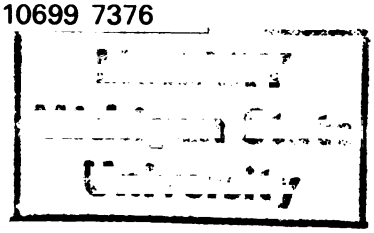


This is to certify that the

dissertation entitled

A MODIFIED LUENBERGER OBSERVER FOR THE INVERTED PENDULUM

presented by

Steve C. Southward

has been accepted towards fulfillment of the requirements for

Master of Science degree in Mechanical Engineering

Clark Radcliffe
Major professor

Date Jan 14, 1986

A MODIFIED LUENBERGER OBSERVER FOR THE INVERTED PENDULUM

By

Steve C. Southward

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirement
for the degree of

MASTER OF SCIENCE

Department of Mechanical Engineering

1985

ABSTRACT

A MODIFIED LUENBERGER OBSERVER FOR THE INVERTED PENDULUM

By

Steve C. Southward

The inverted pendulum, aside from being a fascinating classical control problem, has primary applications in the aerospace industry. This inherently unstable system has geometric as well as discontinuous coulomb nonlinearities. Presented here is an attempt to apply linear control and observation theories, with modifications that include nonlinear coulomb forces, to the real system. The linear plant is shown to admit reductions which allow full state observation through cart position only. Computer simulation and actual system demonstration verify that stability is highly sensitive to the accuracy of the nonlinear damping characterization.

DEDICATION

To my grandmother, Laura Hoffman

ACKNOWLEDGEMENT

I would like to express my most sincere appreciation and gratitude to Dr. Clark Radcliffe for providing me with the benefit of his vast experience in real time control. His knowledge and friendship have made this research an enjoyable part of my academic experience.

I would also like to extend many thanks to Dr. Suhada Jayasuriya, Dr. Steve Shaw, and Yossi Chait for their expertise and aid throughout my masters study.

Special thanks to Jeff Hopwood for his friendship and his invaluable electrical engineering assistance.

Finally, I would like to thank my parents Charles and Mildred, and my brothers David and Bob, for their constant love and support during my academic career.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

$\underline{A}_s$ = (5x5) linearized system plant matrix

$\underline{A}$ = (4x4) reduced order plant matrix

$\underline{B}_s$ = (5x1) input vector

$\underline{B}$ = (4x1) reduced order input vector

b_1 = equivalent angular viscous damping coeff. (pend.)

b_2 = equivalent linear viscous damping coeff. (cart)

b_3 = equivalent viscous damping coeff. (motor)

$\underline{C}_s$ = (5x2) coulomb damping matrix

$\underline{C}$ = (4x2) reduced order coulomb damping matrix

$\underline{D}_s$ = (2x5) output matrix

$\underline{D}$ = (2x4) reduced order output matrix

d = equivalent pendulum length

$\underline{F}$ = (5x1) coeff. matrix for LSF gain solution

F = force applied to the cart

f_x = coulomb damping force in cart motion

f_θ = coulomb damping force in pend. motion

$\underline{G}$ = (4x2) matrix of observer gains

g = gravitational constant

$\underline{H}$ = (5x5) coeff. matrix for LSF gain solution

i = DC servo motor armature current

J = equivalent DC servo motor armature inertia

$\underline{K}_s$ = (1x5) matrix of LSF control gains

$\underline{K}$ = (1x4) reduced order matrix of LSF control gains

k_τ = torque constant for DC servo motor

k_E = electrical constant for DC servo motor

L = DC servo motor armature inductance

M = equivalent cart mass

m = equivalent pendulum mass

n = transformation const. from armature to cart motion

Q_j = net nonconservative force associated with q_j

q_j = generalized coordinate for each degree of freedom

R = equivalent DC servo motor armature resistance

$\underline{S}$ = (4x4) coeff. matrix for OBS gain solution

$\underline{T}$ = (4x1) coeff. matrix for OBS gain solution

T = scalar kinetic energy function

$\underline{U}$ = (1x1) input matrix

U = scalar potential energy function

V = actual voltage applied to the DC servo motor

v = cart velocity

$\underline{X}_s$ = (5x1) state vector: $[x,v,\theta,\omega,i]^T$

$\underline{X}$ = (4x1) reduced order state vector: $[x,v,\theta,\omega]^T$

$\hat{\underline{X}}$ = (4x1) observed state vector

$\underline{X}'$ = (2x1) velocity state vector: $[v,\omega]^T$

x = cart position

$\underline{Y}$ = (2x1) output state vector: $[x,\theta]^T$

$\hat{\underline{Y}}$ = (2x1) observed output state vector

θ = pendulum position

ω = pendulum angular velocity

ξ_i = coefficient of characteristic polynomial

INTRODUCTION

The inverted pendulum is one of the more fascinating classical control problems. The pendulum pivot is fixed to a cart, allowing the pendulum to freely rotate with respect to the cart. When the pendulum pivot is above its center of gravity, the system is at a stable equilibrium position. Inverting the pendulum, such that the center of gravity is above the axis of rotation, places the system in an unstable state. However, a control force can be applied to the cart to keep the pendulum in this inverted position.

Stabilization of the inverted pendulum involves determining the proper control force. In its simplest form, this classic problem has proven to be an excellent example problem for introductory stability and control study [10,14]. In its most complex form, this basic problem has many interesting and important practical applications. Probably the most notable application is in the aerospace industry. A similar control system is employed for attitude control of a space booster on take off [14]. Immediately prior to take off, the space booster is balanced in an upright position by the launch platform. It is free to fall over, just as an inverted pendulum will, unless some control force is applied at the base. Other practical applications that involve balancing mechanical systems in an unstable vertical position include missile guidance systems [3] and mobile robotics [7].

A laboratory inverted pendulum system was studied to test possible control schemes. The 15 inch pendulum, hinged to the cart, allowed only a planar motion. The 4 inch long cart was constrained, via two rails, to a linear motion of about three feet. The cart and pendulum positions provide two degrees of freedom. A flexible toothed drive belt is

connected to each end of the cart, through pulleys at each end of the rails, forming a continuous loop. One pulley shaft is driven by a 12 volt DC servo motor through a gear and chain connection. Control forces can be applied to the cart by applying a voltage to the DC motor.

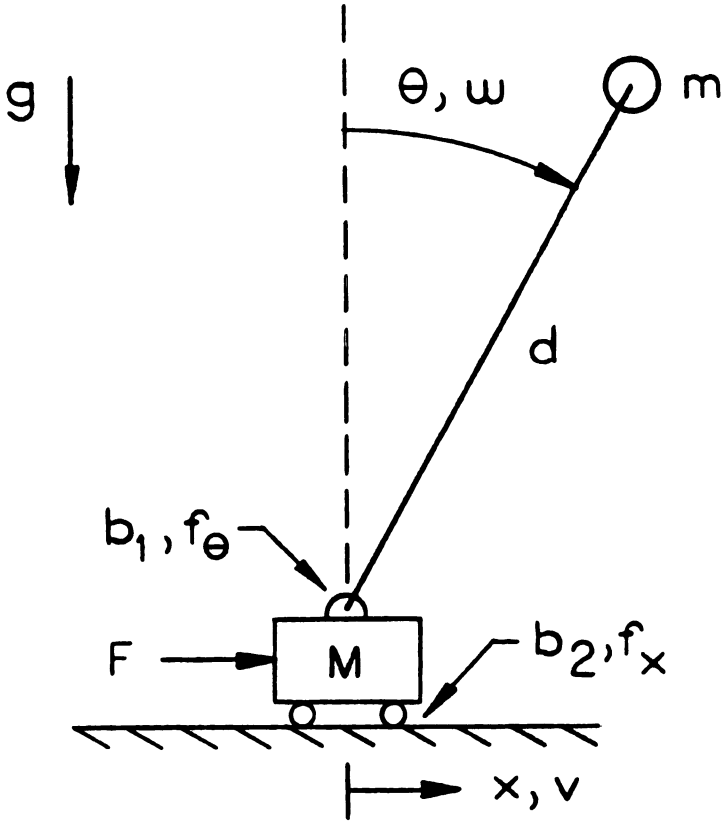


Figure 1: The inverted pendulum system

The complete system, made up of the cart, pendulum, and motor, is modelled here by a fifth order model. The cart/pendulum subsystem, shown in Figure 1, has four coupled first order nonlinear equations of motion with a force input. Primary nonlinearities in these equations are the transcendental functions of the pendulum angle terms. The real system also exhibits coulomb damping primarily associated with the cart

motion, and this discontinuous nonlinearity is included in the model. The DC servo motor is modelled by a second order linear system. Due to the assumed rigid coupling between the motor and the cart, the motor armature velocity is dependent on the cart velocity. There are only five independent states of motion, thus the system is fifth order.

A linear model is required to develop control and observation algorithms. Using small angle approximations, and neglecting coulomb effects, the system model can be linearized about the unstable vertical position. It is this linear model which is studied in classical inverted pendulum control problems. Using the experimental system parameters in the linear model, the open loop poles are:

$$\{-2801.7,\ -7.42,\ -3.7,\ 0.0,\ 5.4\}$$

This linear system has an unstable pole from the pendulum, and a rigid body pole from the cart. The large stable pole is from the DC servo motor, indicating that the motor dynamics are much faster than the cart/pendulum system dynamics.

The nonlinear system can be controlled with Linear State Feedback, designed to stabilize the linearized system. Theoretical simulations, as well as actual experimental demonstration, verify this result. A linear state feedback law produces a control voltage, in this case, which is a sum of proportional contributions from each of the five system states. Linear control theory provides an algorithm for determining the control gains to place the closed loop poles in desired locations.

Implementation of such a control law requires state estimation, or observation. The real experimental system readily provides two of the five required states. Direct measurements are made for the cart position (x) and the pendulum position (θ). A simple reduction allows the DC motor state to be eliminated from the control law. Linear Luenberger observer theory can then be used to design an observer which will provide the remaining states for the LSF control law. This linear observer must be modified for implementation on the real system, where the coulomb forces will act as disturbance inputs.

PLANT MODEL

The nonlinear equations of motion for the cart and pendulum were obtained through application of Lagranges equation, which is based on the interaction between kinetic and potential energy in the system, and the nonconservative forces present [12]. These equations contain geometric and discontinuous nonlinearities (Appendix A):

$$(M+m)\ddot{x} + md\ddot{\theta}\cos(\theta) - md\omega^2\sin(\theta) + b_2 v + f_x \mathrm{sgn}(v) = F \qquad (1)$$

$$md\ddot{x}\cos(\theta) + md^2\ddot{\theta} - mgd\sin(\theta) + b_1 \omega + f_\theta \mathrm{sgn}(\omega) = 0 \qquad (2)$$

The nonlinear cart/pendulum subsystem model has a force input (F), which is supplied by the DC servo motor through the gear train and drive belt assembly. These two coupled second order equations can be rewritten as four first order equations, one for each of the cart/pendulum states.

The DC servo motor can be modelled by a second order linear model with a rigid coupling between the cart and the motor armature. Because this coupling produces a dependent state variable, the second order model reduces to a first order model plus a constraint equation given by:

$$F = (k_\tau n)\, i - (n^2 J)\, \ddot{x} - (n^2 b_2)\, v \qquad (3)$$

$$L\,(di/dt) = V - i R - (k_E n)\, v \qquad (4)$$

The only nonlinearity in the motor model is the saturation of the input voltage. The power supply used to drive the laboratory system has a maximum output given by:

$$|V| < 15.8 \text{ volts} \qquad (5)$$

Equations (1) through (5) represent the complete nonlinear plant model.

This plant describes the system dynamics, and will provide a basis for the control and observation designs.

A linear model must be obtained from this nonlinear plant. The geometric nonlinearities can be removed by small angle approximations. Normally, coulomb damping terms would drop out of a linearized model. They will be retained here, producing a hybrid linear model with a nonlinear part. The complete hybrid model can be represented in bond graph notation as:

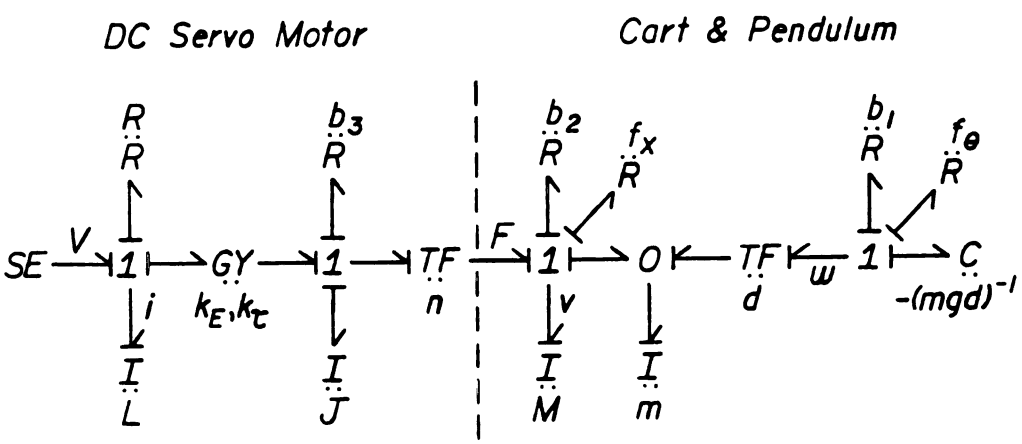


Figure 2: Bond graph for the hybrid linearized plant model

This hybrid linear plant has the matrix representation:

$$\dot{\underline{X}}_s = \underline{A}_s \underline{X}_s + \underline{B}_s \underline{U} + \underline{C}_s \, \mathrm{sgn}(\underline{X}') \tag{6}$$

$$\underline{Y} = \underline{D}_s \underline{X}_s \tag{7}$$

where the linear matrices are defined in Appendix B, along with the complete linearization procedure.

LINEAR STATE FEEDBACK CONTROL

Some control law is needed to stabilize the nonlinear system. A Linear State Feedback (LSF) law of the form:

$$\underline{U} = V = \underline{K}_s \underline{X}_s \qquad (8)$$

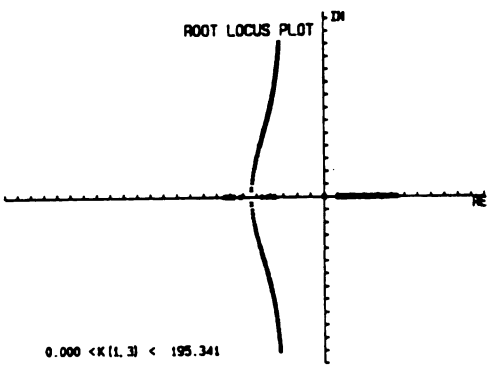
has been designed for the linearized system, and its effectiveness for implementation on the real nonlinear system investigated. The LSF control law must at least push the single unstable open loop pole, and the rigid body pole of the linear system, into the left half plane.

In order to implement the relatively simple LSF control law, the four position and velocitiy states of the cart/pendulum, and the armature current in the DC motor must either be measured or calculated. Five feedback gains must be determined, one for each of the system states. Only a zero valued control gain would eliminate the need to observe a particular state.

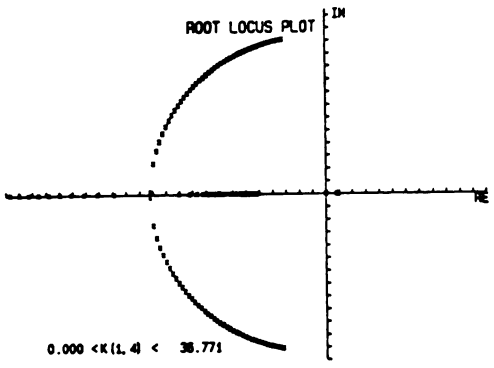
A root locus study was performed to investigate the possibility of eliminating one of the states by zeroing a gain, thus allowing partial state feedback. The LOCUS program (Appendix E) was developed whereby the LSF gains could be interactively tuned to visually see the effect on the linear system poles.

A set of partial state feedback gains were found which placed all poles in the left half plane. Figure 3 shows the root locus as the gains were taken from zero values one by one to their nominal values placing the poles at $\{-4\pm2j,-5\pm2j\}$. On the first branch of the root locus, the pendulum position gain (K3) was varied from 0 to 195.34. The pendulum velocity gain (K4) was then varied from 0 to 36.77. For the third branch, the cart position gain (K1) was varied from 0 to 19.1.

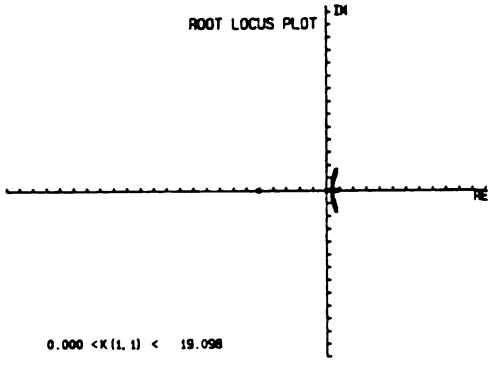
The final branch was obtained by varying the cart velocity gain (K2) from 0 to 19.14.



