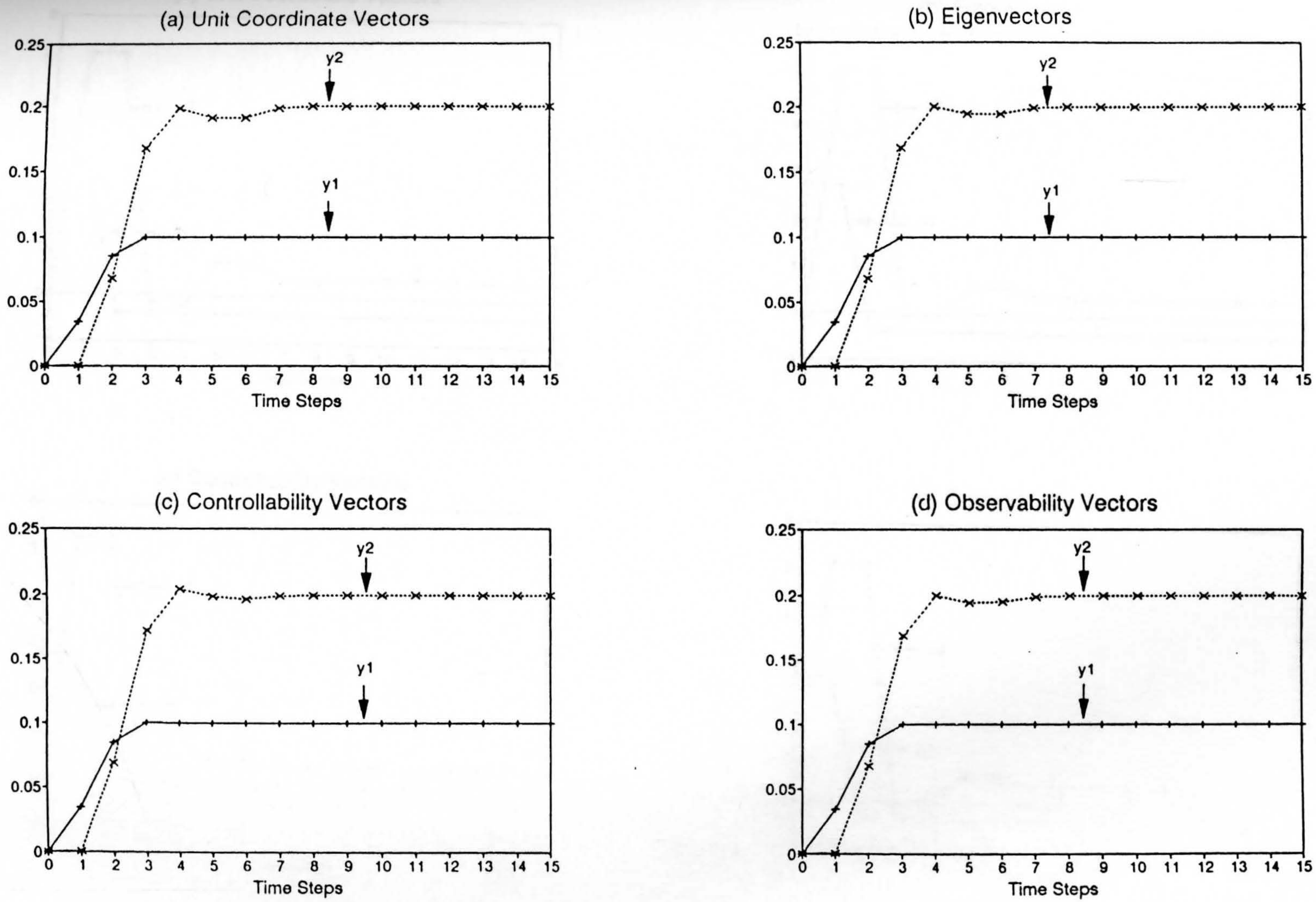

Figure 3.5.1: Tracking Output Responses to x_r=[0 0.1 0 0.2]' for the four sets of Control Gains.

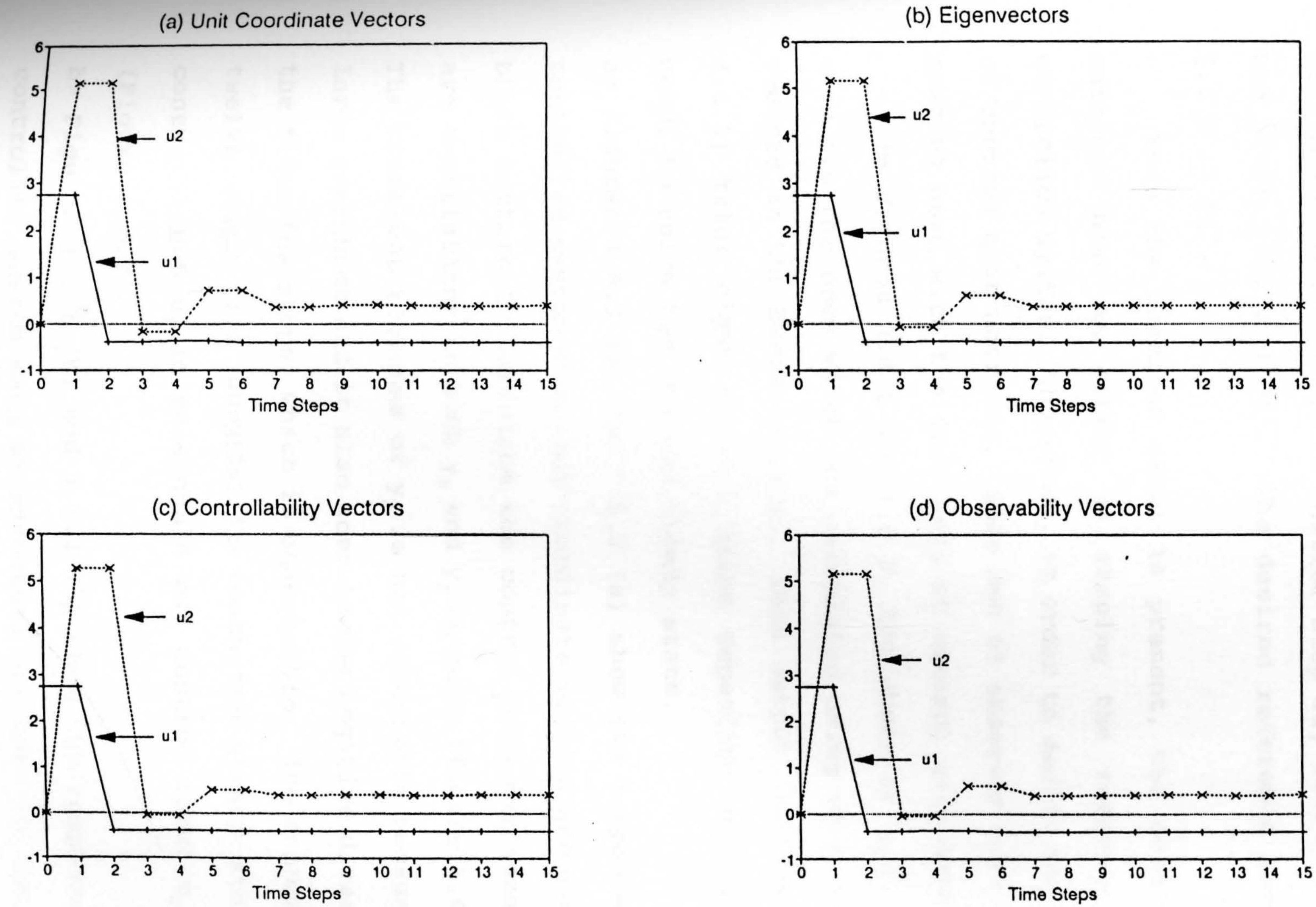

Figure 3.5.2: Control Signal Responses to x_r=[0 0.1 0 0.2]' for the four sets of Control Gains.

the best controller.

Figures 3.6.1 and 3.6.2 show the responses of the system with the observer error in initial step as, x(0)=[.03 0 0 0]' and $\hat{x}(0)$= [.01 0 0 0]'. The desired reference input x_r is [.1 0 .2 0]'.

When the observer error is present, the observer gain matrices have an effect in shaping the response of the controlled system. Therefore, in order to decide the best set of control gain matrices, same set of observer gain matrices must be used with the four sets of control gain matrices.

In figures 3.6.1 and 3.6.2, the observer gain matrices used are the ones which are calculated using unit coordinate vectors as the basis vectors. Each output response (figure 3.6.1) takes eight or more steps depending on the control used, to reach the desired steady state.

a> Figures 3.6.1 (a) and 3.6.2 (a) show the responses of the controlled system when unit coordinate vectors are used as the basis vectors to calculate the control gain matrices. There are oscillations in both y_1 and y_2 outputs (figure 3.6.1 (a)). The transient response of y_2 is not acceptable because it has large overshoot and it also goes in the negative direction for the first few steps, which is undesirable. The response takes twelve steps (1.2 seconds) to reach the steady state. The control input u_1 is reasonable but control input u_2 is high (Figure 3.6.2).

b> Figures 3.6.1 (b) and 3.6.2 (b) show the responses of the controlled system when eigenvectors are used as the basis

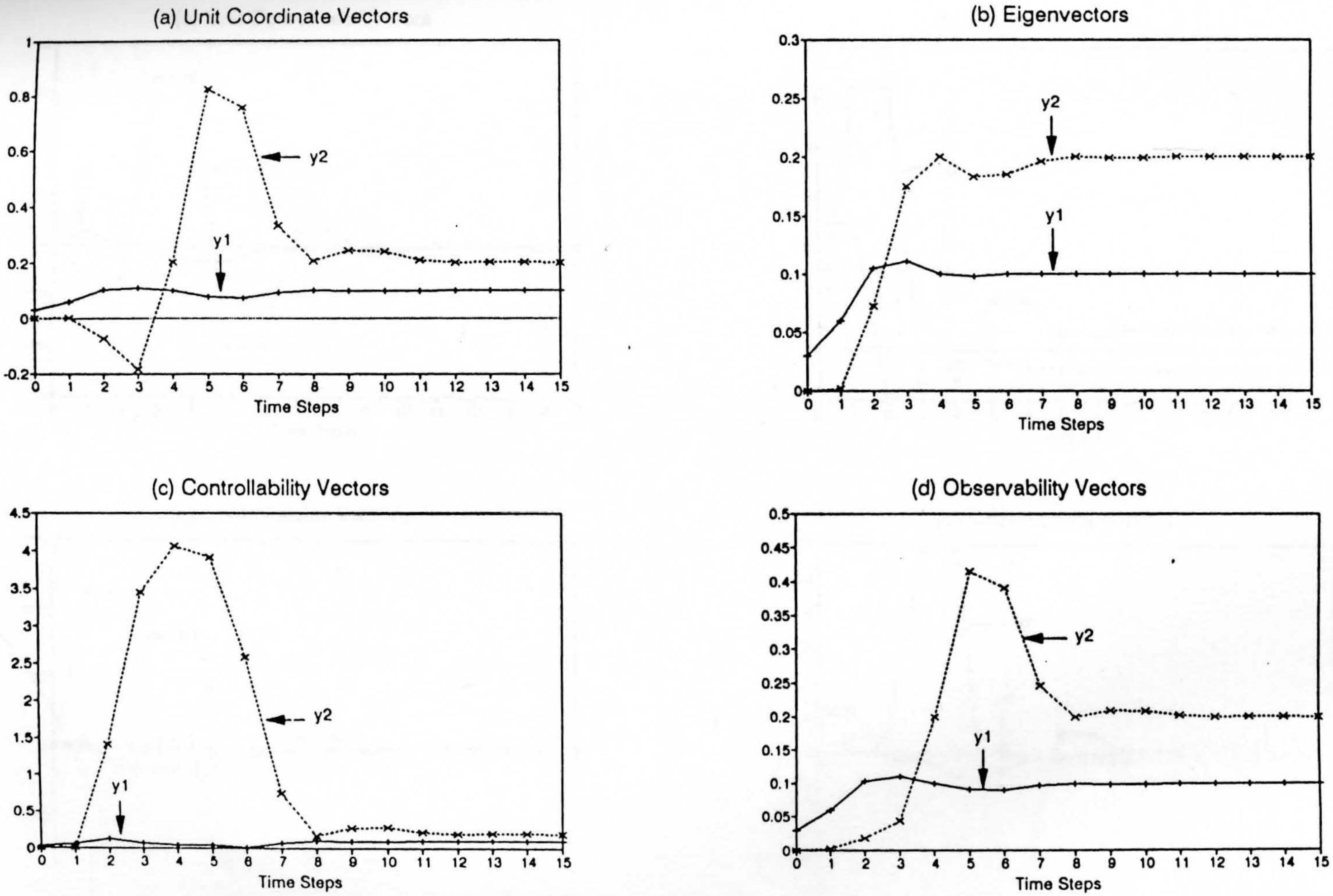

Figure 3.6.1: Tracking Output Responses to x_r=[0 0.1 0 0.2]', x(0)=[0.03 0 0 0]' and $\hat{x}(0)$=[0.01 0 0 0]' for the four sets of Control Gains with "Coordinate Observer".

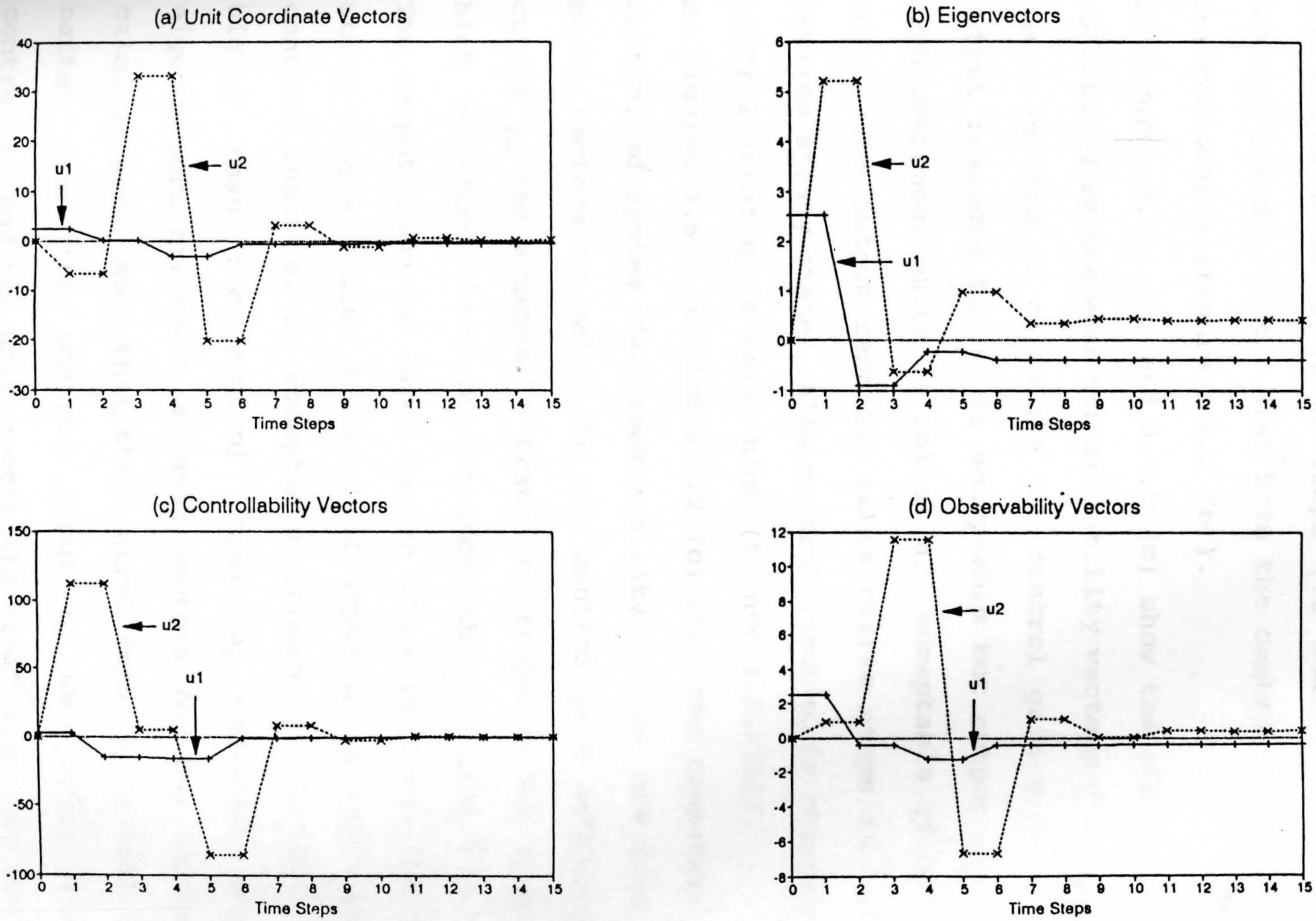

Figure 3.6.2: Control Signal Responses to x_r=[0.1 0 0.2 0]', $x(0)$=[0.03 0 0 0]' and $\hat{x}(0)$=[0.01 0 0 0]' for the four sets of Control Gains with "Coordinate Observer".

vectors to calculate the control gain matrices. Both the outputs (figure 3.6.1 (b)) have acceptable transient response. The response takes eight steps (.8 seconds) to reach the desired steady state. Also both the control inputs u_1 and u_2 are reasonable (figure 3.6.2 (b)).

c> Figures 3.6.1 (c) and 3.6.2 (c) show the responses of the controlled system when controllability vectors are used as the basis vectors to calculate the control gain matrices. The output response for y_1 is acceptable but output y_2 has very high overshoot which is not at all acceptable (figure 3.6.1 (c)). The output response takes twelve steps to reach the desired steady state. Also control input u_1 is reasonable but control input u_2 is very high (figure 3.6.2(c)).

d> Figures 3.6.1 (d) and 3.6.2 (d) show the responses of the controlled system when observability vectors are used as the basis vectors to calculate the control gain matrices. The output y_1 has acceptable transient response but y_2 has very high overshoot which is not acceptable (figure 3.6.1 (d)). The output response takes twelve steps to reach the desired steady state. Also the control input u_1 is reasonable and control input u_2 is acceptable (figure 3.6.2 (d)) but is higher than the control input u_2 required when the eigenvectors are used as basis vectors for the control gain calculations. Note that the control input u_2 in this case is better than the control input u_2 when controllability controller and the unit coordinate controller are used.