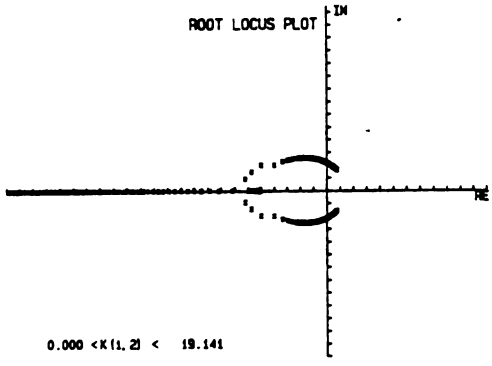
First Branch



Second Branch



Third Branch



Fourth Branch

Figure 3: Inverted pendulum root locus

A conventional controllability test [10], through construction of the controllability matrix, does not provide any information about the possibility of partial state feedback. The root locus study has demonstrated the possibility of stability with a zero current gain.

An algorithm was developed to determine the partial state feedback gains, along with the condition allowing a zero current gain, for a desired set of closed loop poles. The problem was to choose the matrix

$\underline{K}_s$ such that the poles of $[\underline{A}-\underline{B}\underline{K}]_s$ were in the left half plane, where $\underline{A}$ and $\underline{B}$ are the fifth order linear system matrices. Writing out the characteristic polynomial of $[s\underline{I}-(\underline{A}-\underline{B}\underline{K})]_s$ in closed form provided expressions for the coefficients in terms of the system parameters, and the unknown control gains. The next step was to explicitly solve for the five coefficients of the characteristic polynomial in terms of the desired closed loop poles.

Equating the two solutions for the coefficients produces five equations in the five unknown gains of $\underline{K}_s$. Since these equations are linear in the control gains, they can be rearranged into the form:

$$\underline{H}\,\underline{K}_s^T = \underline{F} \qquad (9)$$

where the $\underline{H}$ and $\underline{F}$ matrices (Appendix C) contain the coefficient expressions of the system parameters, and the closed loop poles. A unique solution exists, given a real set of system parameters, since the plant has a single input and is completely controllable.

A constraint must be placed on the matrix system (9) in order to force (K5) to have a zero value. Since the system parameters cannot be changed, the choice of closed loop poles must be restricted.

A valid solution to (9) is one which satisfies all five linear equations. One of these equations contains the single control gain (K5). The LHS of (10) is a function of the desired closed loop poles, and the RHS is a function of the system parameters:

$$\xi_4 = P1(K5) + (A1+A7+A10) \qquad (10)$$

Setting (K5) to zero provides the constraint relation between the actual system parameters and the closed loop poles. Only four of the five

closed loop poles can be chosen independently. The fifth pole will be restricted by (10).

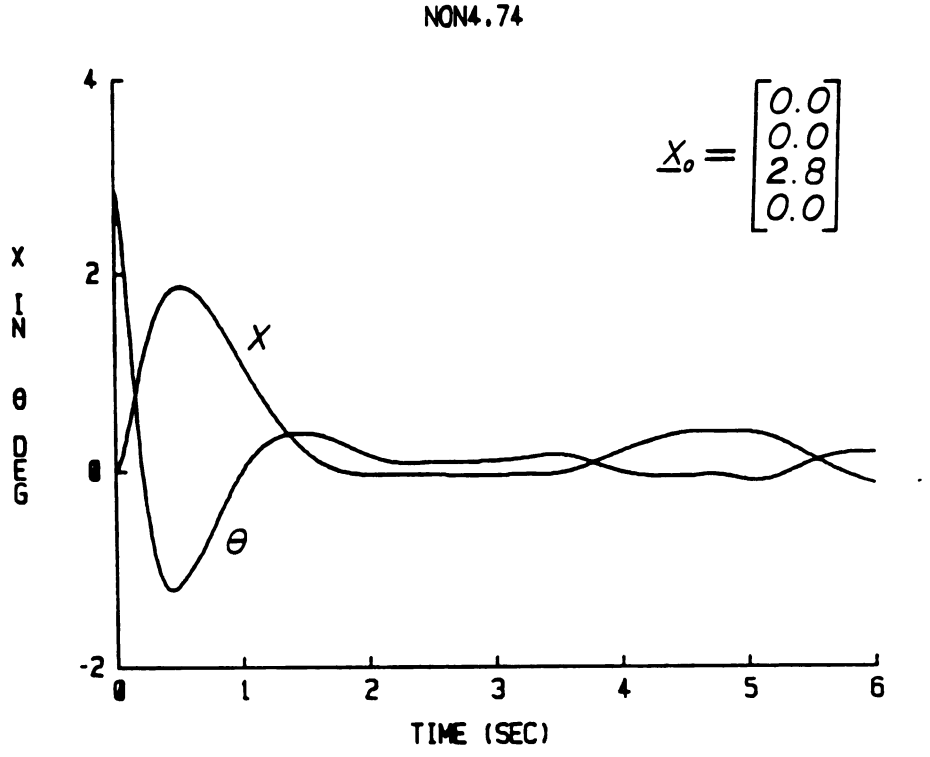


Figure 4: Nonlinear system with LSF control law

The effectiveness of the LSF control law applied to the nonlinear system is demonstrated through a DIFFEQ computer simulation (Figure 4). Since the dynamics are being simulated through a mathematical model, all the system states are readily available for a control law.

LUENBERGER OBSERVER DESIGN

The linear fifth order matrix system model can be reduced to an approximate fourth order model (Appendix K). It has already been shown that the system model is controllable with partial state feedback. Based on this, and the fact that the motor dynamics are much faster than the cart/pendulum system dynamics, as seen in the open loop poles, a simplification is possible [4,9]. The reduced fourth order model is given by:

$$\frac{d}{dt}\begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -A11 & -A2 & A3 \\ 0 & 0 & 0 & 1 \\ 0 & A12 & A6 & -A7 \end{bmatrix} \begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ A13 \\ 0 \\ -A14 \end{bmatrix} V + \begin{bmatrix} 0 & 0 \\ -C1 & C2 \\ 0 & 0 \\ C3 & -C4 \end{bmatrix} \begin{bmatrix} sgn(v) \\ sgn(\omega) \end{bmatrix}$$

where the matrix elements are defined in Appendices B and K.

This reduction of order has several advantages. Since the current state is not needed for stabilization of the system, an observer will not have to estimate this state. The observer only needs to be fourth order, thus reducing the number of computations necessary for implementation. The observer design will be based on this reduced order model.

For full state observation, and neglecting the coulomb damping terms, a linear Luenberger Observer can be designed having the form:

$$\dot{\hat{\underline{X}}} = \underline{A}\,\hat{\underline{X}} + \underline{B}\,\underline{U} + \underline{G}\,(\underline{Y} - \hat{\underline{Y}}) \qquad (11)$$

$$\hat{\underline{Y}} = \underline{D}\,\hat{\underline{X}} \qquad (12)$$

where $\underline{A}$ and $\underline{B}$ are the system matrices given above, $\underline{X}$ is the reduced state vector, and $\underline{Y}$ is the system output vector. The matrix of unknown observer gains $\underline{G}$ is given by:

$$\underline{G} = \begin{bmatrix} G1 & G5 \\ G2 & G6 \\ G3 & G7 \\ G4 & G8 \end{bmatrix} \tag{13}$$

The solution procedure for placement of the observer poles follows along the same lines as for the LSF control gains except that the observer is a multiple input system. The Luenberger observer has both the actual cart and pendulum positions as inputs. The Luenberger observer design procedure involves solving for the observer gains in $\underline{G}$ such that the poles of $[\underline{A}-\underline{GD}]$ are the desired observer poles, which should be to the left of the desired closed loop system poles [10].

Multiple input systems require solving an underdetermined system of equations. Pole placement, in this particular problem, will only provide four equations, with eight unknown observer gains to solve for. An extra design constraint must be imposed on the system in order to obtain a unique solution. Typical imposed design constraints are norm minimization of the gain matrix $\underline{G}$ [11], and eigenvector placement [13].

The extra design constraint chosen here is the elimination of the pendulum position sensor. Observability tests (Appendix G) indicate that the system is observable with only the cart position as an input. It is not observable with only the pendulum position input. This is intuitively correct since the dynamics of the pendulum are not a function of the global location of the cart, whereas the cart dynamics are dependent on the pendulum position. Knowing the cart dynamics, the pendulum states may be estimated.

Following the same design procedure for pole placement, utilizing the design constraint:

$$G5 = G6 = G7 = G8 = 0 \tag{14}$$

the solution for the remaining observer gains for a given system and desired set of observer poles, reduces to solving a linear system of equations given by:

$$\underline{S}\ \mathrm{col}_1(\underline{G}) = \underline{T} \tag{15}$$

where the elements of $\underline{S}$ and $\underline{T}$ are functions of the desired observer poles and the reduced order system parameters (Appendix D).

At this point, linear Luenberger observer theory has been used to design an observer whose states will converge to the linear plant states. The coulomb damping which exists in the actual system is not included in this observer. Some observation theories, such as Kalman filtering [10], allow coulomb forces as nonlinear disturbances, but do not readily allow the elimination of one position sensor.

A modified Luenberger observer can be designed which includes coulomb forces as disturbance inputs. Using the design constraint (14), the observer (11) has a single input, which is the cart position x. The modified observer will require two additional inputs from the actual system, and is given by:

$$\dot{\hat{\underline{X}}} = \underline{A}\,\hat{\underline{X}} + \underline{B}\,\underline{U} + \underline{G}\,(\underline{Y} - \hat{\underline{Y}}) + \underline{C}\ \mathrm{sgn}(\underline{X}') \tag{16}$$

The presence of coulomb damping terms in the modified observer will have no effect on the convergence of the observed states to the linear system states, since the coulomb damping terms are also present in the linear model. The two actual states in $\underline{X}'$, $(\mathrm{sgn}(v), \mathrm{sgn}(\omega))$, are also not readily available. Using the signs of the observed velocities,

$(\mathrm{sgn}(\hat{v}), \mathrm{sgn}(\hat{\omega}))$, this modified observer can be stabilized under certain strict conditions.

This modified Luenberger observer was tested on the linear and nonlinear system models in a DIFFEQ simulation (Appendix H). The hybrid linear plant, as well as the nonlinear plant were stabilized with this observer and linear state feedback.

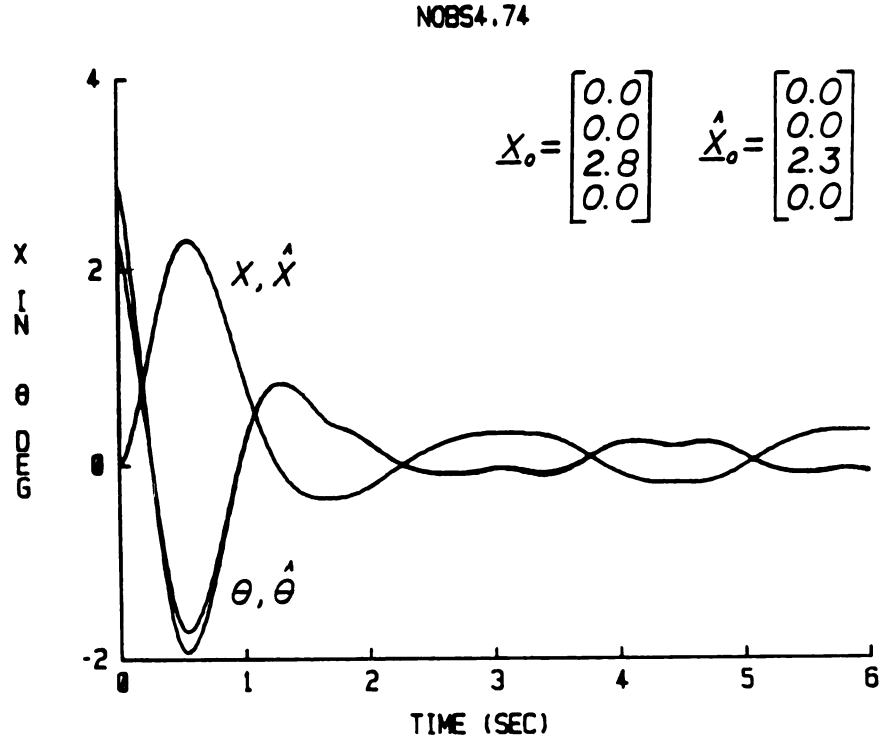


Figure 5: Nonlinear plant with modified Luenberger observer

System stability is achieved under strict conditions, determined through the DIFFEQ simulations. First, all parameters of the observer must be very accurate in describing the system under observation. Convergence of the observer states, with coulomb damping present, is very sensitive to inaccuracies in the observer model. The second

condition for convergence is that all initial conditions for the observer must be very close to the actual initial system states.

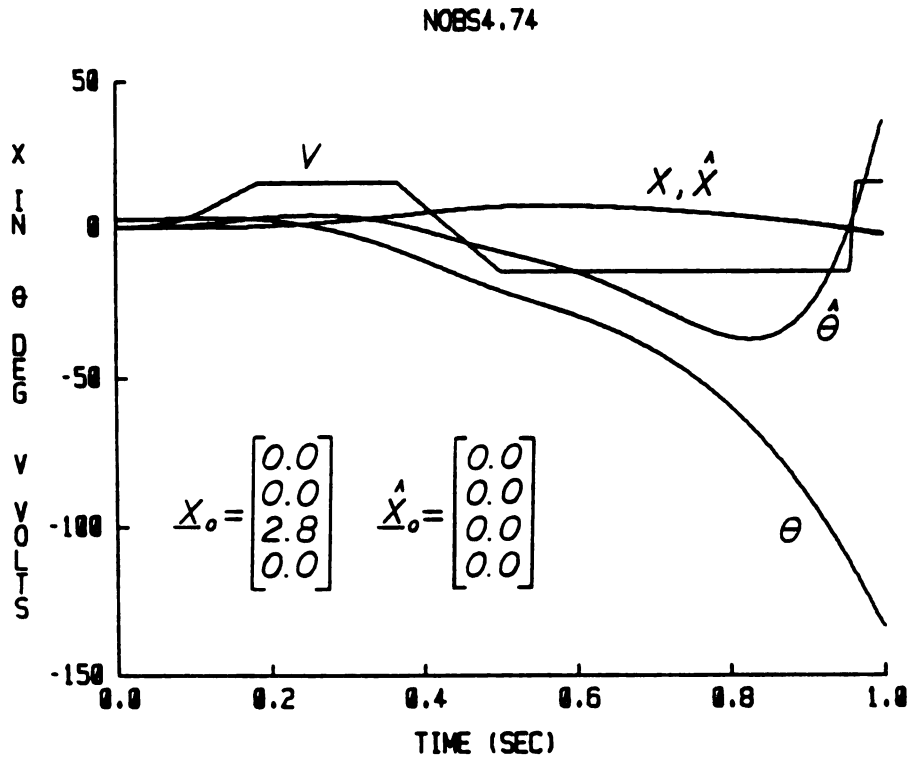


Figure 6: Nonlinear system with convergence conditions not met

The nonlinear plant is unstable with the modified Luenberger observer and the LSF control law, if any of these convergence conditions are not met. Figure 6 shows the nonlinear system response for a nonzero initial tracking error.

EXPERIMENTAL RESULTS

An existing laboratory inverted pendulum setup was used to test the theoretical control algorithms. The primary test stand included the cart and pendulum subsystem connected through a drive belt/gear train to the DC servo motor (Figure 7). A DEC LSI-11/23+ digital computer performed all real time data acquisition and control processing. Other hardware included two power supplies and a power amplifier to interface between the computer and the DC servo.

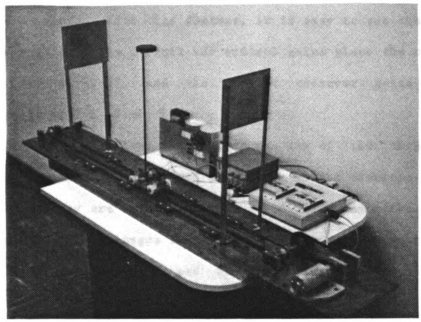


Figure 7: The experimental inverted pendulum setup

Control software was designed to run under all standard DEC single-user RT-11 operating systems [5]. The program is menu driven to facilitate the tasks required for testing and demonstration. Complete source listings and documentation can be found in Appendix J.

There are four major tasks available to the user. The user must first define the origin of the coordinate system referenced by the computer. This task is important since the control algorithms are designed to drive the system to the coordinate system origin, where all states are zero. Another option allows the status of these user-defined zero positions to be checked.

There are many gains and parameters associated with the various control algorithms implemented in the program. Initially, these gains all have default values. The gains option allows the user to check or change these values. With this feature, it is easy to see the effect of changing any gains. The default LSF control gains place the closed-loop poles at {-3,-4,-5,-6}, and the default observer gains place the observer poles at {-8,-8,-9,-9}.

The final option allows the user to run any of the three control algorithms available. All three algorithms have a discrete nature, in the sense that they are implemented on a digital computer. In the general algorithm, voltages from the two position sensors are read and digitized. Using these digitized states, the remaining states are computed. Once all the states are known, a control voltage is computed through the LSF law. The control voltage is then applied to the motor. This process repeats at a user specified rate, or sampling frequency.

Of the observation schemes available, the easiest to implement uses finite difference derivative approximations to calculate the velocity states. Knowing the present and previous positions, velocities can be computed with respect to the sampling period. Increasing the sampling rate increases the accuracy of the simple derivative approximations. A maximum sampling rate of 190 Hz. is possible due to the small number of

floating point operations performed in this algorithm. One advantage of this method is that control gains can be designed in the continuous domain, rather than in a discrete domain. The finite difference approximation with the LSF control law stabilizes the actual nonlinear system, as shown in Figure 8.

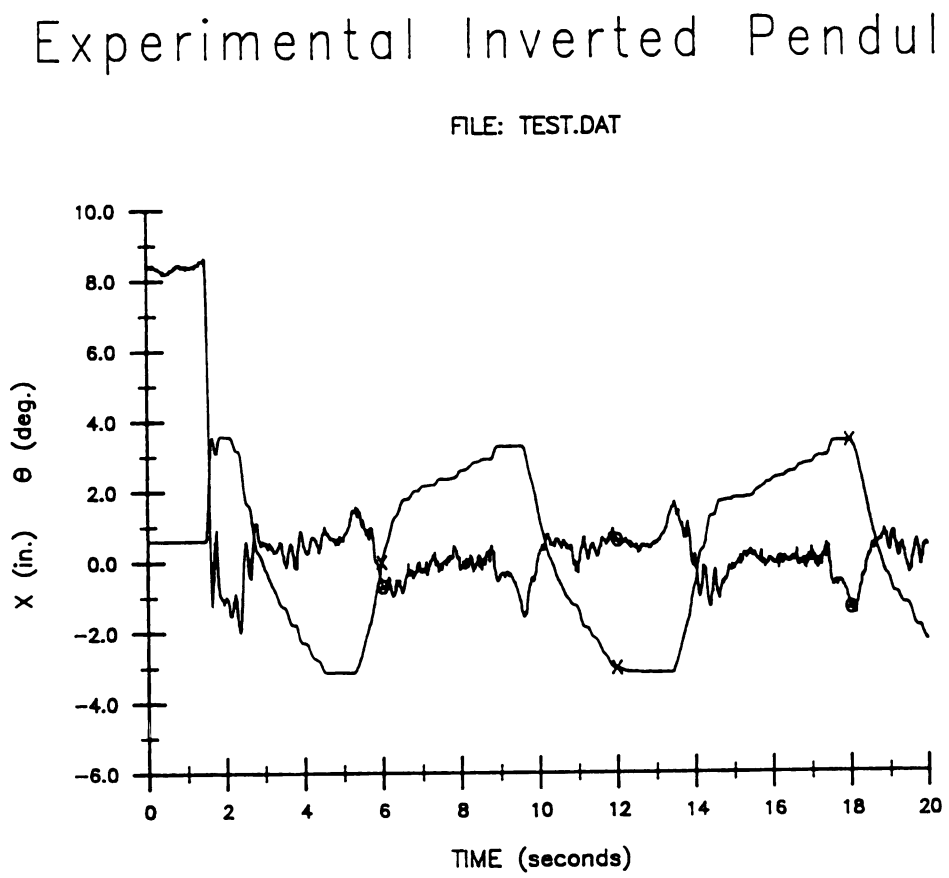


Figure 8: Actual system output under LSF control

The remaining two observation schemes use the modified Luenberger observer to calculate the unknown states. The first scheme uses the continuous observer system, which is a set of four first order differential equations. These equations are integrated by an Euler approximation in real time, and the algorithm is able to run at a

maximum 175 Hz. The second observer is a discretization of the continuous design. It also runs at a maximum 175 Hz. Neither of these control algorithms stabilize the system.

From nonlinear system model simulations, the observer must have initial conditions which are very close to actual initial states in order to track the real system. This problem is easily solved in the actual implementation. The first observation scheme (finite difference) has been shown to stabilize the real system. Upon choosing either Luenberger observer scheme, this first algorithm is turned on. Once this algorithm has stabilized the system, the user may switch over to the Luenberger observer algorithm at the touch of a key. Both observers use the final states calculated from the finite difference approximations for the initial conditions.

CONCLUSIONS

Through computer simulation results, the nonlinear system model can be stabilized with linear control and the modified Luenberger observer, under certain strict conditions. All parameters must be well known, and the initial tracking error of the observer must be near zero. Stability of the simulated model is demonstrated under these conditions. Computer simulations also indicate that system stability is more sensitive to inaccuracies in the viscous and coulomb damping terms.

The results of a parameter study on cart damping (Appendix I) indicate an inadequacy of the assumed damping mechanisms present. Data plotted from the hysteresis loop experiments would have been linear if the assumed linear combination of viscous and coulomb damping forces were correct. Based on the scatter of data points, the actual damping present in the cart motion is more complex than originally assumed.

The nonlinear model, used in the computer simulations, and as a basis for the control and observer designs, is not an exact representation of the actual laboratory system. Since system stability, under linear observation and control, is sensitive to actual parameter values, stability of the real system will not be attainable utilizing the modified observer design and LSF.

The real system response under linear observation is similar to the response of a simulated system whose observer has erroneous parameters. Though the actual system response cannot be directly compared to the

nonlinear model response, the similar outputs indicate that the problem lies in the nonlinear model representation of the actual system. A more accurate representation of the damping mechanisms in the cart motion are needed for the nonlinear model. Then, based on this model, a new observation scheme can be developed.

This investigation, theoretical and experimental, has raised a very important and interesting observation. Linear multiple input systems, where pole placement is desired, define an underdetermined system of equations. This is true for linear state feedback control and for linear Luenberger observer designs. There are an infinity of valid control gain solutions to place the poles at desired locations. If these linear control or observation theories are to be applied to a nonlinear system, some solutions may be better than others. Given several sets of gains which all place the same set of closed loop poles, some may provide stability for the nonlinear system, and some may not.

The finite difference approximation, to compute the velocity states, has proven to be the most effective means of stabilizing the inverted pendulum with linear state feedback control. This combined observation and control scheme appears to be insensitive to the nonlinear coulomb damping disturbances. Though the effects of the coulomb forces can be seen in the actual system response, the pendulum remains in a stable limit cycle.

REFERENCES

REFERENCES

1. Beckwith, T., Buck, N., and Marangoni, R., Mechanical Measurements, Third Ed., Addison-Wesley, 1982.

2. Case Center for Computer-Aided Design, Case Center User's Guide, College of Engineering, Michigan State University, 1984.

3. Chobotov, V., "Dynamic Stability of a Pendulous Missile Suspension System," ASME Journal of Applied Mechanics, Vol. 29, pp. 276-282, June, 1962.

4. Davison, E., "A Method for Simplifying Linear Dynamic Systems," IEEE Transactions on Automatic Control, Vol. AC-11, pp. 93-101, 1966.

5. Eckhouse, R., and Morris, L., Minicomputer Systems, Second Ed., Prentice-Hall, 1979.

6. Electro-Craft Corporation, DC Motors, Speed Controls, Servo Systems, Fifth Ed., August, 1980.

7. Hemami, H., Weimer, F., and Koozekanani, S., "Some Aspects of the Inverted Pendulum Problem for Modeling of Locomotion Systems," IEEE Transactions on Automatic Control, Vol. AC-18, pp. 658-661, Dec. 1973.

8. Karnopp, D., and Rosenberg, R., Introduction to Physical System Dynamics, McGraw-Hill, 1983.

9. Kokotovic, P., O'Malley, R., and Sannuti, P., "Singular Perturbations and Order Reduction in Control Theory - An Overview," Automatica, Vol. 12, pp. 123-132, 1976.

10. Kwakernaak, H., and Sivan, R., Linear Optimal Control Systems, John Wiley and Sons, 1972.

11. Lee, G., and Jordan, D., "Pole Placement with Feedback Gain Constraints," IEEE 1975 Decision and Control Conference, pp. 188-190, Dec. 1975.

12. Mierovitch, L., Analytical Methods in Vibrations, The Macmillan Company, 1967.

13. Moore, B.C., "On the Flexibility Offered by State Feedback in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment," IEEE Transactions on Automatic Control, Vol. AC-21, Oct. 1976, pp. 689-692.

14. Ogata, K., Modern Control Engineering, Prentice-Hall, 1970.

APPENDICES

APPENDIX A

The complete inverted pendulum system consists of two primary coupled subsystems which are easily identified. The cart and pendulum make up the first subsystem. It is nonlinear and completely mechanical in nature. The second subsystem is a DC servo motor, and will be modelled here as a linear system. The servo motor provides a coupling between electrical and mechanical energy. Due to the intrinsic differences between these two subsystems, the equations of motion for each can be derived separately, and then coupled together in the proper fashion to produce a complete set of system equations.

Lagranges method is invoked to determine the system equations for the cart/pendulum system. This method is well suited for the geometric and discontinuous nonlinearities present in the system [12]. Lagranges equation including non-conservative forces is given by:

$$\frac{d}{dt}\left[\frac{\partial T}{\partial \dot{q}_j}\right] - \frac{\partial T}{\partial q_j} + \frac{\partial U}{\partial q_j} = Q_j \quad (j = 1,2) \tag{A1}$$

Placing the datum line at the axis of the pendulum, and noting that the potential energy of the cart is invariant, the potential energy function describing the cart/pendulum subsystem is:

$$U = (m\ g\ d)\cos(\theta) \tag{A2}$$

The kinetic energy of this subsystem is due to the velocity magnitudes of the pendulum and cart masses.

$$T = \frac{1}{2} M v^2 + \frac{1}{2} m v_m^2 \qquad \text{(A3)}$$

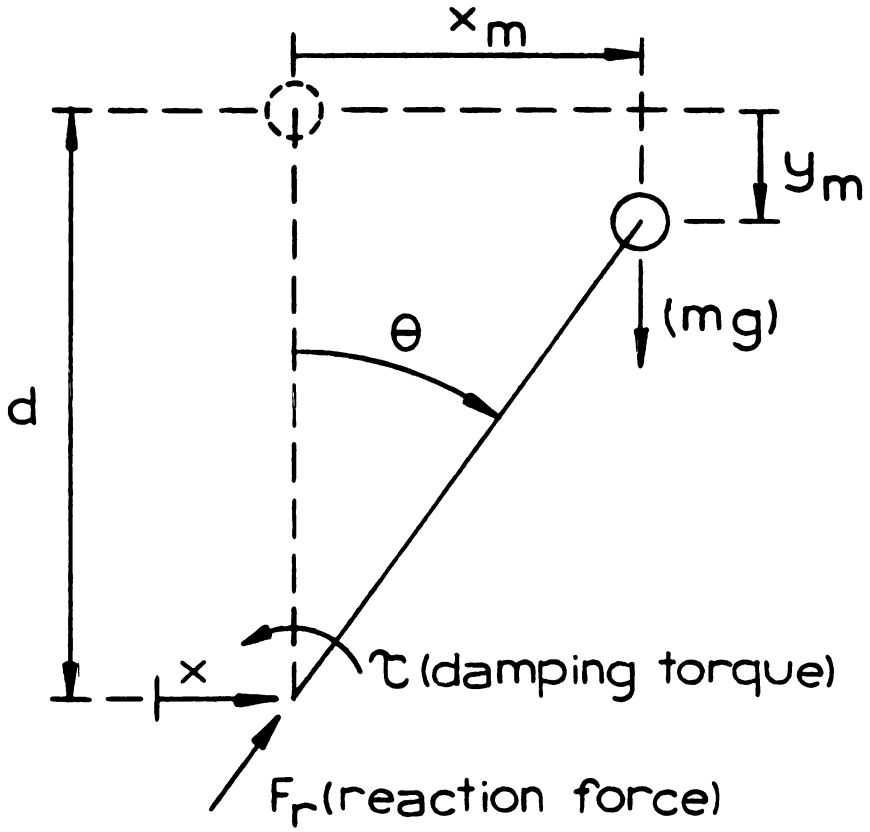


Figure A1: Pendulum free-body diagram.

Using the coordinate definitions in the pendulum free-body diagram of Figure A1, the magnitude of the pendulum velocity can be expressed as:

$$v_m^2 = (v + \dot{x}_m)^2 + (\dot{y}_m)^2 \qquad \text{(A4)}$$

Using simple trigonometric relations between the desired state variable (θ), and the position variables (x_m, y_m), the velocity can be written:

$$v_m^2 = v^2 + 2 d v \omega \cos(\theta) + (d \omega)^2 \qquad (A5)$$

Substituting (A5) into the original expression (A3), and rearranging terms yields the final expression for the kinetic energy of the system:

$$T = \frac{1}{2}(M + m) v^2 + \frac{1}{2} m (d \omega)^2 + m d v \omega \cos(\theta) \qquad (A6)$$

The non-conservative forces acting on this subsystem are the external applied driving force on the cart, and the damping forces. Lagranges method calls for formulation of the net non-conservative forces associated with each of the degrees of freedom. For this particular model, these net forces are given by:

$$Q_x = F - b_2 v - f_x \, sgn(v) \qquad (A7)$$

$$Q_\theta = - b_1 \omega - f_\theta \, sgn(\omega) \qquad (A8)$$

Evaluation of the derivatives of the potential and kinetic energy functions, and substitution of the results into Lagranges equation gives the two nonlinear system equations:

$$(M+m)\ddot{x} + md\ddot{\theta}\cos(\theta) - md\omega^2\sin(\theta) + b_2 v + f_x sgn(v) = F \qquad (A9)$$

$$md\ddot{x}\cos(\theta) + md^2\ddot{\theta} - mgd\sin(\theta) + b_1 \omega + f_\theta sgn(\omega) = 0 \qquad (A10)$$

These two nonlinear equations represent the dynamics of the cart/pendulum subsystem. The force input to this system is provided by the DC servo motor.

Since a linear model for the DC servo motor will be used, a bond graph is the most efficient method to derive the defining equations for this subsystem.

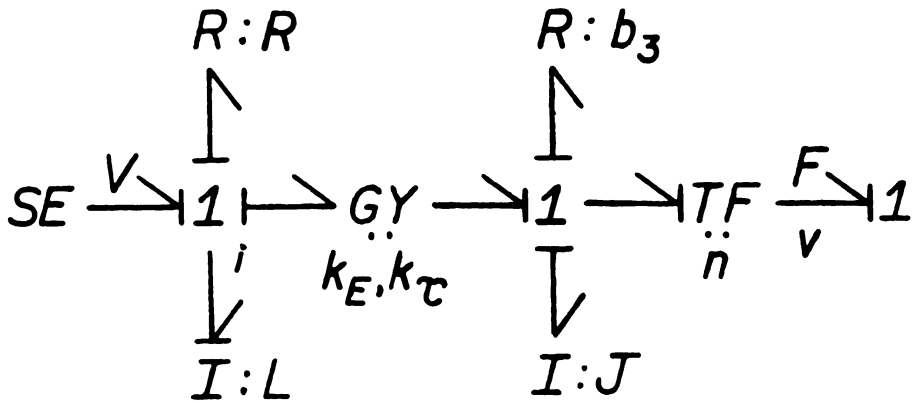


Figure A2: DC servo motor bond graph model

Note that since a force output (F) is desired from the motor, causality indicates that the model will only be first order. This is due to the assumed rigid coupling between the cart and the motor armature. The bond graph should therefore provide a force relation, and a single differential equation of motion for the motor. The force relation is given by:

$$F = (k_\tau\, n)\, i - (n^2 J)\, \ddot{x} - (n^2 b_3)\, v \qquad \text{(A11)}$$

and the equation of motion for the motor is given by:

$$L\,(di/dt) = V - i\,R - (k_E\, n)\, v \qquad \text{(A12)}$$

The force relation (A11), can be substituted directly into (A9), the first nonlinear system equation. Combining this result with (A10) and (A12) provides three nonlinear system equations describing the complete pendulum system dynamics.

$$(M+m+n^2J)\ddot{x} + (md)\ddot{\theta}\cos(\theta) - (md)\omega^2\sin(\theta) = (k_\tau n)i - (b_2+n^2b_3)v - f_x\,\mathrm{sgn}(v) \qquad \text{(A13)}$$

$$(md)\ddot{x}\cos(\theta) + (md^2)\ddot{\theta} - (mgd)\sin(\theta) = -b_1\omega - f_\theta\,\mathrm{sgn}(\omega) \qquad \text{(A14)}$$

$$L\,(di/dt) = V - i\,R - (k_E\,n)\,v \qquad \text{(A15)}$$

Another important nonlinear characteristic of the real system is the saturation of the applied motor voltage. The DC servo power supply has a finite limit, which can be expressed by:

$$|\,V\,| < 15.8 \text{ volts} \qquad \text{(A16)}$$

There are only three assumptions or approximations made in this model. The first is the use of a linear model to describe the DC servo motor. The second assumption is the rigid coupling between the motor armature and the cart. The third approximation is the characterization of the nonconservative damping forces. These areas alone govern the accuracy of the model to describe the pendulum system dynamics.