Comparing Figures 3.6.1 and 3.6.2 (a), (b), (c), (d), it is clear that for the two car train system when eigen vectors are used as the basis vectors for calculating control gain matrices, the desired response is obtained. Therefore eigencontroller is selected out of the four controllers for trying the design in real time. The next step is to choose a better observer design.

Figures 3.7.1 and 3.7.2 show the responses of the controlled system with observer error in initial state as, $x(0)=$ [.03 0 0 0]' and $\hat{x}(0)=$[.01 0 0 0]'. The desired reference input x_r is [.1 0.2 0]'. The controller is the eigen controller while four sets of observer gain matrices are considered.

a> Figures 3.7.1 (a) and 3.7.2 (a) show the responses of the controlled system when unit coordinate vectors are used as the basis vectors to calculate the observer gain matrices.

b> Figures 3.7.1 (b) and 3.7.2 (b) show the responses of the controlled system when eigenvectors are used as the basis vectors to calculate the observer gain matrices.

c> Figures 3.7.1 (c) and 3.7.2 (c) show the responses of the controlled system when controllability vectors are used as the basis vectors to calculate the observer gain matrices.

d> Figures 3.7.1 (d) and 3.7.2 (d) show the responses of the controlled system when observability vectors are used as the basis vectors to calculate the observer gain matrices.

Since all the output responses (figure 3.7.1) go to the desired steady state and have preferable transients, it is difficult to choose the best response. To choose a better

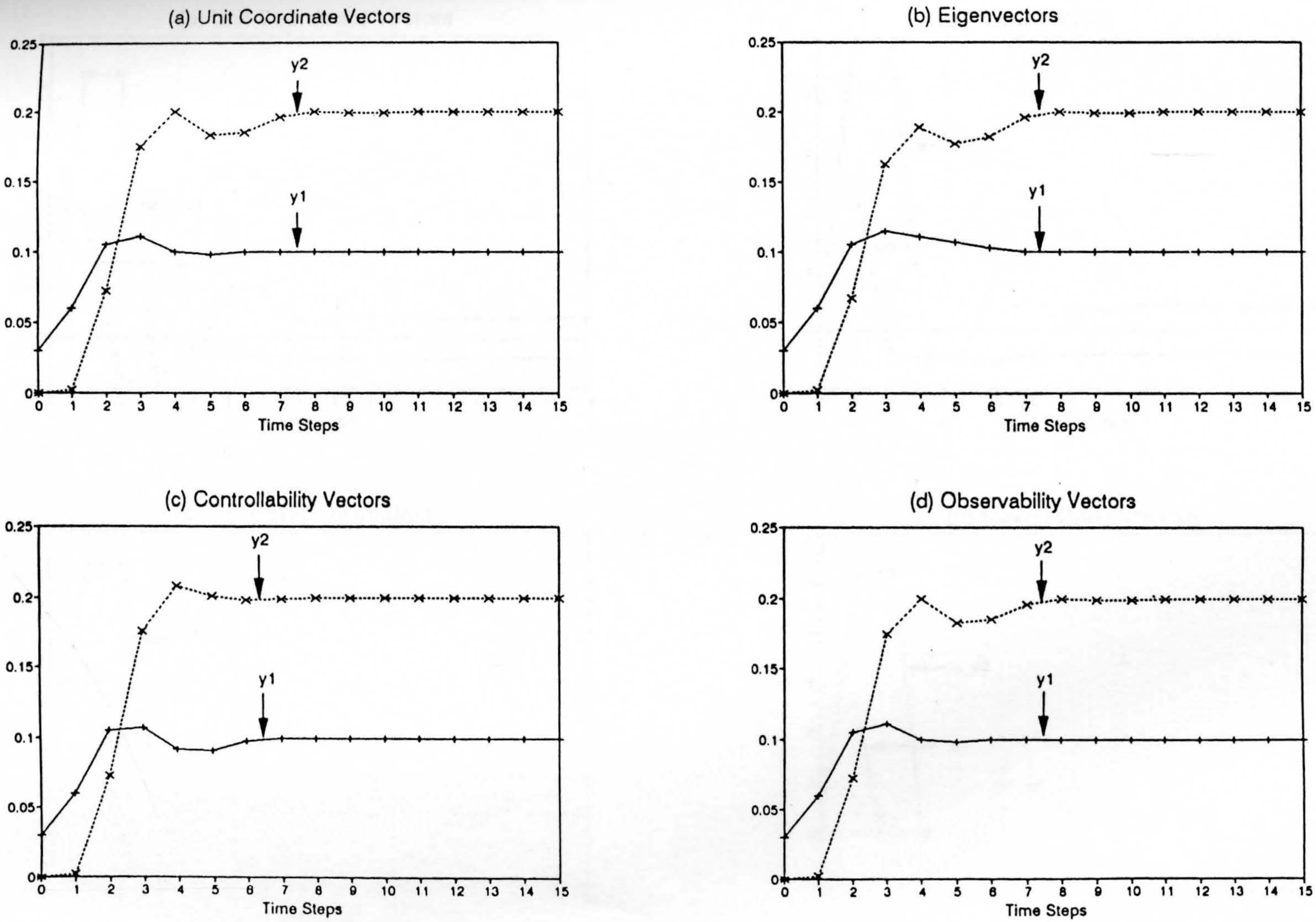

Figure 3.7.1: Tracking Output Responses to x_r=[0.1 0 0.2 0]', x(0)=[0.03 0 0 0]' and $\hat{x}$(0)=[0.01 0 0 0]' for the four sets of Observer Gains with "Eigen-Controller".

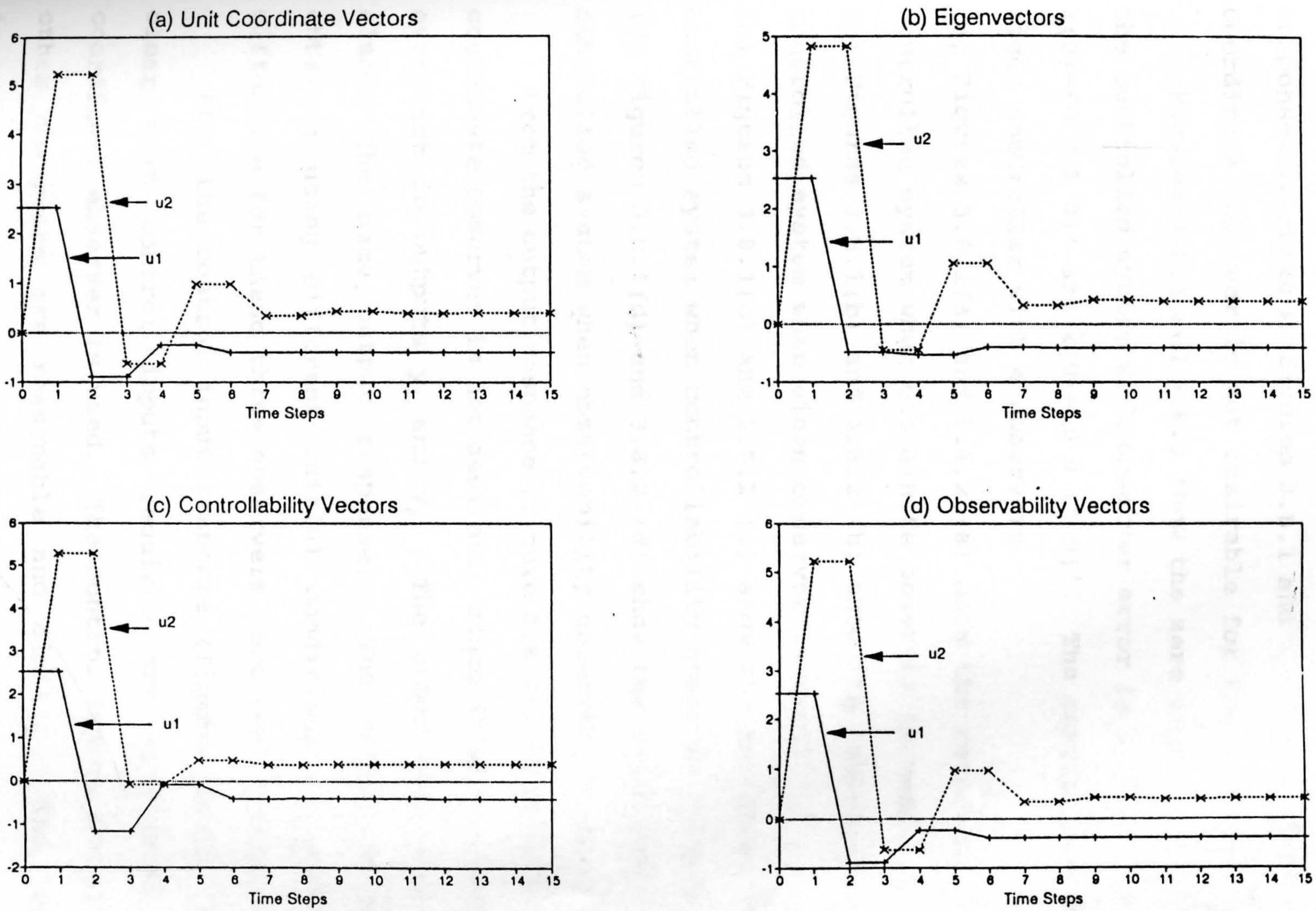

Figure 3.7.2: Control Signal Responses to x_r=[0.1 0 0.2 0]', x(0)=[0.03 0 0 0]' and $\hat{x}(0)$=[0.01 0 0 0]' for the four sets of Observer Gains with "Eigen-Controller".

observer, zero input responses with different initial conditions with observer error are carried out. One of these responses is shown in figures 3.8.1 and 3.8.2, which show that coordinate observer is not desirable for the given system.

Figures 3.8.1 and 3.8.2 show the zero input responses of the controlled system with observer error in initial state as x(0)=[0 0 1 0]' and $\hat{x}(0)$=[0 0 .5 0]'. The controller used is eigen controller with 4 observers.

(a) Figures 3.8.1(a) and 3.8.2 (a) show the responses of the controlled system when coordinate observer is used.

(b) Figures 3.8.1(b) and 3.8.2 (b) show the responses of the controlled system when eigen observer is used.

(c) Figures 3.8.1(c) and 3.8.2 (c) show the responses of the controlled system when controllability observer is used.

(d) Figures 3.8.1(d) and 3.8.2 (d) show the responses of the controlled system when observability observer is used.

From the output response (figure 3.8.1), it is clear that coordinate observer is not desirable since it gives very high overshoot in outputs y_1 and y_2. The other observers give almost the same output response. The output responses obtained using different initial conditions do show some difference for these three observers, but very little.

From the control input response (figure 3.8.2), it is clear that control inputs required are very high when coordinate observer is used. The control inputs for all the other observers are reasonable and similar. The control inputs required for different initial conditions are also

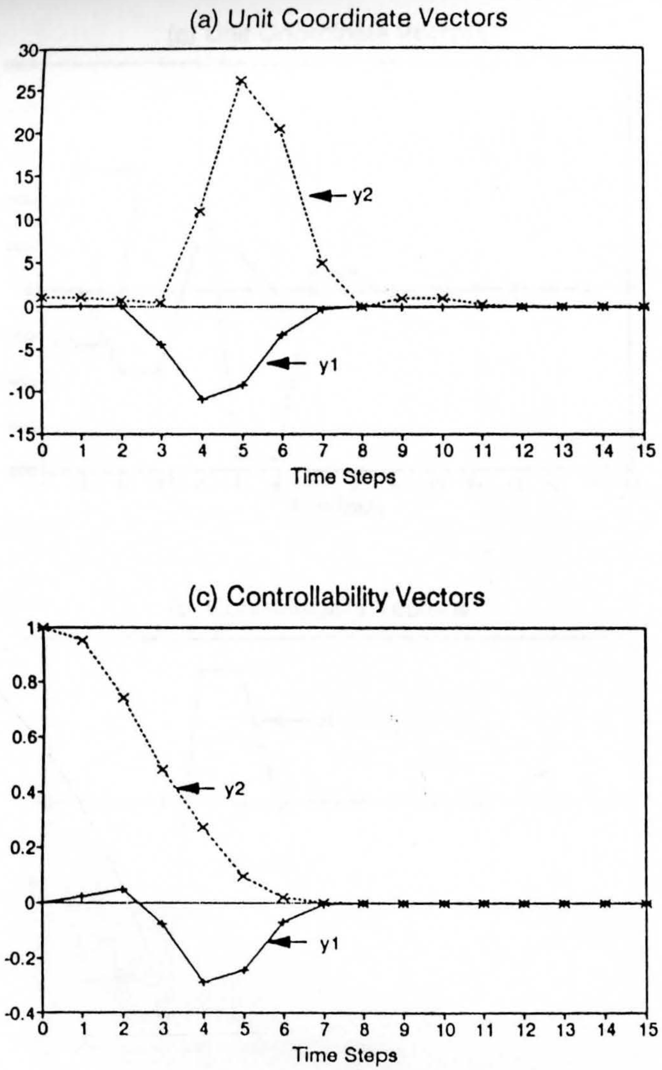

Figure 3.8.1: Tracking Output Responses to $x(0)=[0\ 0\ 1\ 0]'$ and $\hat{x}(0)=[0\ 0\ 0.5\ 0]'$ for the four sets of Observer Gains with "Eigen-Controller".

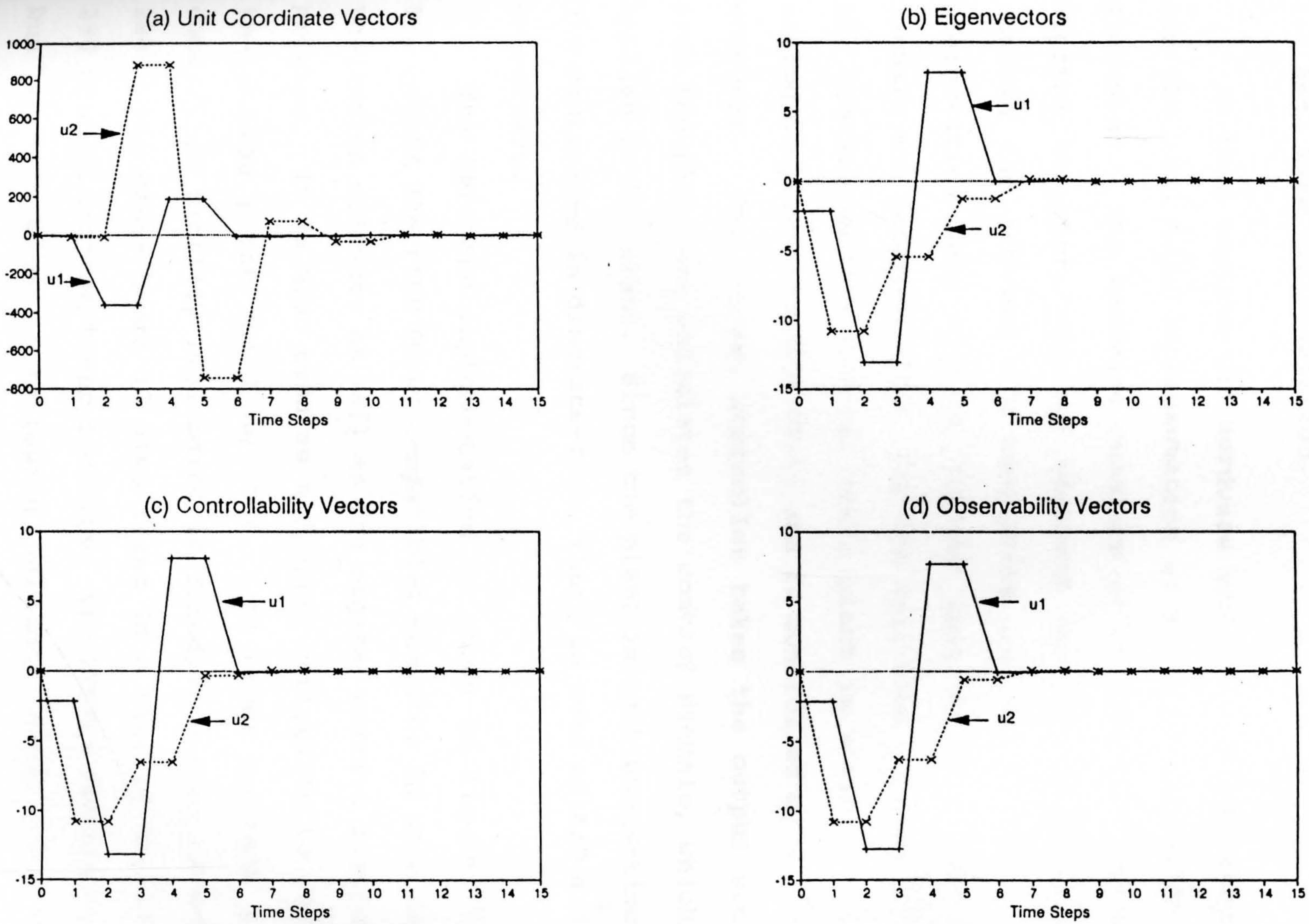

Figure 3.8.2: Control Signal Responses to x(0)=[0 0 1 0]' and $\hat{x}(0)$=[0 0 0.5 0]' for the four sets of Observer Gains with "Eigen-Controller".

comparable for these three observers.

3.3 Real Time Implementation.

In this section the hardware and the software required for the real time implementation of the control design are discussed. The hardware consists of an analog computer, a digital computer, and the required interfacing units that contain the A/D and D/A converters and the timers. The presentation here closely follows that in the respective manuals and reference [5]. For the real-time implementation, the continuous-time two-car train plant is simulated on an analog computer (Comdyna GP-6) and is controlled by a digital computer. The digital controller takes the output signals from the plant and calculates the control signals, which are applied to the plant. Since the plant is continuous-time and the controller is discrete-time, there is need of A/D and D/A converters.

The hardware implementation is shown in figure (3.9). The EVEREX 286 (IBM PC/AT compatible) computer is used to run the design software as well as the digital control simulation program. It is also used as a digital controller to control the analog plant build on GP-6. In order to have AD/DA conversion facility 7905 board is fixed in a slot of EVEREX 286. The DAS-8 board is also fixed in a slot of the EVEREX 286 to have counter/timer facility. The EVEREX 286 has system bus of 62 bits. Following sections gives the brief explanations for the various parts of the hardware.

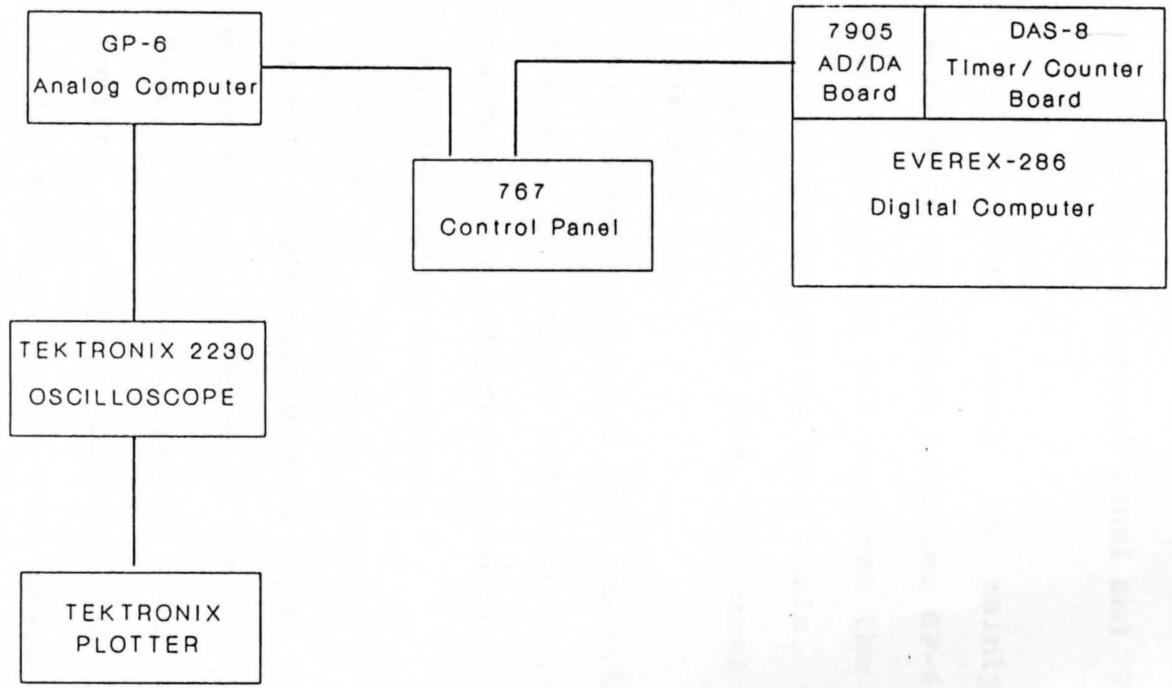

Figure 3.9: Block diagram of Hardware Implementation.

3.3.1 Interface between GP-6 analog computer and EVEREX 286 digital computer

The interfacing between the GP-6 analog computer and the EVEREX 286 digital computer is achieved using model 767 analog/digital position control panel and 7905 AD/DA interface board.

The 767 control panel is mainly used for the interconnection of an analog computer GP-6 and a 7905 AD/DA Board. The operation of 767 requires the connection of two cables, the 7905 cable and the GP-6 cable. The details of the interconnection between the GP-6, 767 panel and 7905 cable are given in reference [6].

The 7905 board is inserted into the EVEREX 286 computer's peripheral slot. Its inputs and outputs are terminated in a 25 pin Data Connector, which is connected to the 767 panel. The board has following functions.

1. A/D conversion ...... 12-bit successive approximation converter with eight multiplexed input channels.
2. D/A conversion ...... Three 12 bits DAC's
3. Logic Sense ...... Three (input) logic sense lines.
4. Logic Control ......Four (output) logic control lines.

The analog multiplexer on 7905 board selects the input signal to the A/D converter out of eight signals which are outputs of first four operational amplifiers of the GP-6 and A_4, A_5, A_6, A_7 connectors on 767 control panel.

The A/D and D/A conversion requires two data bytes since the computer has 8 bits data lines. The communications between 7905 board and the digital computer are conducted via two bilateral tranceiver TH and TL. The 8 bits data bus of the microprocessor is time shared by the tranceivers. The TH transmits high order 8 bits and the TL low order 8 bits. The high byte, data bits D_0-D_7 are A/D or D/A data bits B_4-B_{11}. Out of low byte, data bits D_4-D_7 are A/D or D/A data bits B_0-B_3. The address location and whether the instruction is either input (IN) or output (OUT) determines the function to be executed. During an input instruction, data is transmitted from either TH or TL into the AL accumulator. Then the low order data bits of the low byte are the three logic sense bits C_0-C_2 and the end-of-conversion signal from the A/D if address line A_1 is 1. During an output instruction, data is transmitted from the AL accumulator to either TH or TL. Then the data bits of the low byte are the four logic control bits.

Address words begins at 310hex (784 decimal) where A_4 through A_9 and AEN are fixed as the board code and address bits A_0 through A_3 and the instruction IN or OUT determines the function to be executed as shown in TABLE 3.1.

To control the GP-6 operation modes from the 7905 board, the GP-6 control push button must be in the OP (operation) position. Then in mode control (that is during OUT instruction), control bit C_3 is used to pull the GP-6 bus from operation state to an initial condition state. In mode sense (that is during IN instruction), control bit C_2 is used to check the status of the GP-6 operation bus. This is shown in

TABLE (3.2). More details of 7905 AD/DA board can be found in the reference manual [6].

3.3.2 The Metrobyte DAS-8 Interface Board.

This board is used to provide internal timing, which is required to run the control algorithm in real time. The discussion presented here closely follow that in reference [7].

The board has a highly advanced Intel 8254 timer/counter providing 3, 16-bit count-down registers. The DAS-8 derives its clock cycle from the IBM PC system clock. The connections are made via a standard 37 pin D male connector projecting through the rear of the computer. Each timer/counter has a clock input, a gate input that controls the counting and triggering, and an output. Counter two clock input is internally connected to the computer bus clock. The DAS-8 can be programmed using input and output instructions, but use of these functions usually requires formatting data and dealing with absolute I/O addresses. To simplify program development, a special I/O driver routine "DAS-8.BIN" is included in the DAS-8 software package. This can be accessed from the interpreter BASIC by a single CALL statement, where DAS8 is a variable that specifies the memory offset of the starting address of the CALL routine. Also a "DAS8.OBJ" driver routine is available to link with compiler BASIC, in which case DAS8 becomes the public name of the subroutine that can be called

TABLE 3.1 [5]

List of available functions of 7905 board

A_3	A_2	A_1	A_0	Address Locations Hex (Dec)	I/O	Description
0	0	0	0	310 (784)	OUT	Set Mux only.
0	0	0	1	311 (785)	OUT	Set Control byte only.
0	0	1	0	312 (786)	OUT	Set Mux and starts ADC.
0	0	1	1	313 (787)	OUT	Also starts ADC.
0	1	0	0	314 (788)	OUT	Set low byte of LDAC.
0	1	0	1	315 (789)	OUT	Set LDAC only.
0	1	1	0	316 (790)	OUT	Set low byte and starts LADC.
0	1	1	1	317 (791)	OUT	Set LDAC and starts ADC.
1	0	0	0	318 (792)	OUT	Set low byte of RDAC.
1	0	0	1	319 (793)	OUT	Set RDAC only.
1	0	1	0	31A (794)	OUT	Set low byte and starts ADC.
1	0	1	1	31B (795)	OUT	Set RDAC and starts ADC.
1	1	0	0	31C (796)	OUT	Set low byte of VDAC.
1	1	0	1	31D (797)	OUT	Set VDAC only.
1	1	1	0	31E (798)	OUT	Set low byte and starts ADC
1	1	1	1	31F (799)	OUT	Set VDAC and starts ADC.
0	0	1	0	312 (786)	IN	Read ADC data bits 0-3 as D_4-D_7 and logic sense as D_0-D_3.
0	0	1	1	313 (787)	IN	Read high order ADC data bits 4-11.

TABLE (3.2)

Logic condition of GP-6

GP-6 Control Push Button	OP Bus State	Logic Control C_3	Logic Sense C_2
OP	IC	high	low
OP	OP	low	high

in compiled BASIC program. These routines also save a lot of programming time. The format of the CALL statements is as follows:

CALL DAS8 (MD%,IP%,FLAG%)

where MD% is a data to select particular mode, IP% is a data which may be count value, configuration number, etc., and FLAG% is a error flag value.

The various operating modes of the CALL routine select the functions of DAS8. For our application we need four operation modes, as follow.

1. Mode 0 (Initialize DAS8 board) has to be set before other modes are selected. The initialization is required only once and is done by setting base address 300Hex for DAS-8 board.

2. Mode 10 (configure Timer/Counter) is used to configure Timer/counter. For our application counter 2 and counter 1 are set in configuration 3 while counter 0 is set in configuration 0. These two configurations are briefly discussed below.

Configuration 0 (Pulse high on terminal count):

When this configuration is set, output of a counter goes low. After count is loaded, the output remains low until counter decrements through zero; then it goes high and remains high until the counter is reloaded.

Configuration 3 (Square wave generator):

After loading the counter output goes high half the count and low for the other half. If the count N is even, symmetrical square wave output is obtained.

3. <u>Mode 11</u> (Load timer/counter) is used to load the selected timer/counter with a count.

4. <u>Mode 13</u> (Read digital inputs IP1-3) is used to read the state of digital inputs which is necessary to check whether the sampling period is over or not. In this thesis, the output of counter 1 is connected to IP2, the output of counter 0 is connected to IP1, and IP3 is grounded. Thus, the 3-bit word IP = IP3 IP2 IP1 can take on the values 0, 1, 2, or 3.

Concerning the sampling period setting and checking, counter 2 configuration 3 is set to generate a square wave. Counter 2 is loaded with count 396 to obtain a frequency of approximately 10KHz, which is used as clock input to the counter 1 and counter 0. The counter 1 configuration 3 is set and is loaded with count equal to 10 times the sampling period in ms. The counter 0 configuration 0 is set and is loaded with count equal to 5 times the sampling period. This arrangement is shown in figure (3.10).

From the figure it is clear that, if IP is 3 immediately after completion of control algorithm, the sampling period is too short for the calculations. But, if IP is 1 and then 3,

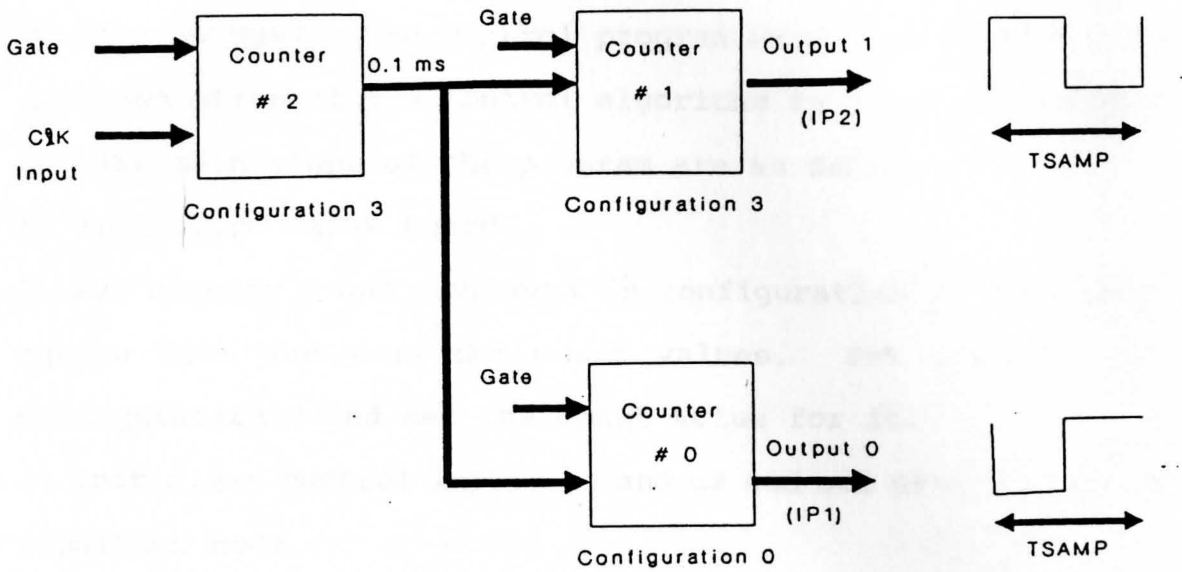

Fig 3.10: Connection Diagram between counters

then the sampling period is long enough for the control algorithm.

3.4 Real-Time Software

This section presents a brief explanation of the control program and also gives the real time plots of the closed-loop system. A few simulation results are also presented for comparison to the real-time results.

There is need of writing a control program, in order to control the analog plant using the digital computer. This program also carries out communication between the digital computer and interfacing boards. The communication between the 7905 board and the digital computer is carried out using

IN and OUT instructions, while for communication between the DAS-8 board and the digital computer, DAS-8 machine language routine is used. The control program used for the real-time implementation of the control algorithm is given in APPENDIX E. The main steps of the program are as follows:

1. Initialize DAS-8 board.

2. Set counter 2 and counter 1 in configuration 3 to generate square wave and load the count values. Set counter 0 in configuration 0 and set the count value for it.

3. Initialize control inputs u1 and u2 and set GP-6 in initial condition mode.

4. Load the data about control gain matrices, observer gain matrices, initial observer state vector, coefficient matrices in observer state equation and output equation, reference state vector and calculate initial control input u1.

5. Set GP-6 in operation mode.

6. a) Load the counter 0 with count value.

   b) Find the step is even or odd and accordingly

      i) send control input u1 to RDAC or u2 to LDAC.

      ii) measure output y1 or y2.

      iii) calculate the next observer state and control input.

      iv) update the values of observer state and previous input.

   c) Check the sampling period.

7. If "F1" key is pressed, set GP-6 in initial condition mode and stop the program run.

This program also has a few subroutines to perform certain tasks. These subroutines are briefly explained below.

1. Scale data subroutine .....The above program carries out calculations in decimal numbering system. But the program gets digital output from A/D converter which is in bipolar (2'complement) binary which needs to be converted into its equivalent decimal value in order to use it for calculations. This subroutine converts the binary output into its equivalent decimal value on 10v scale. Then that output value is utilize to calculate next control input.

2. Prepare data subroutine........The control input calculated in the program is in decimal system. To send it to the D/A converter, it needs to be converted into its equivalent binary form. This subroutine converts that decimal value of control input into 12 bit digital data word and sends it to D/A converter. The output of D/A converter is the control input to the analog plant.

Using the interfacing program, real time results for the controlled system are obtained. Some of these results along with simulated results are given below. The controller used to do real time results is the eigen- controller with the eigen-observer,(see section 3.2.3).

Figure 3.11.1 shows simulated output response and figure 3.11.2 shows real time output response of the controlled system. The responses are zero-input responses with plant state and observer state as $x(0)=[0\ 0\ .5\ 0]'$ and $\hat{x}(0)=[0\ 0\ .3\ 0]'$.