APPENDIX B

A linear set of equations are required to develop either a Linear State Feedback control law, or a Luenberger Observer. The nonlinear system equations must be linearized about some desired operating point in the state space. This is most efficiently accomplished through small angle approximations. Linearization of this type will remove all geometric nonlinearities. Only the discontinuous coulomb nonlinearities will remain. These coulomb nonlinearities will be carried through in the linear model, creating a hybrid linear plant with a nonlinear part.

Upon application of the small angle approximations to the system equations (A13), (A14), and (A15), the following set of equations result:

$$(M+m+n^2J)\ddot{x} + (md)\ddot{\theta} + (b_3+n^2b_3)v + f_x sgn(v) = (k_t n)i \qquad (B1)$$

$$(md)\ddot{x} + (md^2)\ddot{\theta} - (mgd)\theta + b_1\omega - f_\theta sgn(\omega) = 0 \qquad (B2)$$

$$L\,(di/dt) = V - iR - (k_E n)v \qquad (B3)$$

Solving equations (B1) and (B2) simultaneously for $\ddot{x}$ and $\ddot{\theta}$, combined with the definitions:

$$\dot{x} = v \qquad (B4)$$

$$\dot{\theta} = \omega \qquad (B5)$$

produces a set of linear equations which are suitable for matrix notation. The state space representation, including nonlinear coulomb

damping terms is given by:

$$\dot{\underline{X}}_s = \underline{A}_s \underline{X}_s + \underline{B}_s \underline{U} + \underline{C}_s \, \mathrm{sgn}(\underline{X}') \tag{B6}$$

$$\underline{Y} = \underline{D}_s \underline{X}_s \tag{B7}$$

Since some of the individual terms of the above matrices are lengthy expressions, a simpler notation has been adopted. The defining matrix equations are given by:

$$\frac{d}{dt}\begin{bmatrix} x \\ v \\ \theta \\ \omega \\ i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & -A1 & -A2 & A3 & A4 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & A5 & A6 & -A7 & -A8 \\ 0 & -A9 & 0 & 0 & -A10 \end{bmatrix} \begin{bmatrix} x \\ v \\ \theta \\ \omega \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ P1 \end{bmatrix} V + \begin{bmatrix} 0 & 0 \\ -C1 & C2 \\ 0 & 0 \\ C3 & -C4 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{sgn}(v) \\ \mathrm{sgn}(\omega) \end{bmatrix}$$

$$\begin{bmatrix} x \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ v \\ \theta \\ \omega \\ i \end{bmatrix}$$

The elements of the system matrix $\underline{A}$ are defined to be:

$$A1 = (n^2 b_3 + b_2)/(M + n^2 J) \tag{B8}$$

$$A2 = (mg)/(M + n^2 J) \tag{B9}$$

$$A3 = (b_1/d)/(M + n^2 J) \tag{B10}$$

$$A4 = (k_\tau n)/(M + n^2 J) \tag{B11}$$

$$A5 = (A1/d) \tag{B12}$$

$$A6 = (g/d)(M + m + n^2 J)/(M + n^2 J) \tag{B13}$$

$$A7 = (b_1/(md^2))(M + m + n^2 J)/(M + n^2 J) \tag{B14}$$

$$A8 = (k_\tau n/d)/(M + n^2 J) \tag{B15}$$

$$A9 = (k_E n/d) \tag{B16}$$

$$A10 = (R/L) \tag{B17}$$

The input matrix $\underline{B}$ has only one non-zero element, and is defined to be:

$$P1 = (1.0/L) \tag{B18}$$

The coulomb damping matrix $\underline{C}$ has four non-zero elements, which are defined to be:

$$C1 = (f_x)/(M + n^2J) \tag{B19}$$

$$C2 = (f_\theta/d)/(M + n^2J) \tag{B20}$$

$$C3 = C1/d \tag{B21}$$

$$C4 = (f_\theta/(md^2))(M + m + n^2J)/(M + n^2J) \tag{B22}$$

This particular notation was chosen because of the significant number of zero elements in the matrices. All of the newly defined parameters (matrix elements) are functions of the original set of system parameters. All elements have a positive value, characteristic of a parameter. The signs of individual terms in the system equations remain in the matrix definitions.

APPENDIX C

The problem of controlling the linear system with state feedback translates to one of choosing the five control gains $\underline{K}_s$ such that the poles of $[\underline{A}-\underline{B}\underline{K}]_s$ are in the left half plane, where $\underline{A}$ and $\underline{B}$ are the fifth order linear system matrices derived in Appendix B.

Equation (C1) is the characteristic polynomial whose roots are the five desired closed loop poles:

$$s^5 + \xi_4 s^4 + \xi_3 s^3 + \xi_2 s^2 + \xi_1 s + \xi_0 = 0 \qquad \text{(C1)}$$

The coefficients ξ_i are functions of the desired closed loop poles, and represent constant values for a known set of poles.

The actual system characteristic polynomial is obtained by taking the determinant of $[s\underline{I}-(\underline{A}-\underline{B}\underline{K})]_s$. The coefficients of this fifth order polynomial are functions of the elements of the $\underline{A}_s$, $\underline{B}_s$, and $\underline{K}_s$ matrices. Equating these coefficients to those of the desired closed loop polynomial produces five equations in the five unknown control gains of $\underline{K}_s$. These relations are given by:

$$\xi_0 = P1K1(A2A8-A4A6) \tag{C2}$$

$$\xi_1 = P1(K1(A4A7-A3A8) + K2(A2A8-A4A6) + K3(A4A5-A1A8) + K5(A2A5-A1A6) + A10(A2A5-A1A6) + A9(A2A8-A4A6)) \tag{C3}$$

$$\xi_2 = P1(K1(A4) + K2(A4A7-A3A8) + K4(A4A5-A1A8) + K5(A1A7-A3A5-A6)) + A10(A1A7-A3A5-A6) + A9(A4A7-A3A8) - K3(A8) + (A2A5-A1A6) \tag{C4}$$

$$\xi_3 = P1(K2(A4) - K4(A8) + K5(A1+A7)) + A1(A7+A10) + A7A10 - A3A5 - A6 + A4A9 \tag{C5}$$

$$\xi_4 = P1(K5) + (A1+A7+A10) \tag{C6}$$

The LSF control gains $\underline{K}_s$ can be obtained by rearranging equations (C2) through (C6), and solving the equivalent linear matrix equation:

$$\underline{H}\,\underline{K}_s^T = \underline{F} \tag{C7}$$

where the $\underline{H}$ matrix is given by:

$$\underline{H} = P1\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & (A4) & 0 & (-A8) & (A1+A7) \\ (A4) & (A4A7-A3A8) & (-A8) & (A4A5-A1A8) & (A1A7-A3A5-A6) \\ (A4A7-A3A8) & (A2A8-A4A6) & (A4A5-A1A8) & 0 & (A2A5-A1A6) \\ (A2A8-A4A6) & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the $\underline{F}$ matrix is given by:

$$\underline{F} = \begin{bmatrix} \xi_4 - (A1+A7+A10) \\ \xi_3 - (A1(A7+A10)+A7A10-A3A5-A6+A4A9) \\ \xi_2 - (A10(A1A7-A3A5-A6)+A9(A4A7-A3A8)+(A2A5-A1A6)) \\ \xi_1 - (A10(A2A5-A1A6)+A9(A2A8-A4A6)) \\ \xi_0 - 0 \end{bmatrix}$$

Given numerical values for the desired set of closed loop poles, and the system parameters, (C7) can be solved for the stabilizing control gains $\underline{K}_s$ in the linear state feedback law.

To eliminate the need for measurement of the armature current state, the fifth control gain K5 must be be identically zero. For this

to be possible, all five of the linear equations in (C7) must still be satisfied. Equation (C6) is the only one containing the single control gain K5. Setting this current gain to zero, the remaining equation is:

$$\xi_4 = (A1 + A7 + A10) \tag{C8}$$

By choosing the desired set of closed loop poles such that (C8) is satisfied, ξ_4 will take on a value that will force the current gain to be zero.

The algorithm above is implemented in the program CALC (Appendix F). In that program, only two sets of desired closed loop poles are considered:

$$\eta_1 = \{-\alpha_1, -\alpha_2, -\alpha_3, -\alpha_4, -\alpha_5\} \tag{C9}$$

$$\eta_2 = \{-\alpha_1 \pm j\beta_1, -\alpha_2 \pm j\beta_2, -\alpha_3\} \tag{C10}$$

For each of these sets, the respective values of ξ_4 are:

$$\{\xi_4\}_1 = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 \tag{C11}$$

$$\{\xi_4\}_2 = 2(\alpha_1 + \alpha_2) + \alpha_3 \tag{C12}$$

The user must specify four desired closed loop poles from one of the above sets. The fifth pole is then computed such that (C8) will be satisfied. By choosing the computed fifth pole, the current gain is forced to be identically zero.

APPENDIX D

A linear Luenberger Observer can be designed for the linear part of the reduced fourth order model derived in Appendix K. The complete fourth order model is given by:

$$\frac{d}{dt}\begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -A11 & -A2 & A3 \\ 0 & 0 & 0 & 1 \\ 0 & A12 & A6 & -A7 \end{bmatrix}\begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ A13 \\ 0 \\ -A14 \end{bmatrix} V + \begin{bmatrix} 0 & 0 \\ -C1 & C2 \\ 0 & 0 \\ C3 & -C4 \end{bmatrix}\begin{bmatrix} sgn(v) \\ sgn(\omega) \end{bmatrix}$$

Neglecting the coulomb damping terms, a linear Luenberger observer for full state observation will have the form:

$$\dot{\hat{\underline{X}}} = \underline{A}\,\hat{\underline{X}} + \underline{B}\,\underline{U} + \underline{G}\,(\underline{Y} - \hat{\underline{Y}}) \qquad \text{(D1)}$$

$$\hat{\underline{Y}} = \underline{D}\,\hat{\underline{X}} \qquad \text{(D2)}$$

where $\underline{A}$ and $\underline{B}$ are the system matrices given above, and the matrix of unknown observer gains $\underline{G}$ is given by:

$$\underline{G} = \begin{bmatrix} G1 & G5 \\ G2 & G6 \\ G3 & G7 \\ G4 & G8 \end{bmatrix} \qquad \text{(D3)}$$

The observed states can be made to converge to the actual linear system states by choosing the poles of $[\underline{A}-\underline{G}\underline{D}]$ to be to the left of the closed loop system poles. The observer gain solution procedure is similar to the LSF control gain solution found in Appendix C.

First the characteristic polynomial of $[\underline{A}-\underline{G}\underline{D}]$ is obtained. The coefficients of this equation can be written in terms of the system parameters, i.e. the elements of $\underline{A}$, $\underline{D}$, and the observer gains $\underline{G}$. For a desired set of observer poles, the characteristic polynomial will have the form:

$$s^4 + \xi_3 s^3 + \xi_2 s^2 + \xi_1 s + \xi_0 = 0 \qquad \text{(D4)}$$

Equating the coefficients of the two characteristic polynomials:

$$\xi_3 = (A11+A7) + (G1+G7) \qquad \text{(D5)}$$

$$\xi_2 = (A11+A7)(G1+G7)+(G1G7+A11A7)-(A12A3+G4G6)+(G8+G2-A6) \qquad \text{(D6)}$$

$$\xi_1 = (G1+A11)(A7G7+G8-A6)+G1A11(A7+G7)-A12A3(G1+G7)+G4(A3-G5)+ A12(A2+G6)-G3(A2+G6+G5(A11+A7))+G2(A7+G7) \qquad \text{(D7)}$$

$$\xi_0 = (G1A11+G2)(A2G7+G8-A6)+(A12G1-G4)(A2+G6-A3G7)- G3(A3(G8-A6-G5A12)+A7(A2+G6+A11G5))-G5(G4A11+G2A12) \qquad \text{(D8)}$$

These four equations must be solved for the eight observer gains. Since the system of equations is underdetermined, four of the gains are arbitrary [10]. From observability calculations (Appendix G), it was determined that the linear system is observable with only the cart position as input. Since they are arbitrary, the four gains associated with the pendulum position may be set to zero:

$$G5 = G6 = G7 = G8 = 0 \qquad \text{(D9)}$$

Applying (D9) to equations (D5) through (D8), produces a linear set of equations in the unknown gains of $\underline{G}$, having the form:

$$\underline{S}\ \mathrm{col}_1(\underline{G}) = \underline{T} \qquad \text{(D10)}$$

where the $\underline{S}$ matrix is given by:

$$\underline{S} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ (A11+A7) & 1 & 0 & 0 \\ (A11A7-A3A12-A6) & (A7) & (-A2) & (A3) \\ (A2A12-A11A6) & (-A6) & (A3A6-A2A7) & (-A2) \end{bmatrix} \tag{D11}$$

and the $\underline{T}$ matrix is given by:

$$\underline{T} = \begin{bmatrix} \xi_3 - (A11+A7) \\ \xi_2 - (A11A7-A3A12-A6) \\ \xi_1 - (A2A12-A11A6) \\ \xi_0 \end{bmatrix} \tag{D12}$$

It is interesting to note that, given a real set of system parameters, there is a unique solution for the observer gains of (D10). If the cart position measurement had been eliminated instead of the pendulum position, a new system of four equations would have been produced which has no unique solution. This result is as predicted by the observability tests.

The algorithm for determining the observer gains $\underline{G}$ reduces to solving the linear system of equations (D10). This is implemented in the program CALC (Appendix F). All that is required on input are the system parameter values, and the desired observer pole locations. The only desired set of observer poles considered here is:

$$\eta = \{-\alpha_1, -\alpha_2, -\alpha_3, -\alpha_4\} \tag{D13}$$

For this particular set of poles, the coefficients of the characteristic polynomial are:

$$\xi_3 = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 \tag{D14}$$

$$\xi_2 = \alpha_1\alpha_2 + (\alpha_1 + \alpha_2)(\alpha_3 + \alpha_4) + \alpha_3\alpha_4 \tag{D15}$$

$$\xi_1 = \alpha_1\alpha_2(\alpha_3 + \alpha_4) + \alpha_3\alpha_4(\alpha_1 + \alpha_2) \tag{D16}$$

$$\xi_0 = \alpha_1\alpha_2\alpha_3\alpha_4 \tag{D17}$$

Substituting numeric values for these coefficients and the system parameters into the matrices of (D10), a solution for the observer gains can be obtained with a linear algebra subroutine.

APPENDIX E

A root locus program was developed as an interactive tool to study pole placement for linear state space form systems with linear state feedback implemented as the control scheme. The root locus study performed in this research was based on a very simple algorithm given the pre-existence of several canned software packages. The user must input the A (system), B (input), and K (control gain) matrices defining the linear system. Varying a single element of one of the three matrices generates a root locus as the eigenvalues of the new system matrix [A-BK] change.

The FORTRAN 77 source code below was developed on the PRIME 750 computer system for use on the TEKTRONIX graphics terminals, at the MSU Case Center for Computer Aided Design. The Case Center Mathematical Library Subroutine package provides subroutines for the matrix manipulations [2]. The eigenvalues are computed with the EIGRF subroutine from the IMSL Eigensystem Analysis package [2]. After calculation of the eigenvalues, the root locus is plotted on the terminal screen via TCS graphic routines from the Tektronix PLOT10 package.

```
      PROGRAM LOCUS
C
C  This is the main program to compute and plot the
C  root locus for a matrix system with linear state
C  feedback.
C
C  -----Declare all variables
C
      REAL A(9,9),B(9,9),K(9,9),G(9,9),WK(9)
C
      INTEGER N,M
C
      COMPLEX EGV(9)
C
C  -----Format Statements
C
  100 FORMAT(/,/,15X,'ROOT-LOCUS EVALUATOR FOR MATRICES',
     +   /,/,10X,'DX[N,1] = A[N,N] * X[N,1] + B[N,M] * U[M,1]',
     +   /,/,18X,'U[M,1] = K[M,N] * X[N,1]')
  110 FORMAT(/,/,5X,'ENTER THE DIMENSION  N: ',$)
  120 FORMAT(/,5X,'ENTER THE DIMENSION  M: ',$)
  130 FORMAT(/,5X,'DIMENSION MUST BE LESS THAN 10.')
C
C  -----Input the dimensions
C
      WRITE(1,100)
   10 WRITE(1,110)
      READ(1,*,ERR=10) N
      IF (N .GE. 10) THEN
        WRITE(1,130)
        GOTO 10
      ENDIF
C
   20 WRITE(1,120)
      READ(1,*,ERR=20) M
      IF (M .GE. 10) THEN
        WRITE(1,130)
        GOTO 10
      ENDIF
C
C  -----Go on to the main program
C
      CALL INPUT(N,M,A,B,K,G,WK,EGV)
C
      END
```

```
      SUBROUTINE INPUT(N,M,A,B,K,G,WK,EGV)
C
C  This subroutine accepts the input matrices
C  A, B, and K, and also the matrix element to vary
C  between the specified limits.
C
C  -----Declare all variables
C
      REAL A(N,N),B(N,M),K(M,N),LL,UL,DL,G(N,N),WK(N),
     +     XMIN,XMAX,YMIN,YMAX
C
      INTEGER N,M,I,J,IP,JP
C
      COMPLEX EGV(N)
C
      CHARACTER MAT*1,COMAND*9
C
      COMMON /BLOCK/ XMIN,XMAX,YMIN,YMAX
C
C  -----Format Statements
C
  700 FORMAT(/,/,5X,'ENTER THE A[N,N] MATRIX. . .')
  710 FORMAT(/,5X,'ENTER THE B[N,M] MATRIX. . .')
  720 FORMAT(/,5X,'ENTER THE K[M,N] MATRIX. . .')
  730 FORMAT(10X,'A(',I1,',',I1,') = ',$)
  740 FORMAT(10X,'B(',I1,',',I1,') = ',$)
  750 FORMAT(10X,'K(',I1,',',I1,') = ',$)
  760 FORMAT(/,/,5X,'ONLY ONE ELEMENT OF THE THREE MATRICES',
     +   /,10X,'A, B, AND K, MAY BE VARIED.')
  770 FORMAT(/,/,5X,'CHOOSE ONE OF THE MATRICES (A,B,K): ',$)
  780 FORMAT(/,5X,'YOU MAY CHOOSE THE ELEMENT ',A1,'(I,J)')
  800 FORMAT(/,10X,'ENTER THE ROW POSITION     I: ',$)
  810 FORMAT(/,10X,'ENTER THE COLUMN POSITION  J: ',$)
  820 FORMAT(/,/,5X,'ENTER THE MINIMUM VALUE OF ',A1,'(',I1,$)
  830 FORMAT(/,5X,'ENTER THE MAXIMUM VALUE OF ',A1,'(',I1,$)
  840 FORMAT(',',I1,') : ',$)
  860 FORMAT(/,/,5X,'ENTER THE INCREMENT: ',$)
  870 FORMAT(/,10X,'(Y) OR (N) ',$)
C
      XMIN = -10.0
      XMAX = 10.0
      YMIN = -7.0
      YMAX = (XMAX-XMIN)*780.0/1024.0 + YMIN
C
C  -----Input the A matrix
C
      WRITE(1,700)
      WRITE(1,*)'  '
      DO 30  I = 1,N
        DO 20  J = 1,N
   10     WRITE(1,730) I,J
          READ(1,*,ERR=10) A(I,J)
   20   CONTINUE
        WRITE(1,*)'  '
```

```
   30 CONTINUE
C
C  -----Input the B matrix
C
      WRITE(1,710)
      WRITE(1,*)'  '
      DO 60  I = 1,N
        DO 50  J = 1,M
   40     WRITE(1,740) I,J
          READ(1,*,ERR=40) B(I,J)
   50   CONTINUE
        WRITE(1,*)'  '
   60 CONTINUE
C
C  -----Input the K matrix
C
   62 WRITE(1,720)
      WRITE(1,*)'  '
      DO 80  I = 1,M
        DO 70  J = 1,N
   65     WRITE(1,750) I,J
          READ(1,*,ERR=65) K(I,J)
   70   CONTINUE
        WRITE(1,*)'  '
   80 CONTINUE
C
C  -----Choose the parameter to vary
C
   85 WRITE(1,760)
   90 WRITE(1,770)
      READ(1,'(A1)',ERR=90) MAT
      IF ((MAT.NE.'K').AND.(MAT.NE.'B').AND.(MAT.NE.'A')) GOTO 90
C
      WRITE(1,780) MAT
  110 WRITE(1,800)
      READ(1,*,ERR=110) IP
      IF (IP .LT. 1) GOTO 110
C
      IF (MAT .EQ. 'K') THEN
        IF (IP .GT. M) GOTO 110
      ELSE
        IF (IP .GT. N) GOTO 110
      ENDIF
C
  120 WRITE(1,810)
      READ(1,*,ERR=120) JP
      IF (JP .LT. 1) GOTO 120
C
      IF (MAT .EQ. 'B') THEN
        IF (JP .GT. M) GOTO 120
      ELSE
        IF (JP .GT. N) GOTO 120
      ENDIF
C
```

```
C  -----Get the limits on the variable parameter
C
  130 WRITE(1,820) MAT,IP
      WRITE(1,840) JP
      READ(1,*,ERR=130) LL
C
  140 WRITE(1,830) MAT,IP
      WRITE(1,840) JP
      READ(1,*,ERR=140) UL
C
      IF (UL .LT. LL) GOTO 140
C
  150 WRITE(1,860)
      READ(1,*,ERR=150) DL
      DL = ABS(DL)
C
C  -----Calculate and plot the root locus
C
      CALL CRUNCH(N,M,A,B,K,G,WK,EGV,MAT,IP,JP,LL,UL,DL)
C
  160 WRITE(1,*)'  '
      WRITE(1,*)'     DO YOU WANT TO VARY A NEW PARAMETER. . .'
      WRITE(1,870)
      READ(1,'(A9)',ERR=160) COMAND
C
      IF (COMAND(1:1) .NE. 'N') THEN
        GOTO 85
      ENDIF
C
  170 WRITE(1,*)'  '
      WRITE(1,*)'     DO YOU WANT TO INPUT A NEW K MATRIX. . .'
      WRITE(1,870)
      READ(1,'(A9)',ERR=170) COMAND
C
      IF (COMAND(1:1) .NE. 'N') THEN
        GOTO 62
      ELSE
        WRITE(1,*)'  '
        WRITE(1,*)' OK,. . . .'
        WRITE(1,*)'  '
      ENDIF
C
      RETURN
C
      END
```

```
      SUBROUTINE CRUNCH(N,M,A,B,K,G,WK,EGV,MAT,IP,JP,LL,UL,DL)
C
C  This is the main number crunching routine to
C  calculate and plot the root locus.
C
C  -----Declare all variables
C
      REAL A(N,N),B(N,M),K(M,N),LL,UL,DL,V0,VA,
     +     XMIN,XMAX,YMIN,YMAX,G(N,N),WK(N)
C
      INTEGER N,M,IP,JP
C
      COMPLEX EGV(N),Z(9,9)
C
      CHARACTER MAT*1,NAME*9,COMAND*9
C
      COMMON /BLOCK/ XMIN,XMAX,YMIN,YMAX
C
C  -----Format Statements
C
  500 FORMAT(/,/,5X,'WOULD YOU LIKE TO NAME THE G_FILE...')
  510 FORMAT(/,5X,'ENTER THE NEW FILENAME: ',$)
  520 FORMAT(/,/,5X,'WOULD YOU LIKE TO RESET THE WINDOW...')
  530 FORMAT(/,10X,'(Y) OR (N) ',$)
  540 FORMAT(3X,F8.2,9(2X,:,'(',F8.2,',',F8.2,')'))
  550 FORMAT(/,5X,A1,'(',I1,',',I1,')',10X,
     +     'EIGENVALUES OF THE SYSTEM (RE,IM). . .',/)
C
C  -----Calculate and plot the root locus
C
   10 WRITE(1,500)
      WRITE(1,530)
      READ(1,'(A9)',ERR=10) COMAND
C
      IF (COMAND(1:1) .EQ. 'Y') THEN
   20   WRITE(1,510)
        READ(1,'(A9)',ERR=20) NAME
      ELSE
        NAME = 'G_LOCUS'
      ENDIF
C
      IF (MAT .EQ. 'K') THEN
        V0 = K(IP,JP)
      ELSE IF (MAT .EQ. 'A') THEN
        V0 = A(IP,JP)
      ELSE
        V0 = B(IP,JP)
      ENDIF
C
      OPEN(5,FILE='LOCUS.DAT')
      WRITE(5,*)'  '
      WRITE(5,*)'     ROOT LOCUS DATA FOR MATRIX SYSTEM'
      WRITE(5,550) MAT,IP,JP
C
```

```
      LJOB = 0
      COMAND = 'xxxxxxxxx'
C
C  -----Do the graphics kind of stuff
C
      CALL INITT(480)
      CALL OPENTK(NAME,IER)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      CALL AXIS(1.0,1.0,8)
C
      DO 40  VA = LL,UL,DL
        IF (MAT .EQ. 'K') THEN
          K(IP,JP) = VA
        ELSE IF (MAT .EQ. 'A') THEN
          A(IP,JP) = VA
        ELSE
          B(IP,JP) = VA
        ENDIF
C
        CALL MMLT(G,B,K,N,M,N)
C
        DO 30  I = 1,N
          DO 25  J = 1,N
            G(I,J) = A(I,J) + G(I,J)
   25     CONTINUE
   30   CONTINUE
C
        CALL EIGRF(G,N,N,LJOB,EGV,Z,N,WK,IER)
C
        WRITE(5,540) VA,(EGV(I),I=1,N)
C
        DO 35 I = 1,N
          X = REAL(EGV(I))
          Y = AIMAG(EGV(I))
          IF ((X.GT.XMIN).AND.(X.LT.XMAX).AND.
     +        (Y.GT.YMIN).AND.(Y.LT.YMAX)) THEN
            CALL MOVEA(X,Y)
            CALL MOVREL(-3,-4)
            CALL CHARTK(COMAND(I:I),0.6)
          ENDIF
   35   CONTINUE
C
   40 CONTINUE
C
      CALL MOVABS(390,730)
      CALL CHARTK('ROOT LOCUS PLOT',1.2)
      CALL MOVEA(0.0,YMAX)
      CALL MOVREL(18,-18)
      CALL CHARTK('IM',1.0)
      CALL MOVEA(XMAX,0.0)
      CALL MOVREL(-23,-23)
      CALL CHARTK('RE',1.0)
      CALL MOVABS(25,20)
      WRITE(COMAND,'(F9.3)') LL
```

```
      CALL CHARTK(COMAND,1.0)
      CALL CHARTK(' <',1.0)
      CALL MOVREL(20,0)
      CALL CHARTK(MAT,1.0)
      CALL CHARTK('(',1.0)
      WRITE(COMAND,'(I1)') IP
      CALL CHARTK(COMAND(1:1),1.0)
      CALL CHARTK(',',1.0)
      WRITE(COMAND,'(I1)') JP
      CALL CHARTK(COMAND(1:1),1.0)
      CALL CHARTK(') <',1.0)
      WRITE(COMAND,'(F9.3)') UL
      CALL CHARTK(COMAND,1.0)
      CALL MOVABS(0,700)
C
      CALL CLOSTK(IER)
      CALL ANMODE
      CALL HOME
      CLOSE(5)
C
      IF (MAT .EQ. 'K') THEN
        K(IP,JP) = V0
      ELSE IF (MAT .EQ. 'A') THEN
        A(IP,JP) = V0
      ELSE
        B(IP,JP) = V0
      ENDIF
C
C     -----Reset the window
C
   50 WRITE(1,520)
      WRITE(1,530)
      READ(1,'(A9)',ERR=50) COMAND
C
      IF (COMAND(1:1) .EQ. 'Y') THEN
   60   WRITE(1,*)'  '
        WRITE(1,*)'          XMIN = ',XMIN
        WRITE(1,*)'          XMAX = ',XMAX
        WRITE(1,*)'          YMIN = ',YMIN
        WRITE(1,*)'          YMAX = ',YMAX
        WRITE(1,*)'  '
        WRITE(1,*)'     ENTER XMIN, XMAX, AND YMIN. . .'
        WRITE(1,*)'  '
        READ(1,*,ERR=60) XMIN,XMAX,YMIN
        YMAX = (XMAX-XMIN)*780.0/1024.0 + YMIN
        GOTO 10
      ENDIF
C
      RETURN
C
      END
```

APPENDIX F

The Linear State Feedback control gains and the Luenberger Observer gains are all functionally dependent on the linear system model parameters. The solution algorithms for both sets are similar in form. For this reason, the single program CALC was written as an interactive program to facilitate the computation of either set of gains. After supplying the basic system parameters, the user may compute the control gains for either the controller, or the observer, placing the poles of each at user specified locations.

Both the Linear State Feedback controller and Luenberger Observer are formulations of linear theory. The solution for the gains with the indicated pole placement configuration reduces to that of solving a linear system of equations, as derived in Appendices C and D. The LINEQ subroutine, supplied by the Mathematical Subroutine Library of the Case Center [2], is used to perform these calculations.