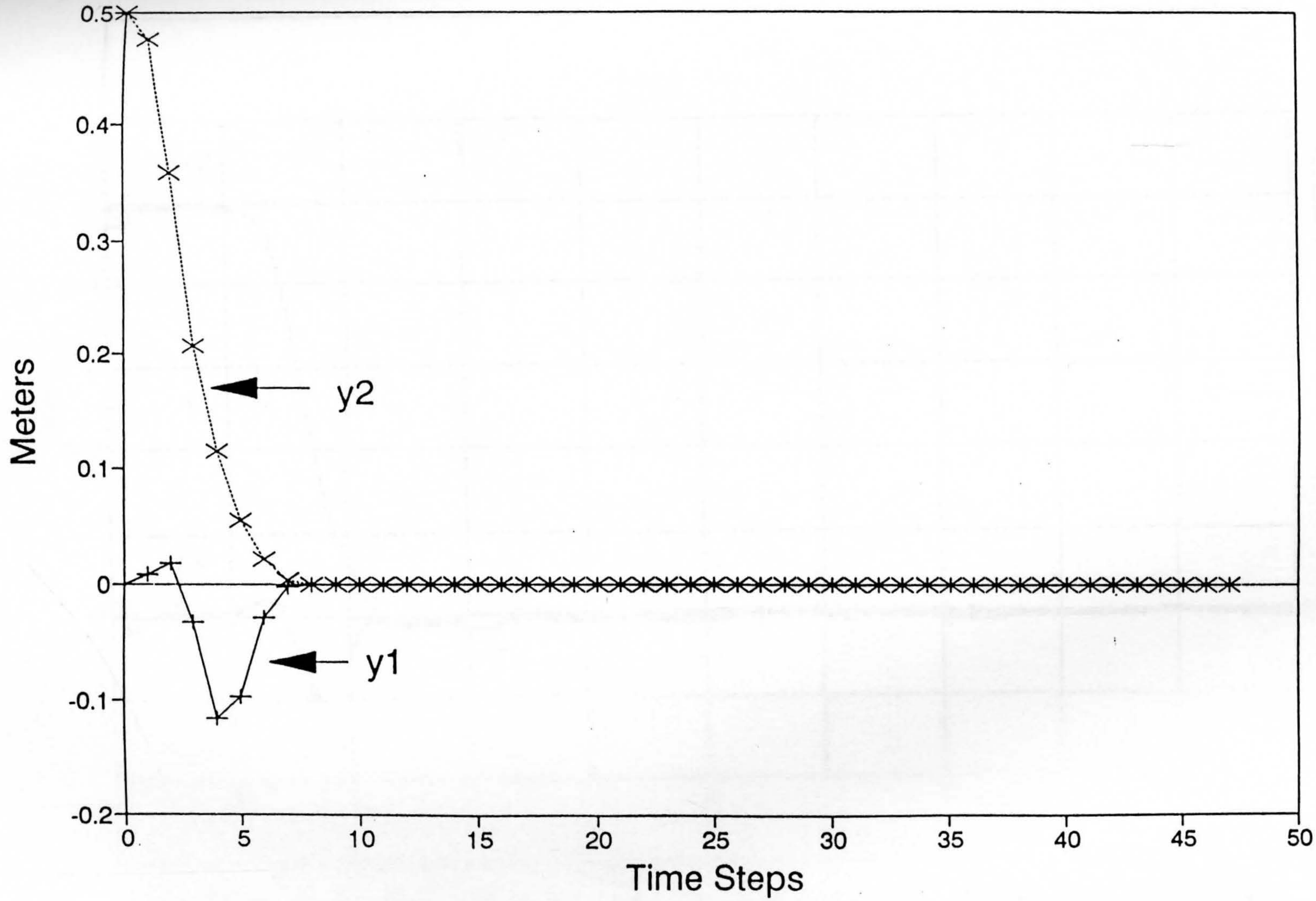

Figure 3.11.1: Simulated Tracking Output Response to x(0)=[0 0 0.5 0]' and $\hat{x}(0)$=[0 0 0.3 0]' with "Eigen-Controller" and "Eigen-Observer".

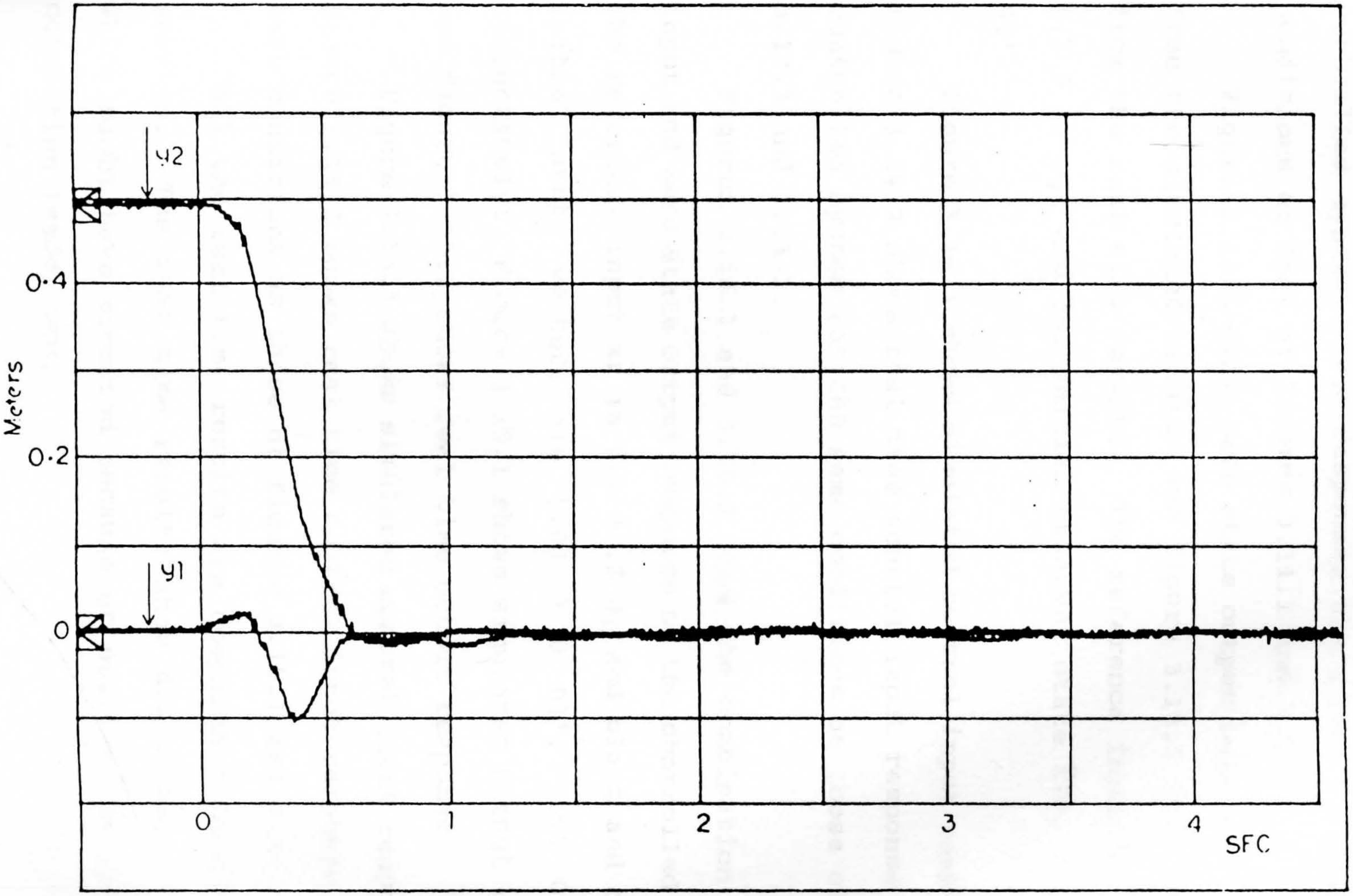

Figure 3.11.2: Real Time Tracking Output Response to x(0)=[0 0 0.5 0]' and $\hat{x}(0)$=[0 0 0.3 0]' with "Eigen-Controller" and "Eigen-Observer".

Figure 3.12.1 shows simulated control input response and figure 3.12.2 shows real time control input response of the controlled system. The responses are plotted for the same conditions as that of figures 3.11.1 and 3.11.2.

Figure 3.13.1 shows zero state output responses obtained from the simulated results and figure 3.13.2 shows the same from the real time results. The reference input state x_r is [.2 0 .3 0]' and the initial observer state $\hat{x}(0)$ is [.5 0 0 0]'.

Figure 3.14.1 shows simulated control input response and figure 3.14.2 shows real time control input response of the controlled system for the same conditions as those of figure 3.13.1 and 3.13.2.

Figures 3.15.1 and 3.15.2 show the combination of zero input and zero state output response of the controlled system. The reference input xr is [.2 0 .3 0]'and plant and observer initial state vectors are [.03 0 0 0]', [.01 0 0 0]' respectively. Figure 3.15.1 shows simulated output response and figure 3.15.2 shows real time output response.

Figure 3.16.1 shows simulated control input response and figure 3.16.2 shows real time control input response for the same conditions as those of figures 3.15.1 and 3.15.2.

All the real time results are comparable to simulation results. The real time results show disturbance or noise which might have occurred because of ocilloscope circuitry, connection leads, etc.

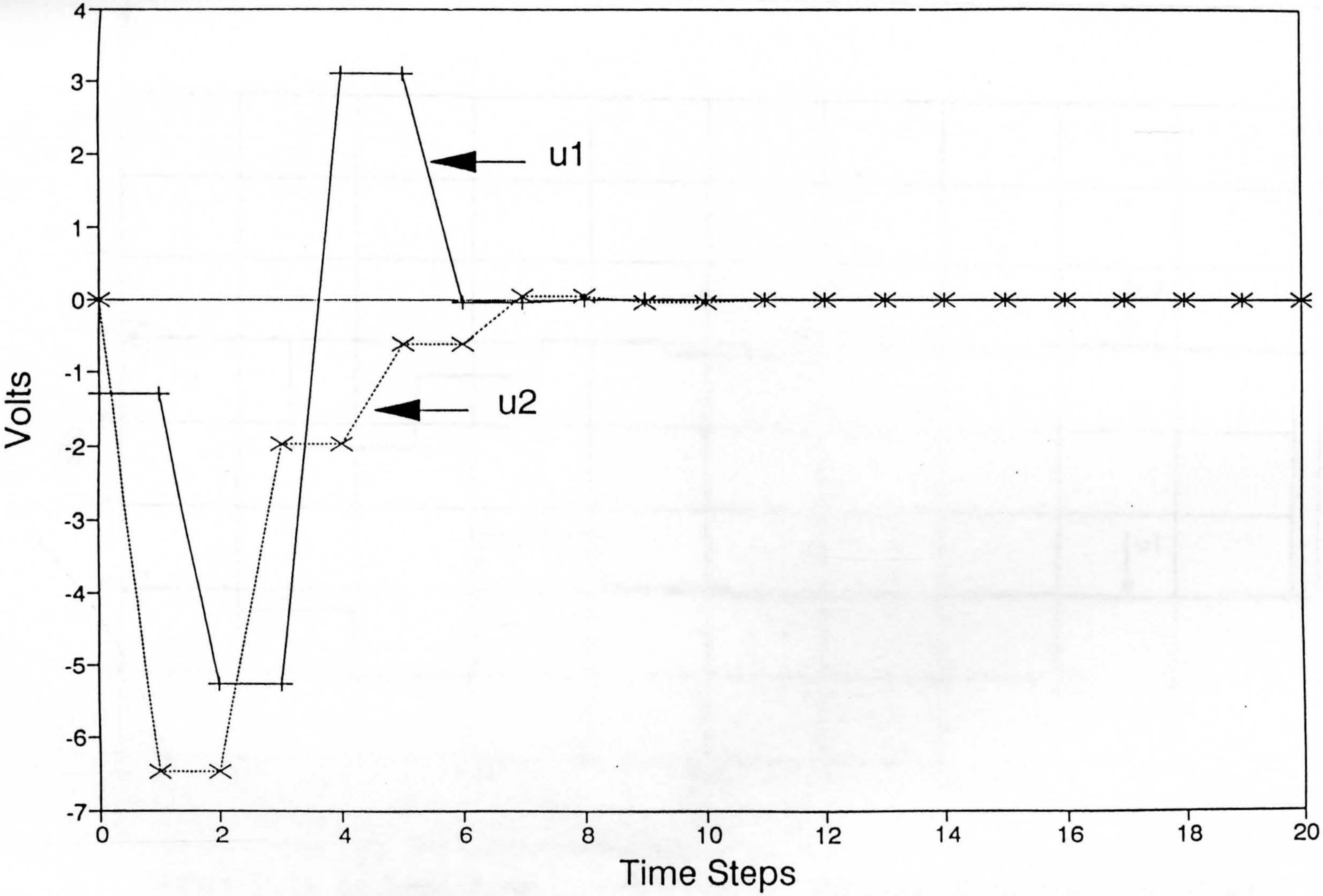

Figure 3.12.1: Simulated Control Signal Response to x(0)=[0 0 0.5 0]' and $\hat{x}$(0)=[0 0 0.3 0]' with "Eigen-Controller" and "Eigen-Observer".

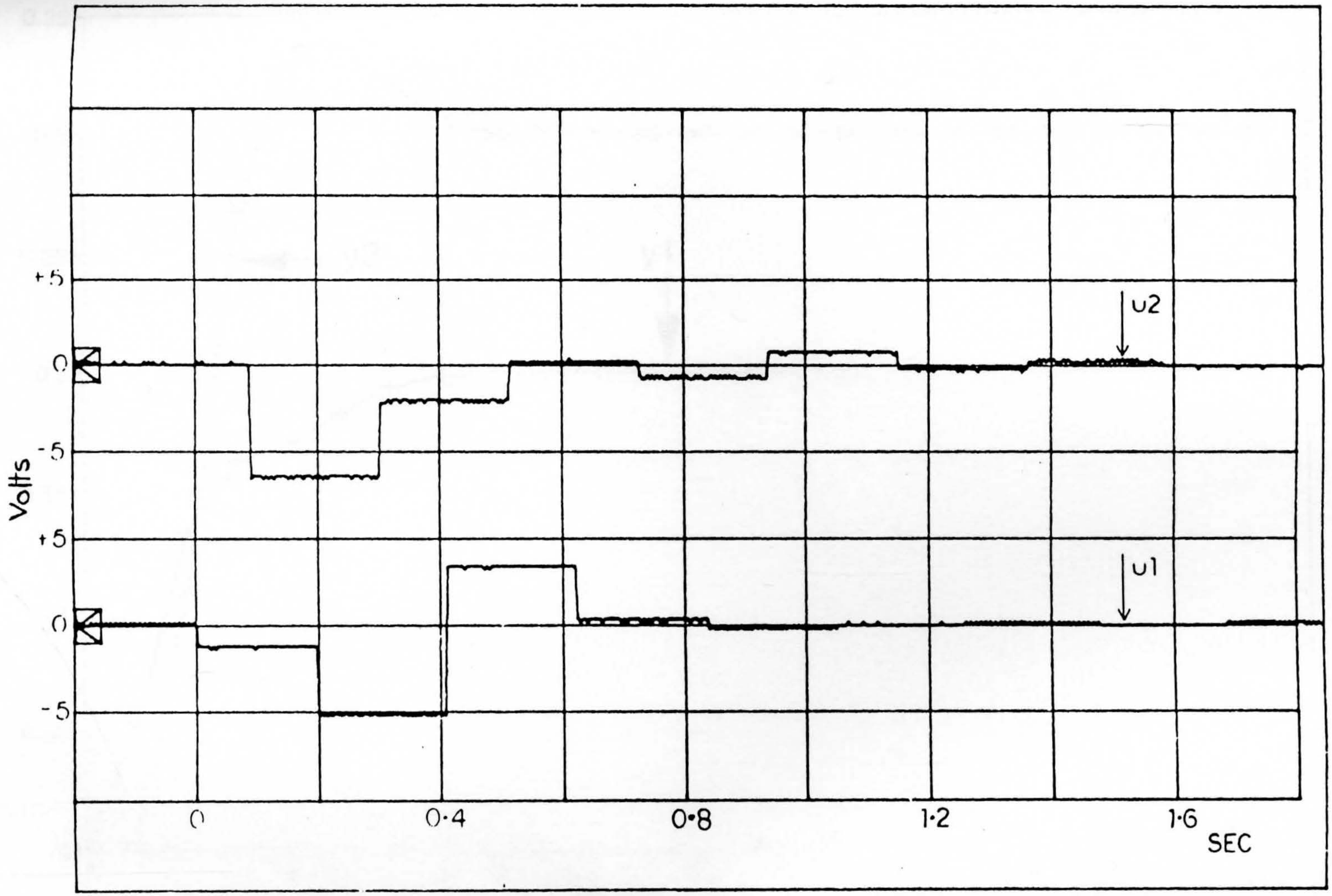

Figure 3.12.2: Real Time Control Signal Response to x(0)=[0 0 0.5 0]' and $\hat{x}(0)$=[0 0 0.3 0]' with "Eigen-Controller" and "Eigen-Observer".