Below is an annotated sample interactive output showing the control gains computed for the given input parameters, and desired closed loop pole locations for both the LSF controller, and the Luenberger observer. The gain solutions here are the default gains in the implemented control routine. For clarity, underlined comments have been added to the output. Following this output is the source code for CALC.

```
ENTER MP:  0.0236      Pendulum mass (slugs)

ENTER MC:  0.0385      Cart mass (slugs)

ENTER D:   1.2396      Pendulum length (ft)

ENTER B1:  1.13E-4     Pendulum viscous damping coeff. (ft-lb-s)

ENTER B2:  0.025       Cart viscous damping coeff. (lb-s/ft)

ENTER B3:  2.16E-4     Motor viscous damping coeff. (ft-lb-s)

ENTER J:   5.2E-5      Armature inertia (ft-lb-s^2)

ENTER L:   6.407E-3    Armature inductance (henry)

ENTER R:   17.97       Armature electrical resistance (Ω)

ENTER KE:  0.0813      Motor gyrator constant (V-s/rad)

ENTER N:   32.0        Gear reduction (rad/ft)

ENTER FX:  0.0698      Cart coulomb damping force (lb)

ENTER FT:  0.002       Pendulum coulomb damping force (lb)

 A1 =      2.68326      A2 =      8.27600
 A3 =    9.935740E-04   A4 =     20.9115
 A5 =      2.16462      A6 =     32.6315
 A7 =    3.917570E-03   A8 =     16.8695
 A9 =     406.056       A10=     2804.74
 A11=      5.71071      A12=      4.60690
 A13=      1.16369      A14=     0.938760
 P1 =     156.079

 C1 =     0.760780      C2 =    1.758538E-02
 C3 =     0.613730      C4 =    6.933752E-02

 (L)INEAR STATE FEEDBACK DESIGN
           - OR -
      (O)BSERVER DESIGN

DESIGN:    L

 WHAT SET OF SYSTEM ROOTS DO YOU WANT. . .

 1.) {-RR1+-J*RI1,-RR2+-J*RI2,-RR3}

 2.) {-RR1,-RR2,-RR3,-RR4,-RR5}

2
```

```
ENTER RR1: 3

ENTER RR2: 4

ENTER RR3: 5

ENTER RR4: 6

 FOR  K5=0 . . . RR5 =       2789.43

ENTER RR5: 2789.43


 THE CHARACTERISTIC EQUATION IS:

    S**5 :       1.0
    S**4 :       2807.43
    S**3 :       50328.7
    S**2 :       332284.
    S**1 :       954345.
    S**0 :      1.004195E+06

 THE K MATRIX OF GAINS IS:

  -11.85394    -16.17418   -175.72162    -33.08949     -0.00175

 IER =              0

 (L)INEAR STATE FEEDBACK DESIGN
             - OR -
      (O)BSERVER DESIGN


DESIGN:     0

 ENTER THE OBSERVER POLES . . .


ENTER RR1: 8

ENTER RR2: 8

ENTER RR3: 9

ENTER RR4: 9

 THE G MATRIX OF GAINS IS:

      28.285       303.973      -425.307     -2331.515

 IER =              0
```

```
      PROGRAM CALC
C
C  This program calculates the LSF control gains for the
C  the 5th order pendulum and motor model, given the system
C  parameters as input.
C
C  -----Declare all variables
C
      REAL MP,MC,D,G,B1,B2,B3,J,L,R,KE,N,FX,FT
C
      REAL A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,P1
C
      REAL RR1,RI1,RR2,RI2,RR3,RR4,RR5
C
      REAL X0,X1,X2,X3,X4,Q1,Q2,Q3,Q4
C
      REAL H(5,5),K(5),F(5),WORK(6,6)
C
      REAL S(4,4),GA(4),T(4),WERK(5,5)
C
      CHARACTER COMAND*9
C
C  -----INPUT THE SYSTEM PARAMETERS
C
   10 FORMAT(/,A12,$)
      WRITE(1,*)'  '
      WRITE(1,10)' ENTER MP:   '
      READ(1,*) MP
      WRITE(1,10)' ENTER MC:   '
      READ(1,*) MC
      WRITE(1,10)' ENTER D:    '
      READ(1,*) D
      G = 32.174
      WRITE(1,10)' ENTER B1:   '
      READ(1,*) B1
      WRITE(1,10)' ENTER B2:   '
      READ(1,*) B2
      WRITE(1,10)' ENTER B3:   '
      READ(1,*) B3
      WRITE(1,10)' ENTER J:    '
      READ(1,*) J
      WRITE(1,10)' ENTER L:    '
      READ(1,*) L
      WRITE(1,10)' ENTER R:    '
      READ(1,*) R
      WRITE(1,10)' ENTER KE:   '
      READ(1,*) KE
      WRITE(1,10)' ENTER N:    '
      READ(1,*) N
      WRITE(1,10)' ENTER FX:   '
      READ(1,*) FX
      WRITE(1,10)' ENTER FT:   '
      READ(1,*) FT
C
```

```
C  -----CALCULATE THE MATRIX PARAMETERS
C
      A1 = (N*N*B3+B2)/(MC+N*N*J)
      A2 = (MP*G)/(MC+N*N*J)
      A3 = B1/(D*(MC+N*N*J))
      A4 = KE*N/(1.356*(MC+N*N*J))
      A5 = (N*N*B3+B2)/(D*(MC+N*N*J))
      A6 = (G/D)*(MC+MP+N*N*J)/(MC+N*N*J)
      A7 = (B1/(D*D*MP))*(MC+MP+N*N*J)/(MC+N*N*J)
      A8 = KE*N/(1.356*D*(MC+N*N*J))
      A9 = KE*N/L
      A10 = R/L
      P1 = 1.0/L
      A11 = A1 + A4*A9/A10
      A12 = A5 + A8*A9/A10
      A13 = A4*P1/A10
      A14 = A8*P1/A10
      C1 = FX/(MC+N*N*J)
      C2 = (FT/D)/(MC+N*N*J))
      C3 = C1/D
      C4 = (FT/(MP*D*D)*(MC+MP+N*N*J)/(MC+N*N*J)
C
      WRITE(1,*)'  '
      WRITE(1,*)' A1 = ',A1,'   A2 = ',A2
      WRITE(1,*)' A3 = ',A3,'   A4 = ',A4
      WRITE(1,*)' A5 = ',A5,'   A6 = ',A6
      WRITE(1,*)' A7 = ',A7,'   A8 = ',A8
      WRITE(1,*)' A9 = ',A9,'   A10= ',A10
      WRITE(1,*)' A11= ',A11,'  A12= ',A12
      WRITE(1,*)' A13= ',A13,'  A14= ',A14
      WRITE(1,*)' P1 = ',P1
      WRITE(1,*)'  '
      WRITE(1,*)' C1 = ',C1,'   C2 = ',C2
      WRITE(1,*)' C3 = ',C3,'   C4 = ',C4
      WRITE(1,*)'  '
C
C  -----CHOOSE BETWEEN OBSERVER OR LSF DESIGN
C
   50 WRITE(1,*)'  '
      WRITE(1,*)' (L)INEAR STATE FEEDBACK DESIGN'
      WRITE(1,*)'            - OR -'
      WRITE(1,*)'      (O)BSERVER DESIGN'
      WRITE(1,*)'  '
      WRITE(1,10) ' DESIGN:    '
      READ(1,'(A9)') COMAND
C
      IF (COMAND(1:1).EQ.'L') THEN
         GOTO 100
      ELSE IF (COMAND(1:1).EQ.'O') THEN
         GOTO 590
      ELSE IF (COMAND(1:1).EQ.'Q') THEN
         CALL EXIT
      ELSE
         GOTO 50
```

```
      ENDIF
C
C  -----GET THE TYPE OF CHARACTERISTIC ROOTS
C
  100 WRITE(1,*)'  '
      WRITE(1,*)'  '
      WRITE(1,*)' WHAT SET OF SYSTEM ROOTS DO YOU WANT. . .'
      WRITE(1,*)'  '
      WRITE(1,*)' 1.) {-RR1+-J*RI1,-RR2+-J*RI2,-RR3}'
      WRITE(1,*)'  '
      WRITE(1,*)' 2.) {-RR1,-RR2,-RR3,-RR4,-RR5}'
      WRITE(1,*)'  '
      READ(1,*,ERR=100) ISET
      IF ((ISET.NE.1).AND.(ISET.NE.2)) GOTO 100
C
      GOTO (200,300) ISET
C
C  -----SET  1: TWO COMPLEX PAIRS AND ONE REAL ROOT
C
  200 WRITE(1,*)'  '
      WRITE(1,10)' ENTER RR1: '
      READ(1,*) RR1
      WRITE(1,10)' ENTER RI1: '
      READ(1,*) RI1
      WRITE(1,10)' ENTER RR2: '
      READ(1,*) RR2
      WRITE(1,10)' ENTER RI2: '
      READ(1,*) RI2
      WRITE(1,*)'  '
      RR3 = (A1+A7+A10) - (2.0*(RR1+RR2))
      WRITE(1,*)' FOR  K5=0 . . . RR3 = ',RR3
      WRITE(1,10)' ENTER RR3: '
      READ(1,*) RR3
C
C  -----CALCULATE THE COEFFICIENTS OF CE
C
      Q1 = RR1*RR1 + RI1*RI1
      Q2 = RR2*RR2 + RI2*RI2
      X0 = RR3 * Q1 * Q2
C
      Q3 = 2*(RR1*Q2 + RR2*Q1)
      X1 = RR3 * Q3 + Q1 * Q2
C
      Q4 = 4*RR1*RR2 + Q1 + Q2
      X2 = RR3*Q4 + Q3
C
      X3 = 2*RR3*(RR1+RR2) + Q4
      X4 = RR3 + 2*(RR1 + RR2)
C
      GOTO 400
C
C  -----SET  2: ALL REAL ROOTS
C
  300 WRITE(1,*)'  '
```

```
      WRITE(1,10)' ENTER RR1: '
      READ(1,*) RR1
      WRITE(1,10)' ENTER RR2: '
      READ(1,*) RR2
      WRITE(1,10)' ENTER RR3: '
      READ(1,*) RR3
      WRITE(1,10)' ENTER RR4: '
      READ(1,*) RR4
      WRITE(1,*)'  '
      RR5  = (A1+A7+A10) - (RR1+RR2+RR3+RR4)
      WRITE(1,*)' FOR  K5=0 . . . RR5 = ',RR5
      WRITE(1,10)' ENTER RR5: '
      READ(1,*) RR5
C
C  -----CALCULATE THE COEFFICIENTS OF CE
C
      X0 = RR1*RR2*RR3*RR4*RR5
      X1 = RR5*(RR3*RR4*(RR1+RR2)+RR1*RR2*(RR3+RR4))
      X1 = X1 + RR1*RR2*RR3*RR4
      X2 = RR5*(RR1+RR2)*(RR3+RR4)+RR3*RR4*(RR1+RR2+RR5)
      X2 = X2 + RR1*RR2*(RR3+RR4+RR5)
      X3 = RR5*(RR1+RR2+RR3+RR4)+RR3*RR4+RR1*RR2
      X3 = X3 + (RR1+RR2)*(RR3+RR4)
      X4 = RR1+RR2+RR3+RR4+RR5
C
C  -----WRITE OUT THE CHARACTERISTIC EQUATION
C
  400 WRITE(1,*)'  '
      WRITE(1,*)'  '
      WRITE(1,*)' THE CHARACTERISTIC EQUATION IS:'
      WRITE(1,*)'  '
      WRITE(1,*)'    S**5 :      1.0'
      WRITE(1,*)'    S**4 : ',X4
      WRITE(1,*)'    S**3 : ',X3
      WRITE(1,*)'    S**2 : ',X2
      WRITE(1,*)'    S**1 : ',X1
      WRITE(1,*)'    S**0 : ',X0
C
      DO 550  I = 1,5
        DO 500 KK = 1,5
          H(I,KK) = 0.0
  500   CONTINUE
  550 CONTINUE
C
C  -----COMPUTE THE H[5,5] MATRIX
C
      H(1,5) = P1
      H(2,2) = P1*A4
      H(2,4) = -P1*A8
      H(2,5) = P1*(A1+A7)
      H(3,1) = H(2,2)
      H(3,2) = P1*(A4*A7-A3*A8)
      H(3,3) = H(2,4)
      H(3,4) = P1*(A4*A5-A1*A8)
```

```
      H(3,5) = P1*(A1*A7-A3*A5-A6)
      H(4,1) = H(3,2)
      H(4,2) = P1*(A2*A8-A4*A6)
      H(4,3) = H(3,4)
      H(4,5) = P1*(A2*A5-A1*A6)
      H(5,1) = H(4,2)
C
C  -----COMPUTE THE F[5] MATRIX
C
      F(1) = X4 - (A1+A7+A10)
      F(2) = X3 - (A1*(A7+A10)+A7*A10-A3*A5-A6+A4*A9 )
      F(3) = X2 - (A10*(A1*A7-A3*A5-A6)+A9*(A4*A7-A3*A8)
     +             +(A2*A5-A1*A6))
      F(4) = X1 - (A10*(A2*A5-A1*A6)+A9*(A2*A8-A4*A6))
      F(5) = X0
C
C  -----COMPUTE THE GAIN MATRIX K[5]
C
      CALL LINEQ(K,F,H,WORK,5,6,IER)
C
      WRITE(1,*)'  '
      WRITE(1,*)' THE K MATRIX OF GAINS IS:'
   20 FORMAT(/,2X,5(F10.5,2X))
      WRITE(1,20) (K(I),I=1,5)
C
      WRITE(1,*)'  '
      WRITE(1,*)' IER = ',IER
C
      GOTO 50
C
C  -----CALCULATE THE OBSERVER GAINS
C
  590 WRITE(1,*)'  '
      WRITE(1,*)' ENTER THE OBSERVER POLES . . .'
      WRITE(1,*)'  '
      WRITE(1,10) ' ENTER RR1: '
      READ(1,*) RR1
      WRITE(1,10) ' ENTER RR2: '
      READ(1,*) RR2
      WRITE(1,10) ' ENTER RR3: '
      READ(1,*) RR3
      WRITE(1,10) ' ENTER RR4: '
      READ(1,*) RR4
C
C  -----CALCU_ATE THE COEFFICIENTS OF CE
C
      X0 = RR1*RR2*RR3*RR4
      X1 = RR1*RR2*(RR3+RR4) + RR3*RR4*(RR1+RR2)
      X2 = RR1*RR2 + RR3*RR4 + (RR1+RR2)*(RR3+RR4)
      X3 = RR1 + RR2 + RR3 + RR4
C
C  -----CALCULATE THE S[4,4] MATRIX
C
      DO 650  I = 1,4
```

```
        DO 600  KK = 1,4
          S(I,KK) = 0.0
  600   CONTINUE
  650 CONTINUE
C
      S(1,1) = 1.0
      S(2,1) = A11 + A7
      S(2,2) = 1.0
      S(3,1) = A11*A7-A3*A12-A6
      S(3,2) = A7
      S(3,3) = -A2
      S(3,4) = A3
      S(4,1) = A2*A12-A11*A6
      S(4,2) = -A6
      S(4,3) = A3*A6-A2*A7
      S(4,4) = -A2
C
C  -----CALCULATE THE T[4] MATRIX
C
      T(1) = X3 - S(2,1)
      T(2) = X2 - S(3,1)
      T(3) = X1 - S(4,1)
      T(4) = X0
C
C  -----COMPUTE THE GAIN MATRIX G[4]
C
      CALL LINEQ(GA,T,S,WERK,4,5,IER)
C
      WRITE(1,*)'  '
      WRITE(1,*)' THE G MATRIX OF GAINS IS:'
   30 FORMAT(/,2X,4(F11.3,2X))
      WRITE(1,30)  (GA(I),I=1,4)
      WRITE(1,*)'  '
      WRITE(1,*)' IER = ',IER
C
      GOTO 50
C
      END
```

APPENDIX G

The observability of the linear fourth order system model approximation derived in Appendix K, can be determined by examining the rank of the observability matrix $\underline{O}$. The transpose of the system matrix $\underline{A}$ is given by:

$$\underline{A}^T = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -A11 & 0 & A12 \\ 0 & -A2 & 0 & A5 \\ 0 & A3 & 1 & -A6 \end{bmatrix} \tag{G1}$$

The transpose of the output matrix $\underline{D}$ is given by:

$$\underline{D}^T = \begin{bmatrix} D1 & 0 \\ 0 & 0 \\ 0 & D2 \\ 0 & 0 \end{bmatrix} \tag{G2}$$

The nonzero elements of the output matrix have been left as variables for analytical generalization. By constructing the observability matrix from these two matrices [14], the observability can be determined as a function of the two variables D1, and D2.

From the definition of the observability matrix, $\underline{O}$ can be written as:

$$\underline{O} = \begin{bmatrix} D1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & D1 & 0 & -A11D1 & A12D2 & (A11^2+A3A12)D1 & -(A12(A11+A6))D2 \\ 0 & D2 & 0 & 0 & -A2D1 & A5D2 & (A2A11+A5A3)D1 & -(A2A12+A5A6)D2 \\ 0 & 0 & 0 & D2 & A3D1 & -A6D2 & -(A3(A11+A6)+A2)D1 & (A3A12+A6^2+A5)D2 \end{bmatrix}$$

The rank of this matrix indicates the observability of the system. It is easy to see that the rank is dependent on the variables D1, and D2. There are three cases of interest to investigate in this system. The first case will assume both the cart and pendulum positions will be measured, or equivalently:

$$D1 = D2 = 1 \tag{G3}$$

For this case, the rank of the observability matrix is:

$$\text{rank}[\underline{Q}] = 4 \tag{G4}$$

When both position states are measured, the observability matrix has full rank, therefore the system is completely observable.

In the second case, only the cart position will be measured and the pendulum position will be neglected:

$$D1 = 1 \tag{G5}$$

$$D2 = 0 \tag{G6}$$

Substituting these values into the observability matrix, the rank is:

$$\text{rank}[\underline{Q}] = 4 \tag{G7}$$

The observability matrix has full rank and thus is completely observable when only the cart position is available. This indicates that an observer may be designed to estimate the full state vector given only the cart position as an input.

The final case allows the pendulum position as an input, and the cart position is neglected, or equivalently:

$$D1 = 0 \tag{G8}$$

$$D2 = 1 \tag{G9}$$

For this case, the rank of the observability matrix is:

$$\text{rank}[\underline{O}] = 3 \tag{G10}$$

Since the observability matrix is of rank 3, there is one unobservable state. This system is not completely observable when only the pendulum position is available for input.

APPENDIX H

All computer simulations of the system response were performed on the PRIME 750 computer system in the MSU Case Center for CAD, utilizing the DIFFEQ software package [2]. This package integrates a given set of coupled first order differential equations and provides a graphical output compatible with Tektronix terminals. The first order equations are coded into a function subroutine supplied by the user.

There were six models studied in the simulations for stability analysis, and therefore six function subroutines. These subroutines hold both fifth order and the equivalent fourth order models for the nonlinear as well as linearized systems, under linear state feedback control. Two of the subroutines contain the equivalent fourth order models, linear and nonlinear, of the system as well as the modified observer equations. The subroutine names and their relevant functions are:

LIN4.74 . . . linear 4th order approximation w/direct LSF

LIN5.74 . . . linear 5th order model w/direct LSF

NON4.74 . . . nonlinear 4th order approximation w/direct LSF

NON5.74 . . . nonlinear 5th order model w/direct LSF

LOBS4.74 . . . linear 4th order approx. w/modified observer

NOBS4.74 . . . nonlinear 4th order approx. w/modified observer

```
CCCCCCC              SUBROUTINE DESCRIPTION                      CCCCC
C                                                                    C
C     SUBROUTINE LIN4.74                                             C
C                                                                    C
C     THIS SUBROUTINE CONTAINS THE FOURTH ORDER LINEARIZED           C
C     INVERTED PENDULUM MODEL, W/LSF IMPLEMENTED.                    C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CVCCCCCCCC           VARIABLE IDENTIFICATION                 CCCCCCCC
C                                                                    C
C     X              THE INDEPENDENT VARIABLE.                       C
C     Y              VECTOR CONTAINING SOLUTIONS.                    C
C     DERY           VECTOR CONTAINING USER DEFINED DERIVATIVES.     C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CSCCCCCCC            ENTRY AND STORAGE BLOCK               BLOCK 0000
C                                                                    C
      SUBROUTINE FCT(X,Y,DERY)
C
      REAL*4         DERY   (20),K1,K2,K3,K4,K5,U,
     +               Y      (20),
     +               PA     (20)
C
C---COMMON FOR FCT PARAMETERS
C
      COMMON/FCTCOM/PA
C
C---DEFINE PARAMETERS OF A, B, AND K MATRICES
C
      DATA A1,A2,A3/2.683,8.276,9.936E-4/
      DATA A4,A5,A6/20.91,2.16,32.63/
      DATA A7,A8,A9/3.92E-3,16.87,406.0/
      DATA A10,B1/2804.7,156.08/
      DATA A11,A12,A13,A14/5.711,4.607,1.164,0.939/
      DATA C1,C2,C3,C4/0.544,1.33E-2,0.439,5.26E-2/
C
      K1 = PA(1)
      K2 = PA(2)
      K3 = PA(3)
      K4 = PA(4)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CPCCCCCCCC           PROCESS BLOCK                         BLOCK 0200
C                                                                    C
C---CALCULATE CONTROL INPUT
C
      U = (K1*Y(1) + K2*Y(2) + K3*Y(3) + K4*Y(4))
      IF (ABS(U).GT.15.0) U = INT(ABS(U)/U)*15.0
      Y(5) = U
C
C---SET DERIVATIVES
```

```
C
      DERY(1) = Y(2)
      DERY(2) = - A11*Y(2) - A2*Y(3) + A3*Y(4) + A13*U
      DERY(3) = Y(4)
      DERY(4) = A12*Y(2) + A6*Y(3) - A7*Y(4) - A14*U
      IF (Y(2) .GT. 1.E-5) THEN
        DERY(2) = DERY(2) - C1*Y(2)/ABS(Y(2))
        DERY(4) = DERY(4) + C3*Y(2)/ABS(Y(2))
      ENDIF
      IF (Y(4) .GT. 1.E-5) THEN
        DERY(2) = DERY(2) + C2*Y(4)/ABS(Y(4))
        DERY(4) = DERY(4) - C4*Y(4)/ABS(Y(4))
      ENDIF
C
      Y(6) = Y(1)*12.0
      Y(7) = Y(3)*180.0/3.1415926
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      RETURN
      END
```

```
CCCCCCC              SUBROUTINE DESCRIPTION                    CCCCC
C                                                                  C
C     SUBROUTINE LIN5.74                                           C
C                                                                  C
C     THIS SUBROUTINE CONTAINS THE FOURTH ORDER LINEARIZED         C
C     INVERTED PENDULUM MODEL COUPLED TO THE SECOND ORDER          C
C     LINEARIZED MOTOR MODEL.  THE COUPLING PRODUCES A             C
C     FIFTH ORDER SYSTEM MODEL OVERALL.                            C
C                                                                  C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                  C
CVCCCCCCCC           VARIABLE IDENTIFICATION               CCCCCCCC
C                                                                  C
C     X              THE INDEPENDENT VARIABLE.                     C
C     Y              VECTOR CONTAINING SOLUTIONS.                  C
C     DERY           VECTOR CONTAINING USER DEFINED DERIVATIVES.   C
C                                                                  C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                  C
CSCCCCCCC            ENTRY AND STORAGE BLOCK             BLOCK 0000
C                                                                  C
      SUBROUTINE FCT(X,Y,DERY)
C
      REAL*4         DERY   (20),K1,K2,K3,K4,K5,U,
     +               Y      (20),
     +               PA     (20)
C
C---COMMON FOR FCT PARAMETERS
C
      COMMON/FCTCOM/PA
C
C---DEFINE PARAMETERS OF A, B, AND K MATRICES
C
      DATA A1,A2,A3/2.683,8.276,9.936E-4/
      DATA A4,A5,A6/20.91,2.16,32.63/
      DATA A7,A8,A9/3.92E-3,16.87,406.0/
      DATA A10,B1/2804.7,156.08/
      DATA A11,A12,A13,A14/5.711,4.607,1.164,0.939/
      DATA C1,C2,C3,C4/0.544,1.33E-2,0.439,5.26E-2/
C
      K1 = PA(1)
      K2 = PA(2)
      K3 = PA(3)
      K4 = PA(4)
      K5 = PA(5)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                  C
CPCCCCCCCC           PROCESS BLOCK                       BLOCK 0200
C                                                                  C
C---CALCULATE CONTROL INPUT
C
      U = (K1*Y(1) + K2*Y(2) + K3*Y(3) + K4*Y(4) + K5*Y(5))
      IF (ABS(U).GT.15.0) U = INT(ABS(U)/U)*15.0
```

```
      Y(6) = U
C
C---SET DERIVATIVES
C
      DERY(1) = Y(2)
      DERY(2) = - A1*Y(2) - A2*Y(3) + A3*Y(4) + A4*Y(5)
      DERY(3) = Y(4)
      DERY(4) = A5*Y(2) + A6*Y(3) - A7*Y(4) - A8*Y(5)
      DERY(5) = - A9*Y(2) - A10*Y(5) + B1*U
      IF (Y(2) .GT. 1.E-5) THEN
        DERY(2) = DERY(2) - C1*Y(2)/ABS(Y(2))
        DERY(4) = DERY(4) + C3*Y(2)/ABS(Y(2))
      ENDIF
      IF (Y(4) .GT. 1.E-5) THEN
        DERY(2) = DERY(2) + C2*Y(4)/ABS(Y(4))
        DERY(4) = DERY(4) - C4*Y(4)/ABS(Y(4))
      ENDIF
C
      Y(7) = Y(1)*12.0
      Y(8) = Y(3)*180.0/3.1415926
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      RETURN
      END
```

```
CCCCCCC              SUBROUTINE DESCRIPTION                      CCCCC
C                                                                    C
C     SUBROUTINE NON4.74                                             C
C                                                                    C
C     THIS SUBROUTINE HOLDS THE NONLINEAR SYSTEM EQUATIONS           C
C     WITH THE FULL STATE-FEEDBACK CONTROLLER IMPLEMENTED.           C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CVCCCCCCCC            VARIABLE IDENTIFICATION                CCCCCCCC
C                                                                    C
C     X               THE INDEPENDENT VARIABLE.                      C
C     Y               VECTOR CONTAINING SOLUTIONS.                   C
C     DERY            VECTOR CONTAINING USER DEFINED DERIVATIVES.    C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CSCCCCCCC            ENTRY AND STORAGE BLOCK               BLOCK 0000
C                                                                    C
      SUBROUTINE FCT(X,Y,DERY)
C
      REAL*4         DERY   (20),K1,K2,K3,K4,K,N,J,
     +               Y      (20),MC,MP,B1,B2,B3,L,R,
     +               PA     (20)
C
C---COMMON FOR FCT PARAMETERS
C
      COMMON/FCTCOM/PA
C
C     DEFINE PARAMETERS OF EQNS. AND K MATRIX
C
      DATA MP,MC/0.0236,0.0385/
      DATA D,G/1.2396,32.174/
      DATA B1,B2,B3/1.13E-4,0.025,2.16E-4/
      DATA K,N/0.0813,32.0/
      DATA J,L,R/5.2E-5,6.407E-3,17.97/
      DATA C1,C2,C3,C4/0.544,1.33E-2,0.439,5.26E-2/
C
      K1 = PA(1)
      K2 = PA(2)
      K3 = PA(3)
      K4 = PA(4)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CPCCCCCCCC            PROCESS BLOCK                         BLOCK 0200
C                                                                    C
C---CALCULATE CONTROL INPUT                                          C
C
      U = (K1*Y(1) + K2*Y(2) + K3*Y(3) + K4*Y(4))
      IF (ABS(U).GT.15.0) U = INT(ABS(U)/U)*15.0
      Y(5) = U
C
C---SET DERIVATIVES
```

```
C
      DERY(1) = Y(2)
      TOP = (K * N/(1.356 * R)) * (U - K * N * Y(2))
      TOP = TOP + MP * D * Y(4) * Y(4) * SIN(Y(3))
      TOP = TOP - MP * G * SIN(Y(3)) * COS(Y(3))
      TOP = TOP + (B1/D) * Y(4) * COS(Y(3))
      TOP = TOP - (N * N * B3 + B2) * Y(2)
      IF (Y(4) .GE. 1.E-5) THEN
        TOP = TOP + C2*(MC+N*N*J)*COS(Y(3))*Y(4)/ABS(Y(4))
      ENDIF
      IF (Y(2) .GE. 1.E-5) THEN
        TOP = TOP - C1*(MC+N*N*J)*Y(2)/ABS(Y(2))
      ENDIF
      DERY(2) = TOP/(MC + MP*(SIN(Y(3))**2) + N*N*J)
      DERY(3) = Y(4)
      TOP = G*SIN(Y(3)) - DERY(2)*COS(Y(3)) - Y(4)*B1/(MP*D)
      IF (Y(4) .GE. 1.E-5) THEN
        TOP = TOP - C4*D*((MC+N*N*J)/(MC+MP+N*N*J))*Y(4)/ABS(Y(4))
      ENDIF
      DERY(4) = TOP/D
C
      Y(6) = Y(1)*12.0
      Y(7) = Y(3)*180.0/3.1415926
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      RETURN
      END
```

```
CCCCCCC              SUBROUTINE DESCRIPTION                      CCCCC
C                                                                    C
C     SUBROUTINE NON5.74                                             C
C                                                                    C
C     THIS SUBROUTINE HOLDS THE NONLINEAR SYSTEM EQUATIONS           C
C     WITH THE FULL STATE-FEEDBACK CONTROLLER IMPLEMENTED.           C
C     DAMPING IS INCLUDED IN THIS STATE MODEL.  THE MOTOR            C
C     IS COUPLED TO THIS SYSTEM MODEL ALSO.                          C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CVCCCCCCCC           VARIABLE IDENTIFICATION                CCCCCCCC
C                                                                    C
C     X              THE INDEPENDENT VARIABLE.                       C
C     Y              VECTOR CONTAINING SOLUTIONS.                    C
C     DERY           VECTOR CONTAINING USER DEFINED DERIVATIVES.     C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CSCCCCCCC            ENTRY AND STORAGE BLOCK              BLOCK 0000
C                                                                    C
      SUBROUTINE FCT(X,Y,DERY)
C
      REAL*4         DERY   (20),K1,K2,K3,K4,K,N,J,
     +               Y      (20),MC,MP,B1,B2,B3,L,R,
     +               PA     (20)
C
C---COMMON FOR FCT PARAMETERS
C
      COMMON/FCTCOM/PA
C
C     DEFINE PARAMETERS OF EQNS. AND K MATRIX
C
      DATA MP,MC/0.0236,0.0385/
      DATA D,G/1.2396,32.174/
      DATA B1,B2,B3/1.13E-4,0.025,2.16E-4/
      DATA K,N/0.0813,32.0/
      DATA J,L,R/5.2E-5,6.407E-3,17.97/
      DATA C1,C2,C3,C4/0.544,1.33E-2,0.439,5.26E-2/
C
      K1 = PA(1)
      K2 = PA(2)
      K3 = PA(3)
      K4 = PA(4)
      K5 = PA(5)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CPCCCCCCCC           PROCESS BLOCK                        BLOCK 0200
C                                                                    C
C---CALCULATE CONTROL INPUT                                          C
C
      U = (K1*Y(1) + K2*Y(2) + K3*Y(3) + K4*Y(4) + K5*Y(5))
      IF (ABS(U).GT.15.0) U = INT(ABS(U)/U)*15.0
```

```
      Y(6) = U
C
C---SET DERIVATIVES
C
      DERY(1) = Y(2)
      TOP = K*N*Y(5)/1.356 + MP*D*Y(4)*Y(4)*SIN(Y(3))
      TOP = TOP - MP * G * SIN(Y(3)) * COS(Y(3))
      TOP = TOP + (B1/D) * Y(4) * COS(Y(3))
      TOP = TOP - (N * N * B3 + B2) * Y(2)
      IF (Y(4) .GE. 1.E-5) THEN
        TOP = TOP + C2*(MC+N*N*J)*COS(Y(3))*Y(4)/ABS(Y(4))
      ENDIF
      IF (Y(2) .GE. 1.E-5) THEN
        TOP = TOP - C1*(MC+N*N*J)*Y(2)/ABS(Y(2))
      ENDIF
      DERY(2) = TOP/(MC + MP*(SIN(Y(3))**2) + N*N*J)
      DERY(3) = Y(4)
      TOP = G*SIN(Y(3)) - DERY(2)*COS(Y(3)) - Y(4)*B1/(MP*D)
      IF (Y(4) .GE. 1.E-5) THEN
        TOP = TOP - C4*D*((MC+N*N*J)/(MC+MP+N*N*J))*Y(4)/ABS(Y(4))
      ENDIF
      DERY(4) = TOP/D
      DERY(5) = (U - R*Y(5) - K*N*Y(2))/L
C
      Y(7) = Y(1)*12.0
      Y(8) = Y(3)*180.0/3.1415926
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      RETURN
      END
```

```
CCCCCCC              SUBROUTINE DESCRIPTION                    CCCCC
C                                                                  C
C     SUBROUTINE LOBS4.74                                          C
C                                                                  C
C     THIS SUBROUTINE CONTAINS THE FOURTH ORDER LINEARIZED         C
C     INVERTED PENDULUM MODEL, AND THE OBSERVER MODEL.             C
C                                                                  C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                  C
CVCCCCCCCC           VARIABLE IDENTIFICATION                CCCCCCCC
C                                                                  C
C     X              THE INDEPENDENT VARIABLE.                     C
C     Y              VECTOR CONTAINING SOLUTIONS.                  C
C     DERY           VECTOR CONTAINING USER DEFINED DERIVATIVES.   C
C                                                                  C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                  C
CSCCCCCCC            ENTRY AND STORAGE BLOCK              BLOCK 0000
C                                                                  C
      SUBROUTINE FCT(X,Y,DERY)
C
      REAL*4         DERY   (20),K1,K2,K3,K4,K5,U,
     +               Y      (20),
     +               PA     (20)
C
C---COMMON FOR FCT PARAMETERS
C
      COMMON/FCTCOM/PA
C
C---DEFINE PARAMETERS OF A, B, AND K MATRICES
C
      DATA A1,A2,A3/2.683,8.276,9.936E-4/
      DATA A4,A5,A6/20.91,2.16,32.63/
      DATA A7,A8,A9/3.92E-3,16.87,406.0/
      DATA A10,B1/2804.7,156.08/
      DATA A11,A12,A13,A14/5.711,4.607,1.164,0.939/
      DATA C1,C2,C3,C4/0.544,1.33E-2,0.439,5.26E-2/
C
      K1 = PA(1)
      K2 = PA(2)
      K3 = PA(3)
      K4 = PA(4)
C
      G1 = PA(5)
      G2 = PA(6)
      G3 = PA(7)
      G4 = PA(8)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                  C
CPCCCCCCCC           PROCESS BLOCK                        BLOCK 0200
C                                                                  C
C---CALCULATE CONTROL INPUT
C
```

```
      U = (K1*Y(5) + K2*Y(6) + K3*Y(7) + K4*Y(8))
      IF (ABS(U).GT.15.0) U = INT(ABS(U)/U)*15.0
      Y(9) = U
C
C---SET DERIVATIVES
C
      DERY(1) = Y(2)
      DERY(2) = - A11*Y(2) - A2*Y(3) + A3*Y(4) + A13*U
      DERY(3) = Y(4)
      DERY(4) = A12*Y(2) + A6*Y(3) - A7*Y(4) - A14*U
C
      IF (Y(2) .GT. 1.E-5) THEN
        DERY(2) = DERY(2) - C1*Y(2)/ABS(Y(2))
        DERY(4) = DERY(4) + C3*Y(2)/ABS(Y(2))
      ENDIF
      IF (Y(4) .GT. 1.E-5) THEN
        DERY(2) = DERY(2) + C2*Y(4)/ABS(Y(4))
        DERY(4) = DERY(4) - C4*Y(4)/ABS(Y(4))
      ENDIF
C
      Y(10) = Y(1)*12.0
      Y(11) = Y(3)*180.0/3.1415926
C
C---OBSERVER SYSTEM
C
      ERR = Y(1) - Y(5)
C
      DERY(5) = Y(6) + G1 * ERR
      DERY(6) = - A11*Y(6) - A2*Y(7) + A3*Y(8) + A13*U + G2 * ERR
      DERY(7) = Y(8) + G3 * ERR
      DERY(8) = A12*Y(6) + A6*Y(7) - A7*Y(8) - A14*U + G4 * ERR
C
      IF (Y(6) .GT. 1.E-5) THEN
        DERY(6) = DERY(6) - C1*Y(6)/ABS(Y(6))
        DERY(8) = DERY(8) + C3*Y(6)/ABS(Y(6))
      ENDIF
      IF (Y(8) .GT. 1.E-5) THEN
        DERY(6) = DERY(6) + C2*Y(8)/ABS(Y(8))
        DERY(8) = DERY(8) - C4*Y(8)/ABS(Y(8))
      ENDIF
C
      Y(12) = Y(5)*12.0
      Y(13) = Y(7)*180.0/3.1415926
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      RETURN
      END
```

```
CCCCCCC              SUBROUTINE DESCRIPTION                      CCCCC
C                                                                    C
C     SUBROUTINE NOBS4.74                                            C
C                                                                    C
C     THIS SUBROUTINE HOLDS THE NONLINEAR SYSTEM EQUATIONS           C
C     AND THE LINEAR OBSERVER MODEL.                                 C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CVCCCCCCCC            VARIABLE IDENTIFICATION                CCCCCCCC
C                                                                    C
C     X               THE INDEPENDENT VARIABLE.                      C
C     Y               VECTOR CONTAINING SOLUTIONS.                   C
C     DERY            VECTOR CONTAINING USER DEFINED DERIVATIVES.    C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
CSCCCCCCC            ENTRY AND STORAGE BLOCK              BLOCK 0000
C                                                                    C
      SUBROUTINE FCT(X,Y,DERY)
C
      REAL*4         DERY   (20),K1,K2,K3,K4,K,N,J,
     +               Y      (20),MC,MP,B1,B2,B3,L,R,P1,
     +               PA     (20),A2,A3,A6,A7,A11,A12,A13,A14
C
C---COMMON FOR FCT PARAMETERS
C
      COMMON/FCTCOM/PA
C
C     DEFINE PARAMETERS OF EQNS. AND K MATRIX
C
      DATA MP,MC/0.0236,0.0385/
      DATA D,G/1.2396,32.174/
      DATA B1,B2,B3/1.13E-4,0.025,2.16E-4/
      DATA K,N/0.0813,32.0/
      DATA J,L,R/5.2E-5,6.407E-3,17.97/
C
      DATA A1,A2,A3/2.683,8.2769.936E-4/
      DATA A4,A5,A6/20.91,2.16,32.63/
      DATA A7,A8,A9/3.92E-3,16.87,406.0/
      DATA A10,P1/2804.7,156.08/
      DATA A11,A12,A13,A14/5.711,4.607,1.164,0.939/
      DATA C1,C2,C3,C4/0.544,1.33E-2,0.439,5.26E-2/
C
      K1 = PA(1)
      K2 = PA(2)
      K3 = PA(3)
      K4 = PA(4)
C
      G1 = PA(5)
      G2 = PA(6)
      G3 = PA(7)
      G4 = PA(8)
C
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
CPCCCCCCCC            PROCESS BLOCK                          BLOCK 0200
C                                                                       C
C---CALCULATE CONTROL INPUT                                             C
C
      IF (PA(20) .EQ. 0.0) THEN
        U = (K1*Y(1) + K2*Y(2) + K3*Y(3) + K4*Y(4))
      ELSE
        U = (K1*Y(5) + K2*Y(6) + K3*Y(7) + K4*Y(8))
      ENDIF
C
      IF (PA(19) .EQ. 1.0) THEN
        IF (ABS(U).GT.15.0) U = INT(ABS(U)/U)*15.0
      ENDIF
C
      Y(9) = U
C
C---SET DERIVATIVES
C
      DERY(1) = Y(2)
      TOP = (K * N/(1.356 * R)) * (U - K * N * Y(2))
      TOP = TOP + MP * D * Y(4) * Y(4) * SIN(Y(3))
      TOP = TOP - MP * G * SIN(Y(3)) * COS(Y(3))
      TOP = TOP + (B1/D) * Y(4) * COS(Y(3))
      TOP = TOP - (N * N * B3 + B2) * Y(2)
      IF (Y(4) .GE. 1.E-5) THEN
        TOP = TOP + C2*(MC+N*N*J)*COS(Y(3))*Y(4)/ABS(Y(4))
      ENDIF
      IF (Y(2) .GE. 1.E-5) THEN
        TOP = TOP - C1*(MC+N*N*J)*Y(2)/ABS(Y(2))
      ENDIF
      DERY(2) = TOP/(MC + MP*(SIN(Y(3))**2) + N*N*J)
      DERY(3) = Y(4)
      TOP = G*SIN(Y(3)) - DERY(2)*COS(Y(3)) - Y(4)*B1/(MP*D)
      IF (Y(4) .GE. 1.E-5) THEN
        TOP = TOP - C4*D*((MC+N*N*J)/(MC+MP+N*N*J))*Y(4)/ABS(Y(4))
      ENDIF
      DERY(4) = TOP/D
C
      Y(10) = Y(1)*12.0
      Y(11) = Y(3)*180.0/3.1415926
C
C---OBSERVER SYSTEM
C
      ERR = Y(1) - Y(5)
C
      DERY(5) = Y(6) + G1 * ERR
      DERY(6) = - A11*Y(6) - A2*Y(7) + A3*Y(8) + A13*U + G2 * ERR
      DERY(7) = Y(8) + G3 * ERR
      DERY(8) = A12*Y(6) + A6*Y(7) - A7*Y(8) - A14*U + G4 * ERR
C
      IF (PA(11) .EQ. 1.0) THEN
        IF (Y(6) .GT. 1.E-5) THEN
```

```
            DERY(6) = DERY(6) - C1*Y(6)/ABS(Y(6))
            DERY(8) = DERY(8) + C3*Y(6)/ABS(Y(6))
          ENDIF
          IF (Y(8) .GT. 1.E-5) THEN
            DERY(6) = DERY(6) + C2*Y(8)/ABS(Y(8))
            DERY(8) = DERY(8) - C4*Y(8)/ABS(Y(8))
          ENDIF
        ENDIF
C
        Y(12) = Y(5)*12.0
        Y(13) = Y(7)*180.0/3.1415926
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        RETURN
        END
```