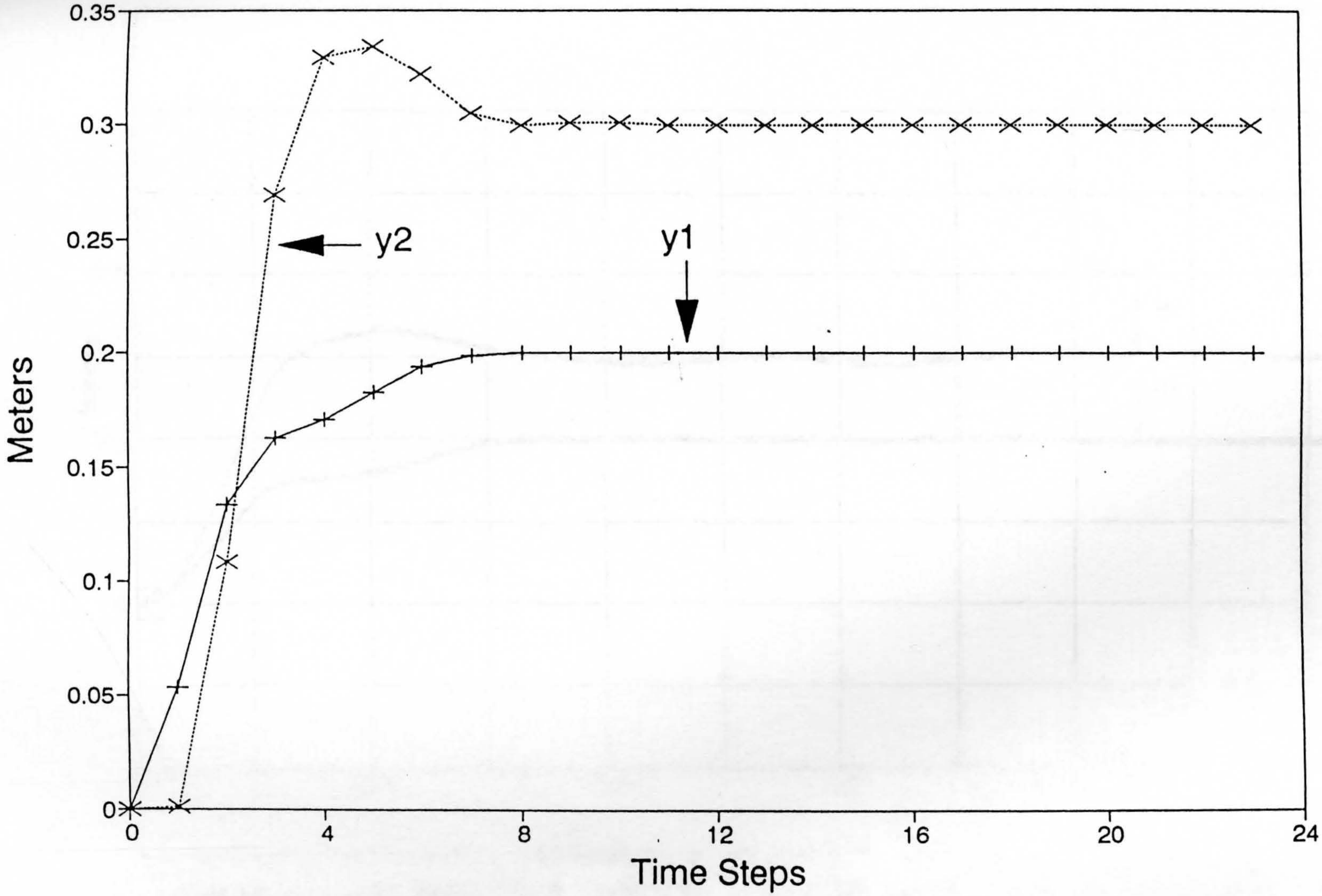

Figure 3.13.1: Simulated Tracking Output Response to x_r=[0.2 0 0.3 0]' and $\hat{x}(0)$=[0.05 0 0 0]' with "Eigen-Controller" and "Eigen-Observer".

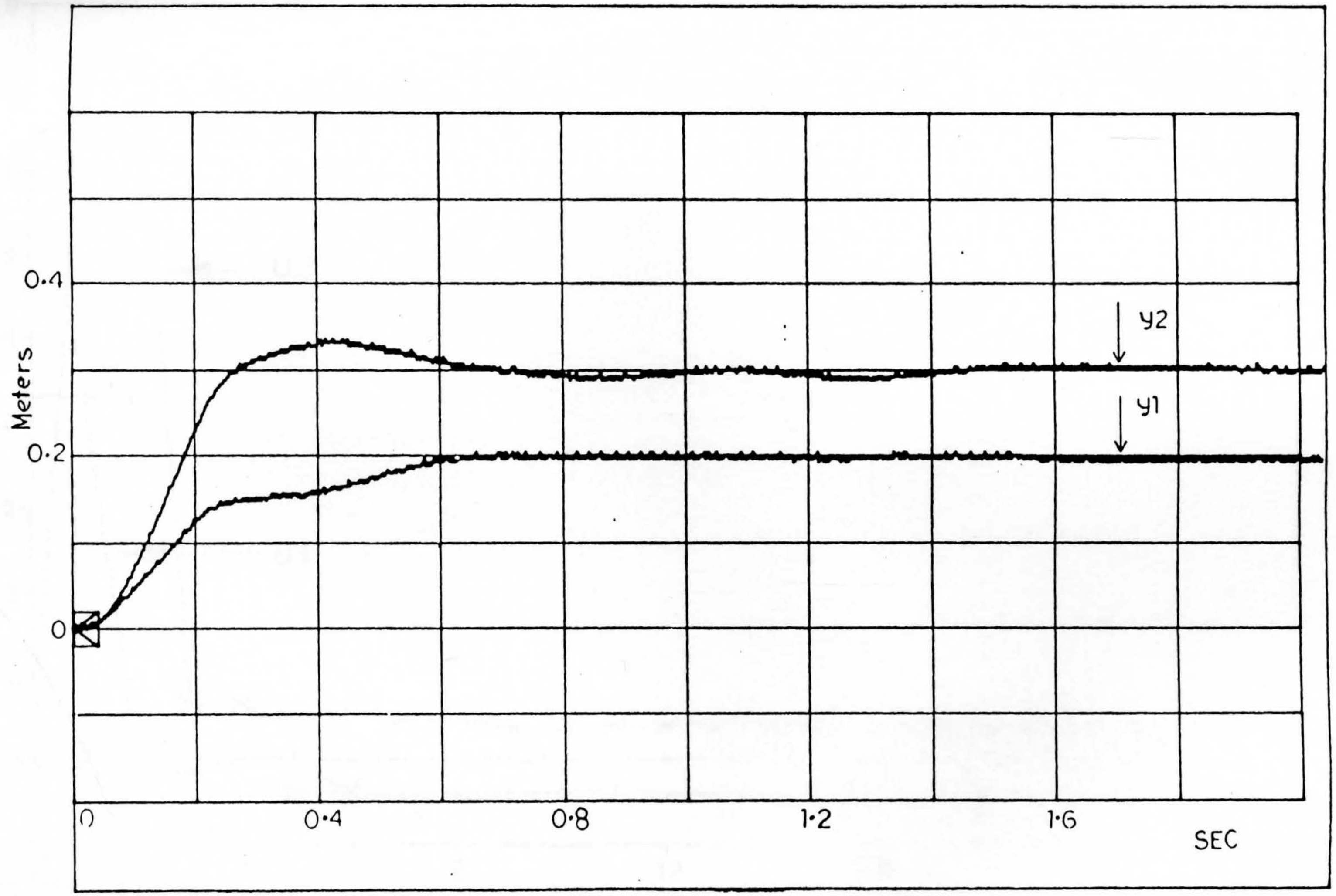

Figure 3.13.2: Real Time Tracking Output Response to x_r=[0.2 0 0.3 0]' and $\hat{x}(0)$=[0.05 0 0 0]' with "Eigen-Controller" and " Eigen-Observer".

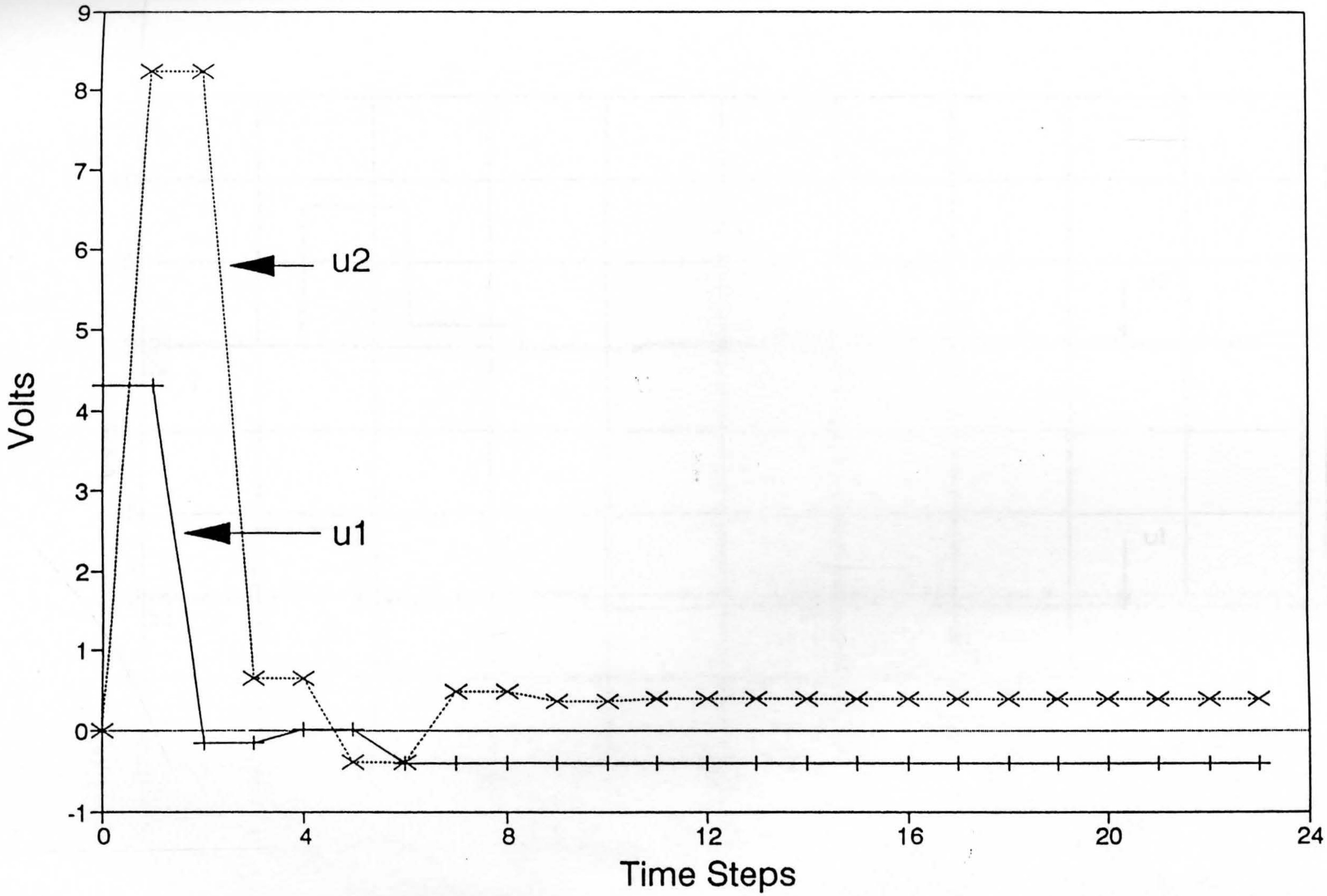

Figure 3.14.1: Simulated Control Signal Response to x_r=[0.2 0 0.3 0]' and $\hat{x}(0)$=[0.05 0 0 0]' with "Eigen-Controller" and "Eigen-Observer".

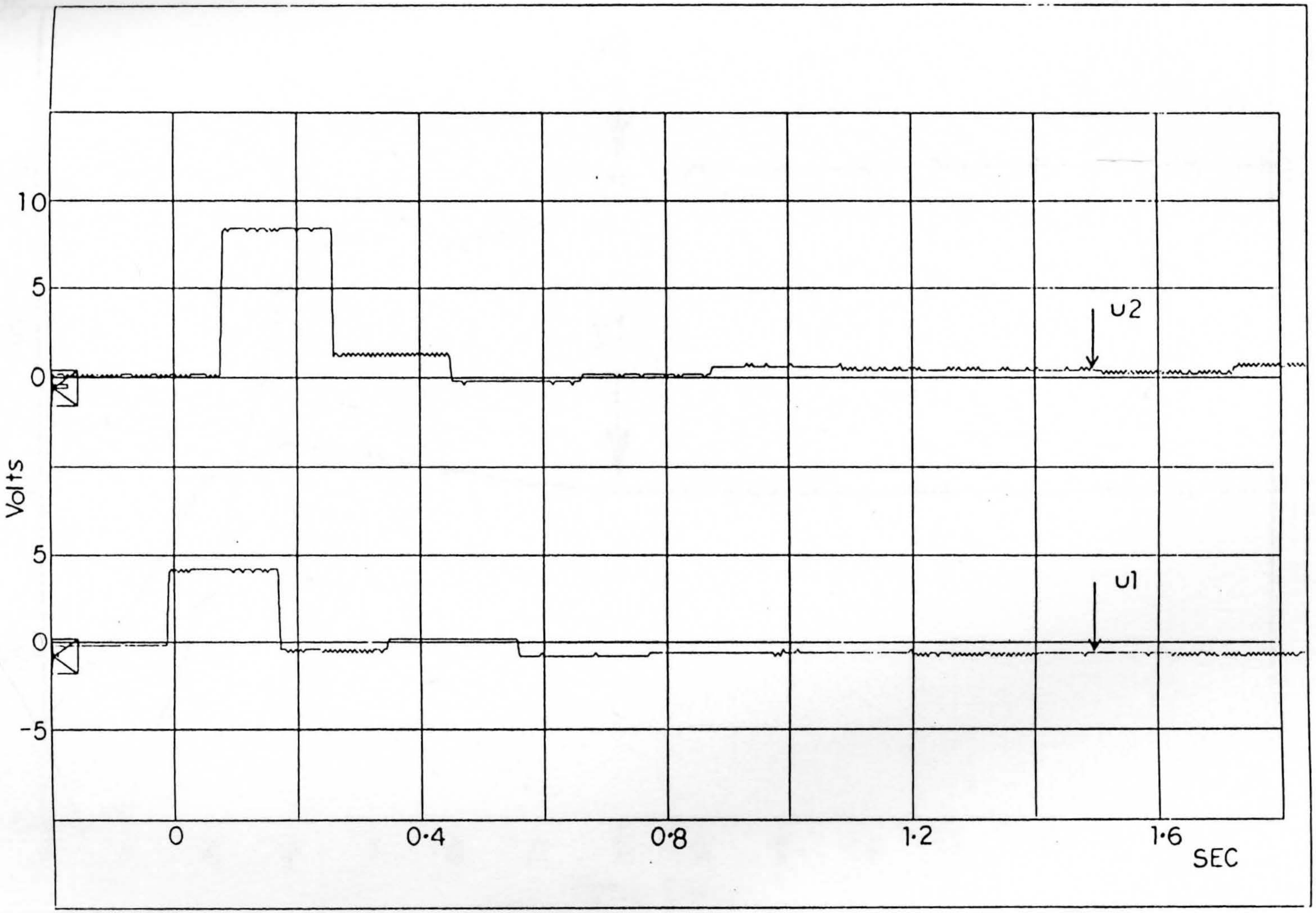

Figure 3.14.2: Real Time Control Signal Response to x_r=[0.2 0 0.3 0]' and $\hat{x}(0)$=[0.05 0 0 0]' with "Eigen-Controller" and "Eigen Observer".

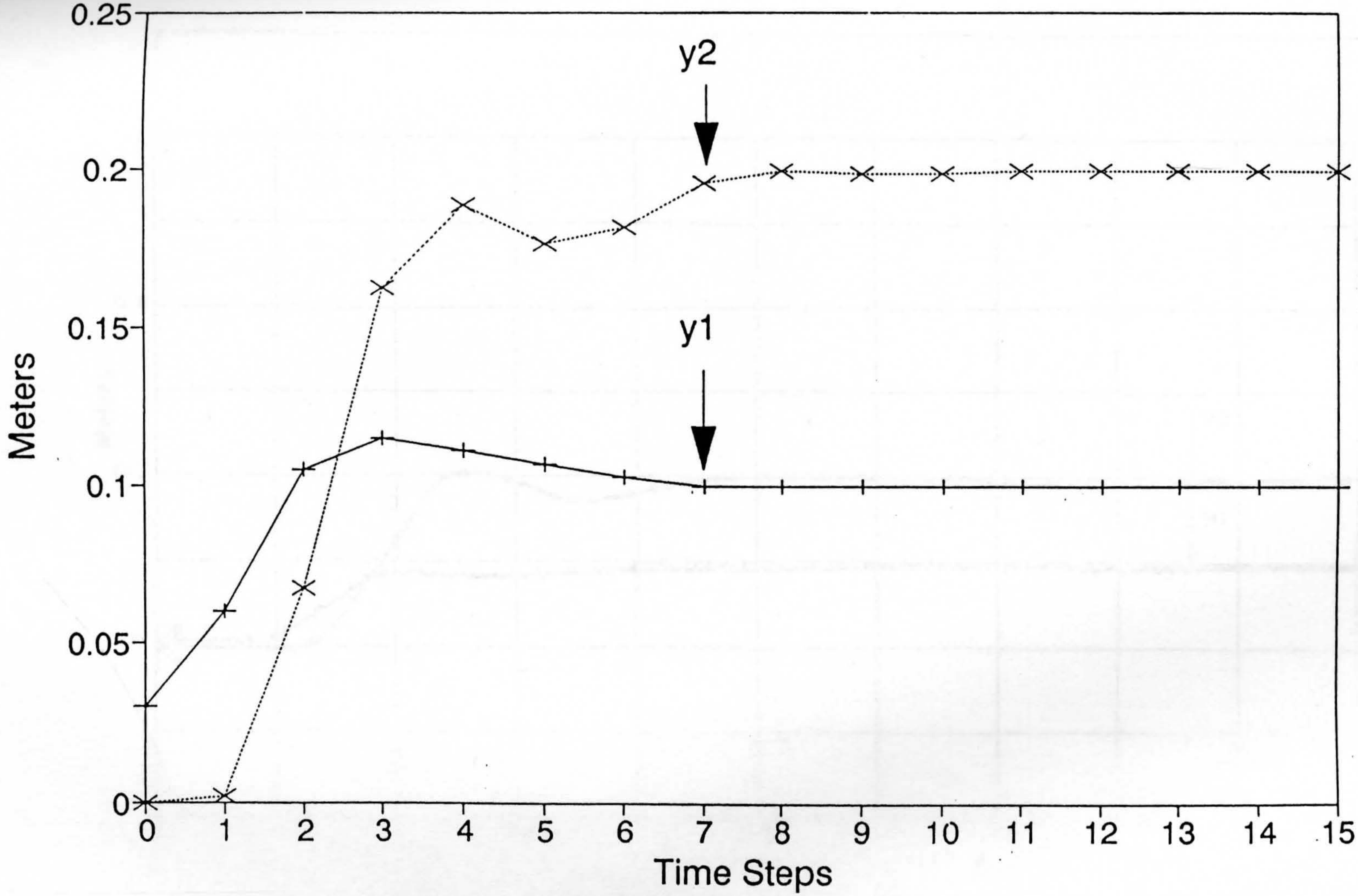

Figure 3.15.1: Simulated Tracking Output Response to x_r=[0.1 0 0.2 0]', x(0)=[0.03 0 0 0]' and $\hat{x}$(0)=[0.01 0 0 0]' with "Eigen-Controller" and "Eigen-Observer".

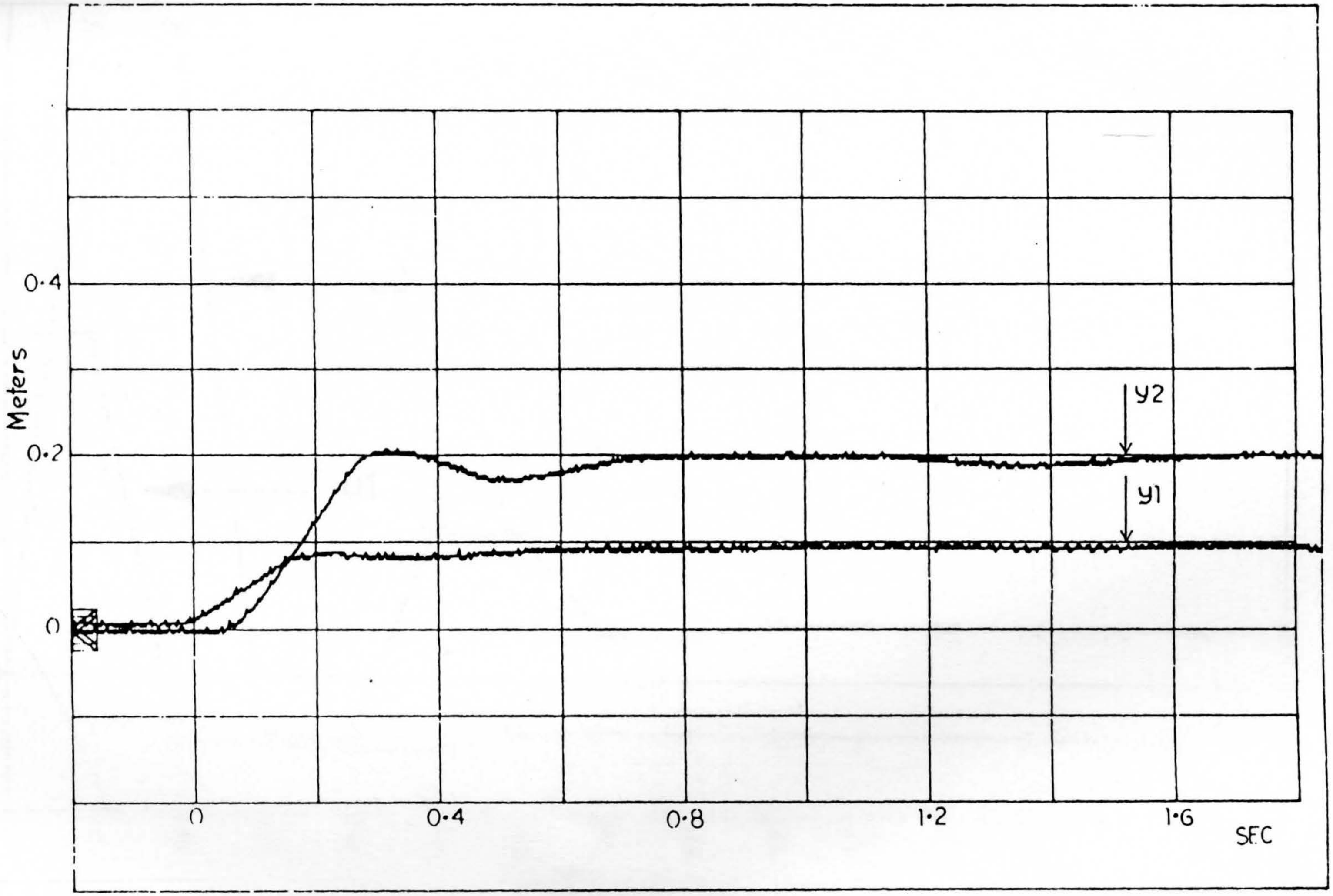

Figure 3.15.2: Real Time Tracking Output Response to x_r=[0.1 0 0.2 0]', x(0)=[0.03 0 0 0]' and $\hat{x}$(0)=[0.01 0 0 0]' with "Eigen-Controller" and "Eigen-Observer".

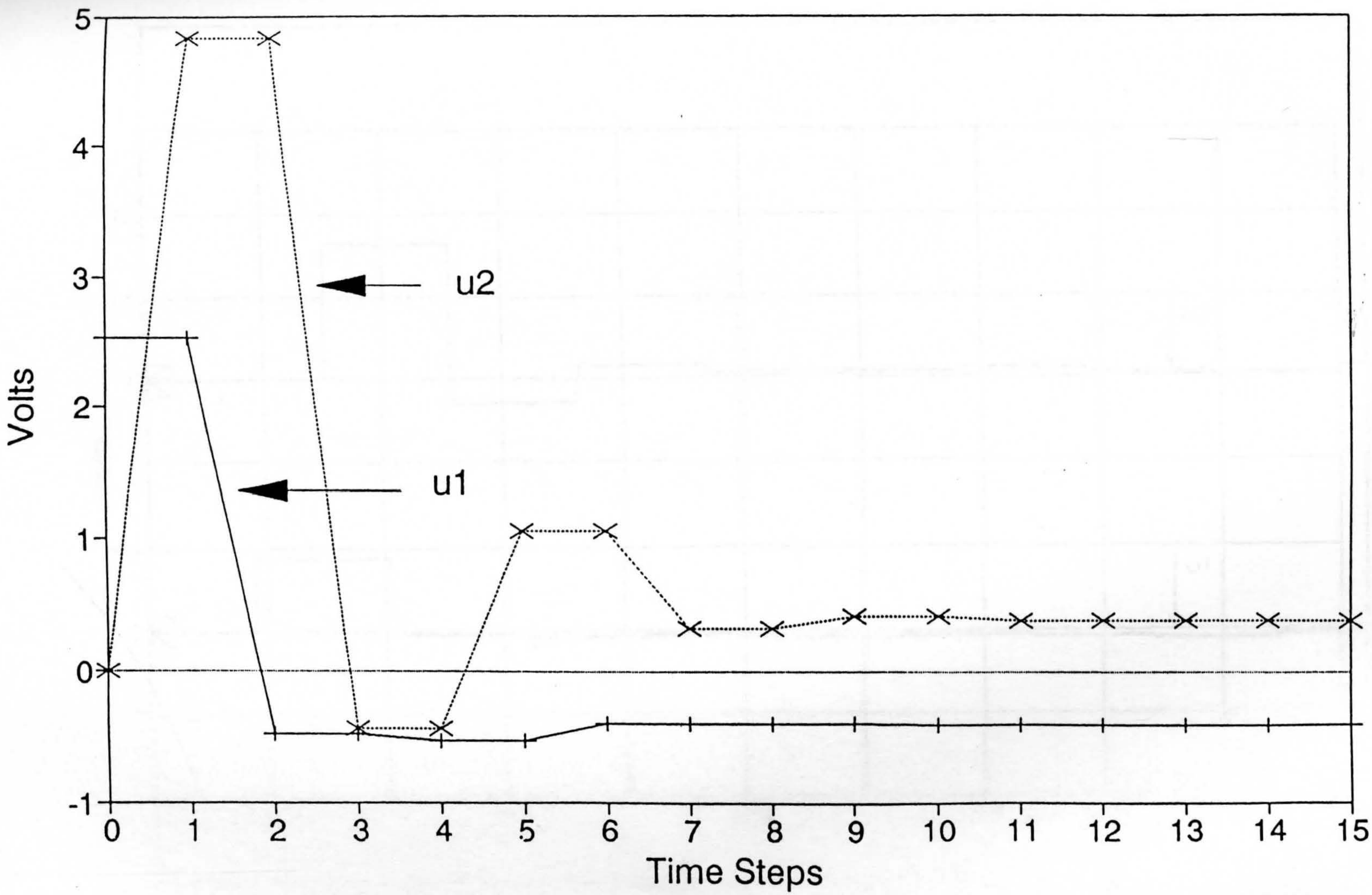

Figure 3.16.1: Simulated Control Signal Response to x_r=[0.1 0 0.2 0]', x(0)=[0.03 0 0 0]' and $\hat{x}$(0)=[0.01 0 0 0]' with "Eigen-Controller" and "Eigen-Observer".

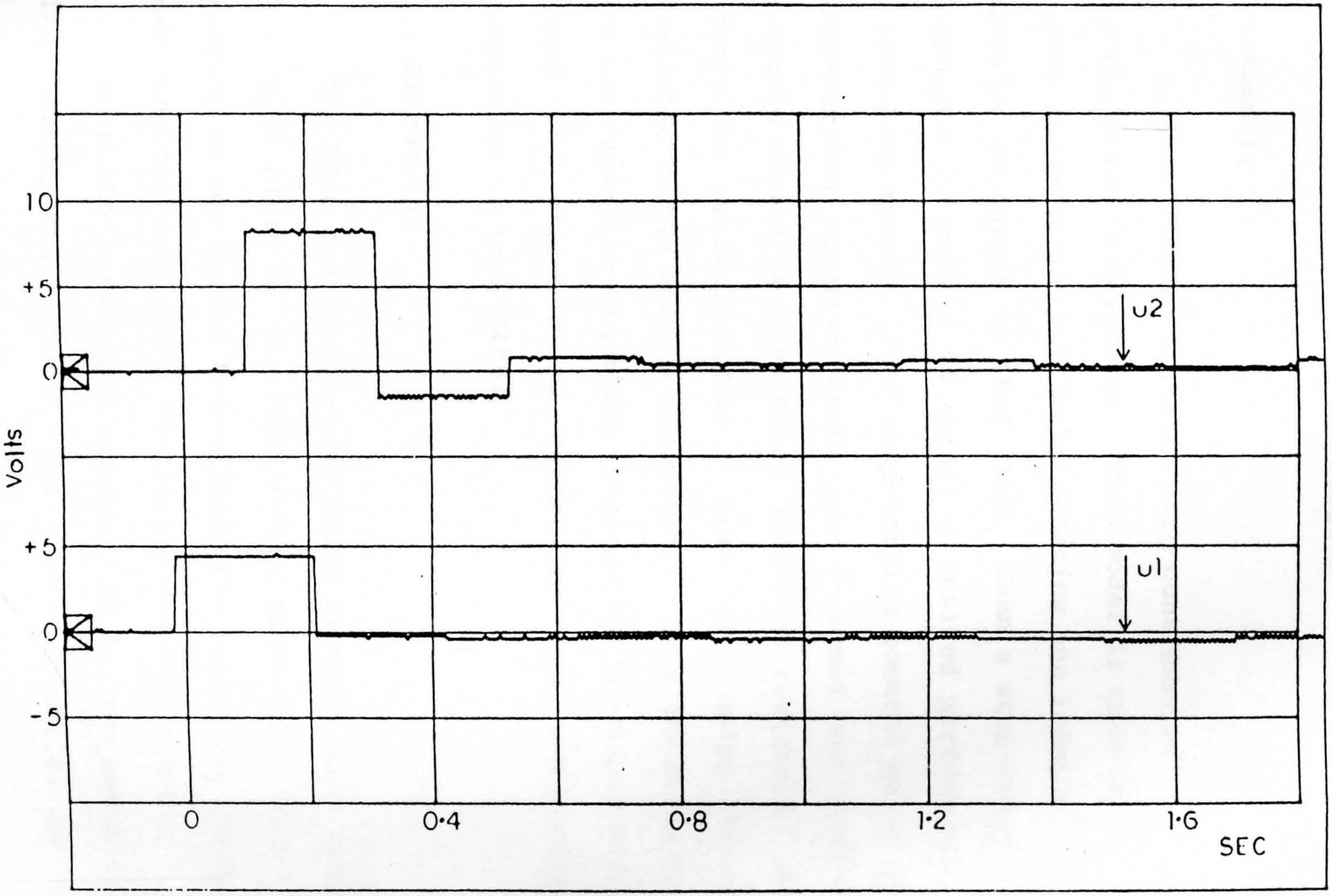

Figure 3.16.2: Real Time Control Signal Response to x_r=[0.1 0 0.2 0]', x(0)=[0.03 0 0 0]' and $\hat{x}$(0)=[0.01 0 0 0]' with "Eigen-Controller" and "Eigen-Observer".

CHAPTER IV

DISCUSSION

4.1 Summary

In this thesis, a state space continuous-time MIMO model for the plant is developed. This model is then converted into a SISO step-varying system by updating one input at a time and sampling an output at that time. Then a step-varying design technique is applied to obtain a desired performance of the closed-loop system. Four step-varying control gains and four step-varying observer gains are designed and simulated in order to have a comparison of different responses. The best controller is implemented in real time, using the hardware available in the control laboratory. The simulation responses and the real-time responses obtained for the application plant using the step-varying controller are satisfactory and comparable to each other.

4.2 Discussion

The MIMO continuous-time plant is marginally stable. The real-time plots of the MIMO analog plant show that tracking outputs of the system do not track the reference inputs, which explains the need for a controller for the plant.

From the simulation results, it is concluded that, for the two-car train plant, the eigen-controller is the best among the four sets of control gains.

When there is no observer error, all the sets of control gains give the same results. But when there is initial observer error, each set of control gains gives different results with the same set of observer gains. This helps us in deciding the best set of control gains among the four sets. Also it has been observed that initial observer error in x_1 and x_3 directions shows different results for a same set of control gain matrices with the same observer and with initial conditions of the same size (for example, response to $x(0)=[.1\ 0\ 0\ 0]'$, $\hat{x}(0)=[.2\ 0\ 0\ 0]'$ and response to $x(0)=[0\ 0\ .1\ 0]'$, $\hat{x}(0)=[0\ 0\ .2\ 0]'$). The results are different in that the response for the first case may be satisfactory but, for the second case, they may not be satisfactory. One reason for this behavior might be that we update input u_1 and measure output y_1 (i.e., x_1) first. Therefore, the observer error in x_1 direction is taken into account by the controller before the error in the x_3 direction. Secondly, the order of basis vectors might have some effect. This odd behavior is present only in the eigen-controller with the coordinate observer. Therefore, the coordinate observer is not preferable for the plant under consideration.

The real-time results of the closed-loop system obtained using real-time step-varying controller are comparable to those results obtained using computer simulation. The real-time results also indicate that step-varying controller works well in real-time for the two-car train plant. However, there are some problems discovered during experiments which are due to limitations of the hardware used in this thesis.

First, since the GP-6 analog computer restricts the maximum voltage to +/- 10 v, there is need to limit the control inputs applied to the plant implemented on an analog computer. It has been observed that when the control inputs are limited to +/- 10v, although actual values needed for the control purpose are higher, the controller fails to control the plant. Simulation results for the same conditions are satisfactory when no limit is used.

A second problem is that the real-time results (input and output responses) show the presence of noise. The presence of noise at steady state may be due to quantization error associated with AD/DA converters. The 7905 board has 12 bits successive approximation converters which allow error to be $20/2^{12} = 4.88$ mv for 20 v (+/- 10 v) span. The real-time results also show some noise in the transient as well as steady state response, which might have occurred because of oscilloscope circuitry, connection leads, etc.

A third problem deals with the sample period. Although precaution is taken to have the sampling period exactly equal to .1 seconds, sometimes the sampling period is observed to be slightly smaller or larger.

In general, the deadbeat step-varying controller worked well for the two-car train plant.

While designing the step-varying controller we have experienced some differences in step-invariant design techniques and step-varying design techniques. In step-invariant feedback control design, a designer selects eigenvalues or poles of the closed-loop system according to design specifications, such as settling time and maximum overshoot. He may try different eigenvalues depending upon the closed-loop response obtained and control required for that purpose. So, in step-invariant design, there is a definite first step and then the designer may have to follow trial-and-error procedure.

In the step-varying design method also, we use feedback but it is step-varying. Since the system is a step-varying system, the designer does not select eigenvalues of the closed-loop system. However, he must select a set of basis vectors to use in calculating the gains. Since no rules exist for selecting the basis vectors, a trial-and-error procedure is used.

4.3 Suggestions for future study

Conversion of MIMO continuous time-invariant plant into SISO step-varying system and then application of step-varying design technique is a vast subject. There is need of research on lots of topics. Some of those are given below.

1. To convert the MIMO plant under consideration, into SISO step-varying system a specific input update pattern and output sampling pattern is tried. Another type of input update pattern and output sampling pattern can be tried and results can be compared.
2. For the given plant, the eigen-controller is the best.

More application examples have to be tried to generalize this conclusion. Also eigenvalues of the MIMO discrete time-invariant system are real for our system, so that we can use eigenvectors as the basis vectors, but for a system with complex eigenvalues, it is not possible.

3. To calculate control and observer gains only one order of the basis vectors is tried. Further analysis is needed to determine what effect the order of the basis vectors has on control and observer gain matrices and also on the responses of the controlled system.

4. In this thesis, only a deadbeat step-varying controller design is tried, in which case, the error between tracking outputs and the reference inputs goes to zero in n steps. Hostetter [2] has suggested another step-varying controller in which error goes to zero progressively depending upon the value of a parameter α. Effect of α on the response of the close-loop system is an another topic for further study.

5. An interesting point noted in deadbeat step-varying control and observer design is that control gain matrices at 0^{th} step are same for all four sets of basis vectors and observer gain matrices are same at the 0^{th} step for all four sets of basis vectors. This might be a general theoretical property, which needs to be investigated.

6. Sampling period of .1 second is selected under the assumption that it might take .1 second to run the step-varying control algorithm in real time. It has been observed that using QBASIC only 20 milliseconds sampling time is

sufficient to run the algorithm in real time if EVEREX-286 (16 Mhz) computer is used. Improvement in closed-loop system response can be observed by designing the system for a smaller sampling period.

APPENDIX A

This appendix gives the iterative method of calculating the control gain matrices for the SISO step-varying system developed by Dr. Foulkes.

In vector form, the error state equation for SISO, step-varying system are (from section 2.4.2)

$$e(k+1) = A(k)\ e(k) + D(k)\ \overline{v}(k) + B(k)\ x_r(k),$$

$$\overline{v}(k+1) = -\ \overline{F}(k)\ e(k) - \overline{L}(k)\ \overline{v}(k) - \overline{G}(k)\ x_r(k),$$

and

$$x_r(k+1) = \Phi_r\ x_r(k) \qquad \text{.........................(a.1)}$$

where

$e(k)$, $x_r(k)$, $\xi_1(k)$, ..., $\xi_r(k)$ each is a n x 1 vector,

$v(k)$ is a (r-1) x 1 vector,

$F(k)$ is a (r-1) x n matrix,

$G(k)$ is a (r-1) x n matrix,

$L(k)$ is a (r-1) x (r-1) matrix,

$F(k)$ is a 1 x n feedback gain matrix,

$G(k)$ is a 1 x n feedforward gain matrix,

v_1 ,..., v_{r-1} are previous scalar inputs,

$l_1(k)$,.., $l_{r-1}(k)$ are scalar gains on the previous inputs.

Also,

$$\bar{v}(k) = \begin{bmatrix} v_1(k) \\ v_2(k) \\ \cdot \\ \cdot \\ \cdot \\ v_{r-1}(k) \end{bmatrix}$$

$$A(k) = [\ \Phi - \xi_1(k)\ F(k)\ ]$$

$$D(k) = [\ (\ \xi_2(k) - \xi_1(k)\ l_1(k)\ )\ \ldots.(\ \xi_r(k) - \xi_1(k)\ l_{r-1}(k)\ )\ ]$$

$$B(k) = [\ \Phi - \Phi_r - \xi_1(k)\ G(k)\ ]$$

$$\bar{L}(k) = \begin{bmatrix} l_1(k) & l_2(k) & . & . & . & l_{r-1}(k) \\ -1 & 0 & . & . & . & 0 \\ 0 & -1 & . & . & . & 0 \\ \cdot & & & & & \\ \cdot & & & & & \\ 0 & 0 & . & . & -1 & 0 \end{bmatrix}$$

$$\bar{G}(k) = \begin{bmatrix} G(k) \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

$$\bar{F}(k) = \begin{bmatrix} F(k) \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

Note that n is the order of MIMO step-invariant system.

At n^{th} step,

$$e(n) = A(n-1)\ e(n-1) + D(n-1)\ \overline{v}(n-1) + B(n-1)\ x_r(n-1) \quad \text{..(a.2)}$$

Now suppose,

$$e(n) = \eta(k)\ e(n-k) + \theta(k)\ \overline{V}(n-k) + \psi(k)\ x_r(n-k) \quad \text{........(a.3)}$$

where

$\eta(k)$ is a n x n vector,

$\theta(k)$ is a n x (r-1) vector,

$\psi(k)$ is a n x n vector.

Putting the expressions for $e(n-k)$, $\overline{v}(n-k)$, $x_r(n-k)$ in terms of $e(n-k-1)$, $\overline{v}(n-k-1)$, $x_r(n-k-1)$ by using the relation (a.1), and simplifying,

$$\begin{aligned} e(n) = {} & [\eta(k)\ A(n-k-1) - \theta(k)\ \overline{F}(n-k-1)]\ e(n-k-1) \\ & + [\eta(k)\ D(n-k-1) - \theta(k)\ \overline{L}(n-k-1)]\ \overline{v}(n-k-1) \\ & + [\eta(k)\ B(n-k-1) - \theta(k)\ \overline{G}(n-k-1) + \psi(k)\ \Phi_r]\ x_r(n-k-1) \end{aligned}$$

Comparing the above equation with equation (a.3), we have

$$e(n) = \eta(k+1)\ e(n-k-1) + \theta(k+1)\ \overline{v}(n-k-1) + \psi(k+1)\ x_r(n-k-1)$$

which gives the following recursive expressions for $\eta(k)$, $\theta(k)$, $\psi(k)$:

$$\eta(k+1) = \eta(k)\ A(n-k-1) - \theta(k)\ \overline{F}(n-k-1)]\ e(n-k-1)$$

$$\theta(k+1) = [\eta(k)\ D(n-k-1) - \theta(k)\ \overline{L}(n-k-1)]\ \overline{v}(n-k-1)$$

$$\psi(k+1) = [\eta(k)\ B(n-k-1) - \theta(k)\ \overline{G}(n-k-1) + \psi(k)\ \Phi_r]\ x_r(n-k-1)$$

$$\text{....................................(a.4)}$$

To find out boundary conditions, that is, expressions for $\eta(0)$, $\Theta(0)$, $\psi(0)$, put k=0 in equation (a.3)

$$\therefore \; e(n) = \eta(0)\, e(n) + \theta(0)\, \overline{v}(n) + \psi(0)\, x_r(n)$$

Therefore to have e(n)= e(n) choose

$$\eta(0) = I, \quad \theta(0) = 0, \quad \psi(0) = 0$$

Now, in order for the error to go to zero in n steps, choose $j_1, \ldots j_n$ as n basis vectors. Then use,

$$j_k' \, e(n) = 0 \qquad for \;\; k = 1,2,3,\ldots.n$$

to calculate the gain matrices at all the n steps.

For example, consider k=1

$$\therefore \; j_1' \, e(n) = j_1' \, [\eta(1)\, e(n-1) + \theta(1)\, \overline{v}(n-1) + \psi(1)\, x_r(n-1)] = 0$$

$$\therefore \; j_1' \, \eta(1) = 0$$

Then using equation (a.4) and boundary conditions we have,

$$j_1' \, [\eta(0)\, A(n-1) - \theta(0)\, \overline{F}(n-1)] = 0$$

which gives

$$F(n-1) = \frac{j_1'[\Phi - I\,\alpha]}{j_1' \, \xi_1(n-1)}$$

Also

$$j_1' \, \theta(1) = 0$$

$$\therefore j_1' \, [[\xi_2(n-1) - \xi_1(n-1)\, l_1(n-1)] \ldots [\xi_r(n-1) - \xi_1(n-1)\, l_{r-1}(n-1)]] = 0$$

which gives

$$l_1(n-1) = \frac{j_1' \, \xi_2(n-1)}{j_1' \, \xi_1(n-1)}$$

$$\vdots$$

$$l_{r-1}(n-1) = \frac{j_1' \, \xi_r(n-1)}{j_1' \, \xi_1(n-1)}$$

Also

$$j_1' \, \psi(1) = 0$$

$$\therefore \; j_1' \, B(n-1) = 0$$

$$\therefore \; j_1' \, [\Phi - \Phi_r - \xi_1(n-1) \, G(n-1)] = 0$$

which gives

$$G(n-1) = \frac{j_1'[\Phi - \Phi_r]}{j_1' \, \xi_1(n-1)}$$

Once F(n-1), $\bar{L}$(n-1), G(n-1) are known, we can find A(n-1), D(n-1), B(n-1) and hence $\eta(1)$, $\theta(1)$, $\psi(1)$. Then putting k=2 in equation (a.3) we have,

$$j_2' \, e(n) = [\eta(2) \, e(n-2) + \theta(2) \, \bar{v}(n-2) + \psi(2) \, x_r(n-2)] = 0$$

which gives,

$$F(n-2) = \frac{j_2' \, \eta(1) \, \Phi}{j_2' \, [\eta(1) \, \xi_1(n-2) + \theta_1(1)]}$$

$$l_1(n-2) = \frac{j_2'[\eta(1) \, \xi_2(n-2) + \theta_2(1)]}{j_2'[\eta(1) \, \xi_1(n-2) + \theta_1(1)]}$$

.

.

$$l_{r-1} = \frac{j_2'[\eta(1)\ \xi_r(n-2)]}{j_2'[\eta(1)\ \xi_1(n-2) + \theta_1(1)]}$$

$$G(n-2) = \frac{j_2'[\eta(1)\ \Phi - \eta(1)\ \Phi_r + \psi(1)\ \Phi_r]}{j_2'[\eta(1)\ \xi_1(n-2) + \theta_1(1)]}$$

where θ_i (1) is the i^{th} column of $\theta(1)$.

In general, at $(k+1)^{th}$ step,

$$j_{k+1}'\ e(n) = j_{k+1}'[\eta(k+1)\ e(n-k-1) + \theta(k+1)\ \overline{v}(n-k-1) + \psi(k+1)\ x_r(n-k-1)] = 0$$

$$F(n-(k+1)) = \frac{j_{k+1}'\ \eta(k)\ \Phi}{j_{k+1}'[\eta(k)\ \xi_1(n-(k+1)) + \theta_1(1)]}$$

$$l_1(n-(k+1)) = \frac{j_{k+1}'[\eta(k)\ \xi_2(n-(k+1)) + \theta_2(k)]}{j_{k+1}'[\eta(k)\ \xi_1(n-(k+1)) + \theta_1(k)]}$$

.

.

.

$$l_{r-1}(n-(k+1)) = \frac{j_{k+1}'[\eta(k)\ \xi_r(n-(k+1))]}{j_{k+1}[\eta(k)\ \xi_1(n-(k+1)) + \theta_1(k)]}$$

$$G(n-(k+1)) = \frac{j_{k+1}'[\eta(k)\ \Phi - \eta(k)\ \Phi_r + \psi(k)\ \Phi_r]}{j_{k+1}'[\eta(k)\ \xi_1(n-(k+1) + \theta_1(k)]} \qquad \text{.....(a.5)}$$

Therefore to calculate the gain matrices in iterative way, follow the following steps, start with k=0,

1. Then use equation (a.4) with boundary conditions to find $\eta(k+1)$, $\theta(k+1)$, $\psi(k+1)$.

2. Compute $F(n-(k+1))$, $\bar{L}(n-(k+1))$, $G(n-(k+1))$ using equation (a.5).

3. Increase k by 1 and go back to step 1 if k is less than n.

APPENDIX B

Listing of a Program CC macro used to calculate the control gain matrices for a set of unit coordinate vectors.

```
state
p1z
cc
j1=(1,0,0,0)
j2=(0,1,0,0)
j3=(0,0,1,0)
j4=(0,0,0,1)
zeta2e=(1.703e-4;5.458e-3;1.248e-2;1.781e-1)
zeta2o=(1.248e-2;1.781e-1;1.703e-4;5.458e-3)
zeta1e=zeta2o
zeta1o=zeta2e
f3=(j1*p1z(a))/(j1*zeta1o)
f3
l3=(j1*zeta2o)/(j1*zeta1o)
l3
g3=(j1*(p1z(a)-i))/(j1*zeta1o)
g3
pause
a3=p1z(a)-zeta1o*f3
d3=zeta2o-zeta1o*l3
b3=p1z(a)-i-zeta1o*g3
f2=(j2*a3*p1z(a))/(j2*(a3*zeta1e+d3))
f2
l2=(j2*a3*zeta2e)/(j2*(a3*zeta1e+d3))
l2
g2=(j2*(a3*p1z(a)-a3*i+b3*i))/(j2*(a3*zeta1e+d3))
g2
pause
a2=p1z(a)-zeta1e*f2
d2=zeta2e-zeta1e*l2
b2=p1z(a)-i-zeta1e*g2
f1=(j3*(a3*a2*p1z(a)-d3*f2*p1z(a)))/(j3*(a3*a2*zeta1o-d3*f2*zeta1o+a3*d2-d3*l2))
f1
l1=(j3*(a3*a2*zeta2o-d3*f2*zeta2o))/(j3*(a3*a2*zeta1o-d3*f2*zeta1o+a3*d2-d3*l2))
l1
g1=(j3*(a3*a2*p1z(a)-a3*a2*i-d3*f2*p1z(a)+d3*f2*i+a3*b2*i-d3*g2*i+b3*i*i))/(j3*(a3*a2*zeta1o-d3*f2*z
eta1o+a3*d2-d3*l2))
g1
a1=p1z(a)-zeta1o*f1
d1=zeta2o-zeta1o*l1
b1=p1z(a)-i-zeta1o*g1
pause
f0=(j4*(a3*a2*a1*p1z(a)-d3*f2*a1*p1z(a)-a3*d2*f1*p1z(a)+d3*l2*f1*p1z(a)))
denom=j4*((a3*a2*a1-d3*f2*a1-a3*d2*f1+d3*l2*f1)*zeta1e+a3*a2*d1-d3*f2*d1-a3*d2*l1+d3*l2*l1)
f0=f0/denom
l0=(j4*(a3*a2*a1-d3*f2*a1-a3*d2*f1+d3*l2*f1)*zeta2e)/denom
g0=(j4*((a3*a2*a1-d3*f2*a1-a3*d2*f1+d3*l2*f1)*(p1z(a)-i)+(a3*a2*b1-d3*f2*b1-a3*d2*g1+d3*l2*g1+a3*b2*
i-d3*g2*i+b3*i*i)*i))/(denom)
f0
l0
g0
```

APPENDIX C

Listing of a Program CC macro used to calculate the observer gain matrices for a set of obsevability vectors.

```
state
p1z
c0=(1,0,0,0)
c1=(0,0,1,0)
j1=(1.248633e-2;1.781246e-1;1.703513e-4;5.4584e-3)
j1
j2=(1.703513e-4;5.4584e-3;1.248633e-2;1.781246e-1)
j2
j3=(1.823123e-2;2.321339e-3;1.165995e-3;1.274807e-2)
j3
j4=(1.165995e-3;1.274807e-2;1.823123e-2;2.321339e-3)
j4
p0=(p1z(a)*j1)/(c0*j1)
p0
a0=p1z(a)-p0*c0
a0
p1=(p1z(a)*a0*j2)/(c1*a0*j2)
p1
a1=p1z(a)-p1*c1
p2=(p1z(a)*a1*a0*j3)/(c0*a1*a0*j3)
p2
a2=p1z(a)-p2*c0
p3=(p1z(a)*a2*a1*a0*j4)/(c1*a2*a1*a0*j4)
p3
a3=p1z(a)-p3*c1
a3*a2*a1*a0*j1
a3*a2*a1*a0*j2
a3*a2*a1*a0*j3
a3*a2*a1*a0*j4
```

APPENDIX D

Listing of QUICKC Simulation program for Step-Varying Controller.

```
       /******* SIMULATION PROGRAM FOR STEP-VARYING CONTROLLER *******/

#include<stdio.h>        /* standard I/O header file */

main()
{

float x1[4],zeta1[4],zeta2[4];
float xr[4]={0.10,0,.100,0};
float x0[4]={.0,0,.00,0};
float xe1[4];
float xe0[4]={.100,0,.0,0};

                                   /*  array of control gains    */

float f[4][4]={{2.267406e1,9.336732e-1,4.316701,1.457069e-1},
               {-6.821325,-3.138055e-1,2.256269e1,9.431117e-1},
               {2.009368e1,8.056389e-1,2.627241e1,1.0336508},
               {7.904951e1,2.902429,7.904951e1,2.902429}};

                                   /*  array of gains from referense  */

float g[4][4]={{-3.782079,5.447524,3.782079,7.01368e-2},
               {4.729583,-2.275324,-4.729583,2.90463},
               {-2.057808,-9.666447e-1,2.057808,2.808792},
               {3.341534e-13,2.902429,-3.149877e-13,2.902429}};

                                    /*  array of gains from v     */

float l[4]={5.508251e-2,-.1817109,.4855799,1};

float phi[4][4]={{9.507361e-1,3.562492e-2,4.926391e-2,1.09168e-3},
                 {-6.906648e-1,6.011309e-2,6.906648e-1,2.197191e-2},
                 {4.926391e-2,1.09168e-3,9.507361e-1,3.562492e-2},
                  {6.906648e-1,2.197191e-2,-6.906648e-1,6.011309e-2}};

float zeta2e[4]={1.703513e-4,5.4584e-3,1.248633e-2,1.781246e-1};
float zeta2o[4]={1.248633e-2,1.781246e-1,1.703513e-4,5.4584e-3};
float zeta1o[4]={1.703513e-4,5.4584e-3,1.248633e-2,1.781246e-1};
float zeta1e[4]={1.248633e-2,1.781246e-1,1.703513e-4,5.4584e-3};
int ce[4]={1,0,0,0};
int co[4]={0,0,1,0},c[4];

                              /* array of observer gains   */

float p[4][4]={{1,1.5768584e-1,9.2888806e-1,1.424716e-1},
               {-2.895292e-15,6.614367e-1,-7.216324e-1,6.74709e-1},
               {1,9.278345e-1,6.038664e-2,9.346653e-1},
               {3.031123e-15,-6.614367e-1,7.216324e-1,-6.598071e-1}};

int i,j,n,r,s;
float *ptrz1,*ptrz2;     /* pointer to array zeta1 and zeta2  */
int *ptrc;               /* pointer to array c */
float u0,v1,y0,ye0;
float v0=0.0;
FILE *fptr;              /* fptr is a file pointer */

                          /* FILE is a structure defined in STDIO.h*/

fptr=fopen("a:big2.data","w");       /* open a file */
```

```
                    /******  starting of calculations ********/
for (r=0;r<6;r=r+1)

    {

    for (n=0;n<4;n=n+1)        /*  starting of main loop   */

        {       /*  choose even & odd  n  */

        if (n%2)
        {   /*   odd choice   */
                ptrz1=zeta1o;ptrz2=zeta2o;ptrc=co;

                for (i=0;i<4;i=i+1)

                {
                zeta1[i]=*(ptrz1+i);
                zeta2[i]=*(ptrz2+i);
                c[i]=*(ptrc+i);

                }
        }

        else

        {       /*  even choice */

        ptrz1=zeta1e;ptrz2=zeta2e;ptrc=ce;
        for (i=0;i<4;i=i+1)
                {

                zeta1[i]=*(ptrz1+i);
                zeta2[i]=*(ptrz2+i);
                c[i]=*(ptrc+i);
                }
         }


                /*   calculation for uo   */

         u0=0;
         for (i=0;i<4;i=i+1)

                {

                 u0=u0-f[n][i]*xe0[i]+(f[n][i]-g[n][i])*xr[i];
                 }

          u0=u0-l[n]*v0;
          y0=0;ye0=0;
           for (i=0;i<4;i=i+1)
                {
                        y0=y0+c[i]*x0[i];
                ye0=ye0+c[i]*xe0[i];
                 }

            for (i=0;i<4;i=i+1)
                {
                x1[i]=0;
                for (j=0;j<4;j=j+1)
                        {
                         x1[i]=x1[i]+phi[i][j]*x0[j];
                         }
                         x1[i]=x1[i]+zeta2[i]*v0+zeta1[i]*u0;
                 }

                v1=u0;

                for (i=0;i<4;i=i+1)
```

```
                {
                xe1[i]=0;
                for (j=0;j<4;j=j+1)
                 {
                xe1[i]=xe1[i]+phi[i][j]*xe0[j];
                }
                xe1[i]=xe1[i]+zeta2[i]*v0+zeta1[i]*u0+p[i][n]*(y0-ye0);

                 }

         s=4*r+n;

         fprintf (fptr,"\n%d  %2.5f  %2.5f  %2.5f   %2.5f   %2.5f\n ",
                s,x0[0],x0[1],x0[2],x0[3],u0);
         printf ("\n %d  %2.5f  %2.5f  %2.5f  %2.5f  %2.5f\n ",
                s, xe0[0],xe0[1],xe0[2],xe0[3],u0);
                       /* values are updated    */
         v0=v1;
         for (i=0;i<4;i=i+1)
                {
                x0[i]=x1[i];
                xe0[i]=xe1[i];
                 }
    }                    /* loop for n ends  */

  }                   /* loop for  r ends */

  fclose(fptr);           /* file a:big2.dat is closed */

}               /*   main ends  */
```

APPENDIX E

Listing of QUICKBASIC Program for Real-Time Implementation of a Step-Varying Controller.

```
REM ***************************************************************
REM *            INTERFACING PROGRAM                              *
REM ***************************************************************

INPUT "TSAMP=", T
DIM DIO%(5)
ON KEY(1) GOSUB lable10:     KEY(1) ON      ' TO STOP THE RUN
REM  STEP 1--- INTIALIZE DAS8 WITH MODE 0  -----
MD% = 0
BASADR% = &H300
FLAG% = 0
CALL DAS8(MD%, BASADR%, FLAG%)
IF FLAG% <> 0 THEN PRINT "INSTALLATION ERROR "
REM  STEP2--- SET UP COUNTER 2 IN CONFIGURATION 3 ----
MD% = 10                                     ' MODE 10 IS SELECTED
DIO%(0) = 2                                  ' SELECT COUNTER 2
DIO%(1) = 3                                  ' SELECT CONFIGURATION 3
CALL DAS8(MD%, DIO%(0), FLAG%)
MD% = 11                ' MODE 11 IS SELECTED TO LOAD COUNTER WITH COUNT
DIO%(1) = 395
CALL DAS8(MD%, DIO%(0), FLAG%)
REM  STEP3--- SET UP COUNTER 1 IN CONFIGURATION 3 -----
MD% = 10                                     ' MODE 10 IS SELECTED
DIO%(0) = 1                                  ' SELECT COUNTER 1
DIO%(1) = 3                                  ' SELECT CONFIGURATION 3
CALL DAS8(MD%, DIO%(0), FLAG%)
MD% = 11                ' MODE 11 IS SELECTED TO LOAD COUNTER WITH COUNT
DIO%(1) = 10 * T                             ' COUNT FOR COUNTER 1
CALL DAS8(MD%, DIO%(0), FLAG%)
REM  STEP4--- SET UP COUNTER 0 IN CONFIGURATION 0 -----
MD% = 10                                     ' MODE 10 IS SELECTED
DIO%(0) = 0                                  ' SELECT COUNTER 1
DIO%(1) = 0                                  ' SELECT CONFIGURATION 0
CALL DAS8(MD%, DIO%(0), FLAG%)
MDL% = 11                 ' MODE 11 IS SELECTED TO LOAD COUNTER WITH COUNT
MD% = 13              ' MODE 13 IS SELECTED TO READ DIGITAL INPUTS IP1-IP3
DIO%(1) = 5 * T                              ' COUNT FOR COUNTER 0
REM STEP 5--- INITIALIZE CONTROL INPUTS AND GP-6 ----
OUT 788, 0:    OUT 789, 0                    ' INITIALIZE U1=0
OUT 792, 0:    OUT 793, 0                    ' INITIALIZE U2=0
OUT 785, 8                                   ' SET C3 HIGH --> GP-6 IC MODE
WAIT 786, 4, 4             ' WAIT FOR C2 TO HIGH ( i.e till ic mode is set)
OUT 784, 2                 ' SET MUX ADDRESS TO OPM 2
GOSUB lable1
GOTO lable2
lable1: REM ----  DATA SUB-ROUTINE  --------
DIM XE0(4), F(4, 4)
DIM L(4), G(4, 4), M(4, 4)
DIM XE1(4), PHI(4, 4), ZETA1E(4)
DIM ZETA1O(4), ZETA2E(4)
DIM ZETA2O(4), CE(4)
DIM CO(4), P(4, 4)
DIM XR(4)
DATA .0,0,.0,0
FOR I = 1 TO 4                                       'XEO
READ XE0(I)
NEXT I
DATA .1,0,.2,0
FOR I = 1 TO 4                                       'Xr
```

```
READ XR(I)
NEXT I
DATA 2.267406e1,9.336732e-1,4.316701,1.45069e-1
DATA -6.821325,-3.138055e-1,2.256269e1,9.431117e-1
DATA 2.009368e1,8.056389e-1,2.627241e1,1.0336508
DATA 7.904951e1,2.902429,7.904951e1,2.902429
FOR I = 0 TO 3                                     'F(0) TO F(3)
FOR J = 1 TO 4
READ F(I, J)
NEXT J
NEXT I
DATA 5.508251e-2,-.1817109,.4855799,1
FOR I = 0 TO 3
READ L(I)                                'L(0) TO L(3)
NEXT I
REM DATA FOR G(0) TO G(3)
DATA -3.782079,5.447524,3.782079,7.01368e-2
DATA 4.729583,-2.275324,-4.729583,2.90463
DATA -2.057808,-9.666447e-1,2.057808,2.808792
DATA 3.341534e-13,2.902429,-3.149877e-13,2.902429
FOR I = 0 TO 3
FOR J = 1 TO 4
READ G(I, J)                                      ' G(0) to G(3)
NEXT J
NEXT I
FOR I = 0 TO 3
FOR J = 1 TO 4
M(I, J) = F(I, J) - G(I, J)                      ' M(0) to M(3)
NEXT J
NEXT I
DATA 1,1.5768584E-1,9.2888806E-1,1.424716E-1
DATA -2.895292E-15,6.614367E-1,-7.216324E-1,6.74709E-1
DATA 1,9.278345E-1,6.038664E-2,9.346653E-1
DATA 3.031123E-15,-6.614367E-1,7.216324E-1,-6.598071E-1
FOR I = 1 TO 4
FOR J = 0 TO 3
READ P(I, J)                                     ' P(0) to P(3)
NEXT J
NEXT I
DATA 9.507361E-1,3.562492E-2,4.926391E-2,1.09168E-3
DATA -6.906648E-1,6.011309E-2,6.906648E-1,2.197191E-2
DATA 4.926391E-2,1.09168E-3,9.507361E-1,3.562492E-2
DATA 6.906648E-1,2.197191E-2,-6.906648E-1,6.011309E-2
FOR I = 1 TO 4                                        'PHI
FOR J = 1 TO 4
READ PHI(I, J)
NEXT J
NEXT I
DATA 1.248633E-2,1.781246E-1,1.703513E-4,5.4584E-3
FOR I = 1 TO 4                                      'ZETA1E
READ ZETA1E(I)
NEXT I
DATA 1.703513E-4,5.4584E-3,1.248633E-2,1.781246E-1
FOR I = 1 TO 4                                      'ZETA1O
READ ZETA1O(I)
NEXT I
DATA 1.703513E-4,5.4584E-3,1.248633E-2,1.781246E-1
FOR I = 1 TO 4                                      'ZETA2E
READ ZETA2E(I)
NEXT I
DATA 1.248633E-2,1.781246E-1,1.703513E-4,5.4584E-3
FOR I = 1 TO 4                                      'ZETA2O
READ ZETA2O(I)
NEXT I
DATA 1,0,0,0
FOR J = 1 TO 4                                      'CE
READ CE(J)
NEXT J
DATA 0,0,1,0
FOR J = 1 TO 4                                      'CO
```

```
 READ CO(J)
 NEXT J
 V0 = 0
 U0 = 0
 FOR I = 1 TO 4
 U0 = U0 - F(0, I) * XE0(I) + (F(0, I) - G(0, I)) * XR(I)
 NEXT I
 U0 = U0 - L(0) * V0
 RETURN
lable2: REM------ PERFORM CONTROL ROUTINE  ------
 INPUT "press enter to start ", S$
 OUT 785, 0                              ' SET GP-6 IN OP MODE
 WAIT 786, 4                             ' WAIT UNTIL OP MODE IS SET
lable3: N = 0
lable4: CALL DAS8(MDL%, DIO%(0), FLAG%)
 REM ----SELECTION OF ODD AND EVEN ZETA1,ZETA2,C -----
 IF (N / 2 = INT(N / 2)) THEN GOTO lable6 ELSE GOTO lable5
lable5: GOSUB lable11                    ' PREPARE DATA TO SEND TO DAC
 OUT 792, ULB                            ' RDAC LOW BYTE
 OUT 795, UHB                            ' RDAC high byte & START ADC
 GOSUB lable12                           ' SCALE DATA subroutine
 OUT 784, 2                              ' SET MUX ADDRESS TO OPM 2
 YE0 = 0                                 ' calculations for next step
 FOR I = 1 TO 4
 YE0 = YE0 + CO(I) * XE0(I)
 NEXT I
 FOR I = 1 TO 4
 XE1(I) = 0
 FOR J = 1 TO 4
 XE1(I) = XE1(I) + PHI(I, J) * XE0(J)
 NEXT J
 XE1(I) = XE1(I) + ZETA2O(I) * V0 + ZETA1O(I) * U0 + P(I, N) * (Y0 - YE0)
 NEXT I
 GOTO lable7                              ' next line for even choice
lable6: GOSUB lable11                     ' PREPARE DATA TO SEND TO DAC
 OUT 788, ULB
 OUT 791, UHB                             ' LDAC HIGH BYTE & start adc
 GOSUB lable12                            ' SCALE DATA SUB-ROUTINE
 OUT 784, 6                               ' SET MUX ADDRESS TO OPM 4
 YE0 = 0                                  ' calculations for next step
 FOR I = 1 TO 4
 YE0 = YE0 + CE(I) * XE0(I)
 NEXT I
 FOR I = 1 TO 4
 XE1(I) = 0
 FOR J = 1 TO 4
 XE1(I) = XE1(I) + PHI(I, J) * XE0(J)
 NEXT J
 XE1(I) = XE1(I) + ZETA2E(I) * V0 + ZETA1E(I) * U0 + P(I, N) * (Y0 - YE0)
 NEXT I
 GOTO lable7
lable7: REM -----  updation of values------
 V0 = U0                        ' FIRST V1=U0 THEN  V0=V1 i.e  V0=U0
 FOR I = 1 TO 4
 XE0(I) = XE1(I)
 NEXT I
 N = N + 1: IF N > 3 THEN N = 0
 U0 = 0
 FOR I = 1 TO 4
 U0 = U0 - F(N, I) * XE0(I) + M(N, I) * XR(I)
 NEXT I
 U0 = U0 - L(N) * V0
 REM ---- CHECK TIMING REQUIRED FOR CALCULATIONS -------
 CALL DAS8(MD%, IP%, FLAG%)
 IF IP% = 3 THEN PRINT "TOO SHORT ": GOTO lable4
lable8: CALL DAS8(MD%, IP%, FLAG%)           ' READ IP AGAIN
 IF IP% = 3 GOTO lable4
 IF IP% = 1 GOTO lable9
 GOTO lable8
lable9: CALL DAS8(MD%, IP%, FLAG%)           ' READ IP AGAIN
```

```
 IF IP% = 1 GOTO lable9
 IF N < 4 GOTO lable4                              ' FINALLY IP%=3 IS OK
 GOTO lable3
lable10: GOSUB lable13
 END
lable11: REM ----- PREPARE DATA TO SEND  (DAC) -------
 UK = (U0 * 2048) / 10                             ' FORM BINARY DATA WORD
 IF UK > 2047 THEN UK = 2047
 IF UK < -2048 THEN UK = -2048
 IF UK < 0 THEN UK = 4096 + UK                     ' FOR 2'COMPLIMENT WORD
 UHB = INT(UK / 16)                                ' SERERATE HIGH BYTE
 ULB = (UK - UHB * 16) * 16                        ' SEPERATE LOW BYTE
 IF ULB > 255 THEN ULB = 255
 RETURN
lable12: REM -------SUBROUTINE  SCALE DATA --------
 QA = INP(787) * 16                                ' GET DATA H.BYTE & SCALE
 QB = INP(786) / 16                                ' GET DATA L.BYTE & SCALE
 AA = QA + INT(QB)                              ' DROP SENSE BITS & COMBINE H&L
 IF AA > 2047 THEN AA = AA - 4096                  ' SCALE NEGATIVE DATA
 Y0 = (AA / 2048) * 10                             ' SCALE TO 10 V REF
 RETURN
lable13: REM ------ PUT GP-6 IN IC MODE AT THE END -------
 OUT 785, 8                                  ' SET C3 HIGH FOR GP-6 IC MODE
 WAIT 786, 4, 4                                    ' WAIT TILL IC MODE IS SET
 RETURN
```

REFERENCES

[1] Ogata Katsuhito, Discrete-Time Control Systems, New Jersey: Prentice-Hall, Inc., 1987.

[2] Hostetter Gene H., Digital Control System Design, New York: Holt, Rinehart and Winston, Inc., 1988.

[3] Friedland Bernard, Control System Design (An Introduction to State-Space Methods), New York:McGraw-Hill, Inc., 1986.

[4] Hass Violet B., Analog and Digital Computer Handbook, Indiana: College Town Press, 1985.

[5] Chungcharoen Ekachidd, A Multiple Sampled-Data Controller with Multiplexed Inputs and Outputs, Masters Thesis, Electrical Engineering Department, Youngstown State University, 1990.

[6] Model 767 Analog/Digital Positional Control Panel Operators and Maintenance Manual, Illinois: Comdyna, Inc.

[7] DAS-8 and DAS-8PGA User's Manual, Taunton, MA: MetraByte Corporation, 1988.

[8] Thompson Peter M., Program CC Version 4 Reference Manual, Hawthorne, CA: Systems Technology, Inc., 1989.

[9] Lien David A., The Basic Handbook (Encyclopedia of the BASIC Computer Language), San Diego, CA: Compusoft Publishing, 1986.

[10] Microsoft QuickBASIC (Learning to use Microsoft QuickBASIC) version 4.5, Microsoft Corporation, 1990.

[11] Microsoft Quick C (C for yourself) Version 2.0, Microsoft Corporation, 1988.

[12] Lafore Robert, The Waite Group's Microsoft C (Programming for IBM), Indiana: Howard W. Sams and Company, 1988.

[13] Tomkins Willis J., Webster John G., Interfacing Sensors to the IBM PC, New Jersey: Prentice-Hall, 1988.

[14] Eggebrecht Lewis C., Interfacing to the IBM Personal Computer, Indianapolis, IN: Howard W. Sams and Company, Inc., 1983.