APPENDIX I

As with any real control system, an accurate parameter study must be completed in order to determine the various control gains which will be implemented. In this particular system, several sub-systems could be identified and the parameters determined separately for each.

DC Servo Motor

The drive motor used in the control system was a 12 volt D.C. Specialty servo motor. There are six parameters required to define the linear motor model given in Appendix A. Of these, the most difficult parameter to determine experimentally is the armature inertia, therefore the manufacturer specification has been used.

J_m = 0.005 oz-in-s^2 = 2.6E-5 ft-lb-s^2

On the electrical side of the model, the armature inductance is in series with the armature resistance. The inductance of a series RL circuit can be measured directly with an impedance bridge. The HP 4260-A Universal Bridge was used to determine this parameter value as:

L = 6.41E-3 Henry

The gyrator constant couples the mechanical and electrical sides of the motor. This parameter can easily be determined experimentally using one of the constitutive relations for a gyrator:

$$V_b = k_E \omega_m \qquad (I1)$$

By applying a known and constant armature speed (ω_m), a back emf (V_b) is generated. For a series of fixed input speeds, a set of voltages can be read. When this data is plotted, the slope of the line is the gyrator constant (k_E).

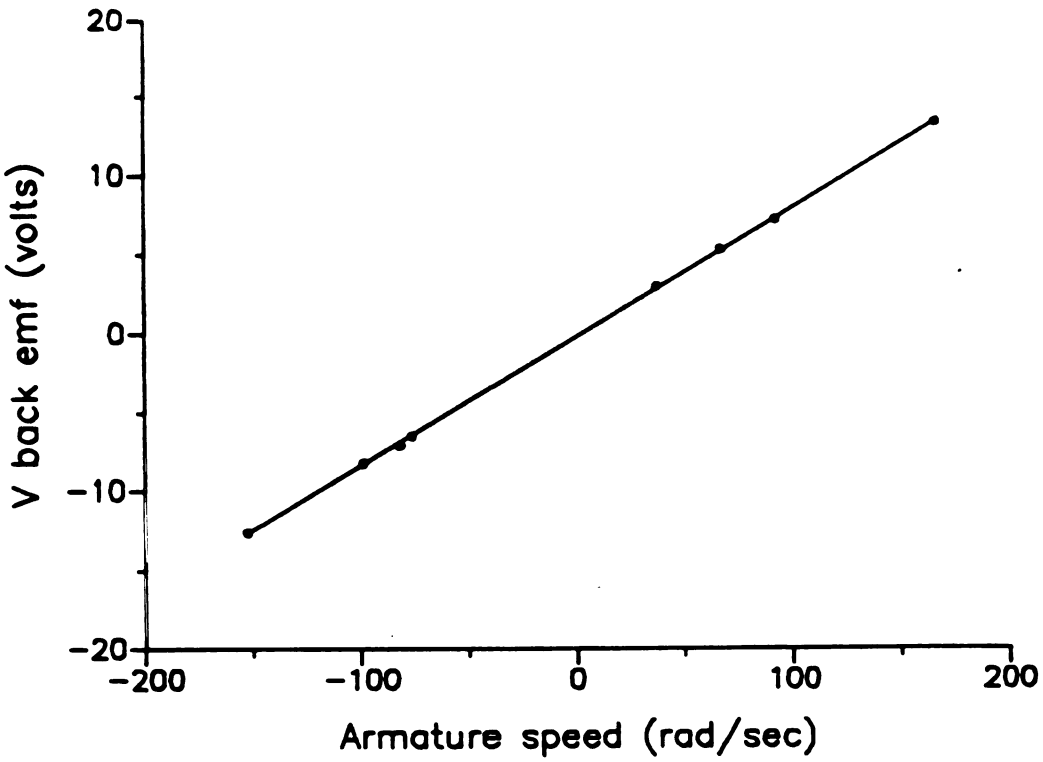


Figure I1: Determination of Gyrator Constant

For this DC servo, the linear constitutive relation defined in (I1) correctly models the physics of the motor as the plotted data of Figure I1 indicates. The slope of the line, i.e. the electrical gyrator constant, is thus:

$k_E = 0.0813$ V-s/rad

In standard bond graph use, only one constant is required for a gyrator. Due to the particular choice of units here, another gyrator constant must be defined such that power will be conserved through the gyrator. A torque constant which accounts for this choice of units, is related to the electrical constant by:

$$k_\tau = k_E / 1.356 \quad \text{(ft-lb/amp)} \tag{I2}$$

The two damping parameters, electrical resistance and mechanical viscous damping, are not easily measured. Looking at the two equations of motion for the linear motor, a simple procedure to experimentally determine these parameters can be obtained [6]. The two equations are very similar in form: one representing the electrical side, and the other representing the mechanical side.

$$V = L \frac{di}{dt} + R\, i + k_E\, \omega_m \tag{I3}$$

$$\tau = -J_m \frac{d\omega_m}{dt} - b_m\, \omega_m + k_\tau\, i \tag{I4}$$

Applying a voltage to the motor under no-load conditions, the armature rotational speed and current will reach a steady state. Thus both derivatives in (I3) and (I4), and the torque input will be zero, leaving two simple linear equations. The system states ω_m and i can be measured for a range of input voltages. For each of the test points, values for R and b_m can then be computed from equations (I3) and (I4).

The linear motor model breaks down in representing the actual motor with the given test data. Instead of constant damping values, the parameters are actually functions of ω_m, as shown in Figure I2.

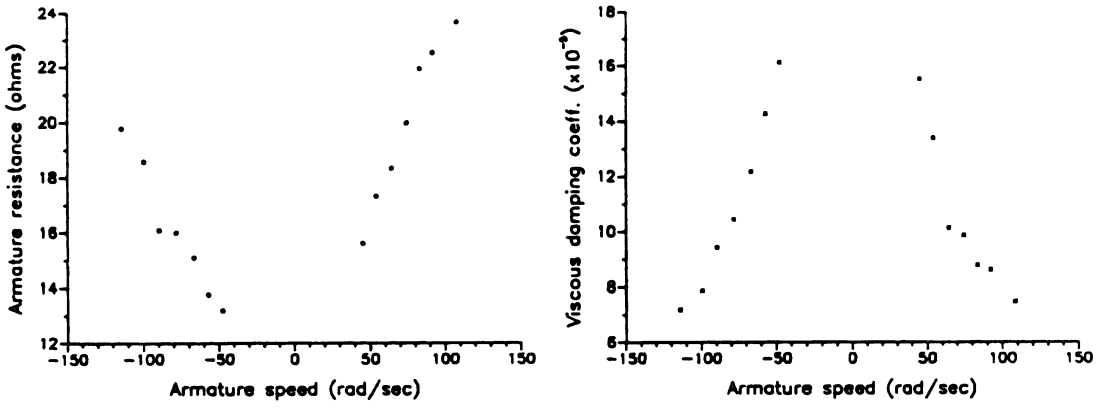


Figure I2: Nonlinear damping parameters of DC motor

To use the linear model, constant values for the damping parameters are required. A simple arithmetic mean was chosen, which effectively provides a linearization about some nominal ω_m. The computed values are given by:

$R = 17.97\ \Omega$

$b_m = 10.81E\text{-}5$ ft-lb-s

Drive Train

The motor armature shaft is connected through a gear train to the belt drive attached to the cart. It is assumed that there are no dynamics between the armature motion and the cart motion. The cart motion is rigidly connected to the armature motion. Therefore a simple linear transformer between armature position and cart position is

needed. This parameter is measured to be:

$n = 32.0$ rad/ft

The gear train also has an inertia and mechanical damping associated with it. Due to the rigid coupling assumption, this inertia and damping can be lumped into the motor parameters. The equivalent inertia of the motor armature, and the equivalent damping are:

$J = 5.2E\text{-}5$ ft-lb-s^2

$b_3 = 2.16E\text{-}4$ ft-lb-s

This set of equivalent parameters will be used for the motor parameters thus eliminating the direct inclusion of the gear train.

Pendulum

The pendulum has three parameters associated with it, they are the equivalent mass, equivalent length, and eqivalent viscous damping. Since the model used here assumes a massless rod, with no loss of generality, the equivalent mass is just the total mass of the pendulum. The equivalent length is the distance from the rotational axis to the pendulum center of gravity. These values are determined to be:

$m = 0.76$ lbm. $= 0.0236$ slugs

$d = 14.87$ in. $= 1.24$ feet

$b_1 = 1.13E\text{-}4$ ft-lb-s

Cart

The cart also has three parameters associated with it. The first, and easiest to determine, is the cart mass. Once again, due to the rigid connection between the cart and armature via the gear train, the

connecting drive belt will be lumped into the cart mass. The equivalent cart mass is thus measured to be:

$$M = 1.32 \text{ lbm} = 0.041 \text{ slugs}$$

The most difficult cart parameters to determine were the equivalent damping coefficients. A linear combination of viscous and coulomb damping effects have been assumed in the model formulation. These parameter values are determined experimentally utilizing a conservation of energy principle [12].

Taking the first nonlinear system equation (A13), from Appendix A, and fixing the pendulum such that:

$$\theta = \dot{\theta} = \ddot{\theta} = 0 \tag{I5}$$

the second nonlinear system equation (A14), can be eliminated, and (A13) can then be written as:

$$m'\ddot{x} + b'\dot{x} + f_x \text{sgn}(\dot{x}) = F(t) \tag{I6}$$

where the primed parameters are defined to be:

$$m' = M + m + n^2 J \tag{I7}$$

$$b' = n^2 b_3 + b_2 + k_E k_T n^2 / R \tag{I8}$$

Equation (I6) represents a simple cart mass being forced against viscous and coulomb friction, and includes the reduction of order approximation derived in Appendix K. By driving this system with a sinusoidal force, a nearly sinusoidal cart output motion is expected. This experiment can actually be performed on the real system. Multiplying the four force terms of (I6) by dx, and integrating the whole equation over one cycle,

an energy equation is obtained:

$$\int_c (m'\ddot{x})dx + \int_c (b'\dot{x})dx + \int_c f_x \mathrm{sgn}(\dot{x})dx = \int_c F_o \sin(\omega_f t)dx \qquad (I9)$$

The result (I9), is a representation of the flow of energy through one cycle of the forcing input.

Due to the presence of the nonlinear coulomb damping term, the system output x(t) will not be purely sinusoidal for a sinusoidal forcing input. Since the coulomb damping magnitude is assumed small, the output will approximately be sinusoidal, and can be represented by:

$$x(t) = X \sin(\omega_f t - \phi) \qquad (I10)$$

Evaluating the terms in the energy equation (I9) by direct substitution of (I10), the energy terms become:

$$\int_c (m'\ddot{x})dx = 0 \qquad (I11)$$

(The net momentum in one cycle is zero.)

$$\int_c (b'\dot{x})dx = \pi b' \omega_f X^2 \qquad (I12)$$

$$\int_c f_x \mathrm{sgn}(\dot{x})dx = 4 f_x X \qquad (I13)$$

The final term in the energy equation is the total energy input to the system. The value of this integral can be obtained by plotting the input force F(t) versus the actual output x(t). A hysteresis loop is obtained, and the area inside the loop is, by definition, the total work input in one cycle.

$$\int_c F_o \sin(\omega_f t)dx = W \qquad (I14)$$

Experimentally fixing the pendulum such that (I5) is satisfied, and plotting the sinusoidal input force versus the resulting cart position, nine hysteresis loops were obtained. Several sample hysteresis loops are shown in Figure I3. These loops were digitized on the PRIMOS Digitizing facility in the Case Center [2]. The areas inside the digitized loops were then computed with the program AREA, developed to compute areas inside digitized closed coordinate boundaries. The program AREA is included at the end of this appendix.

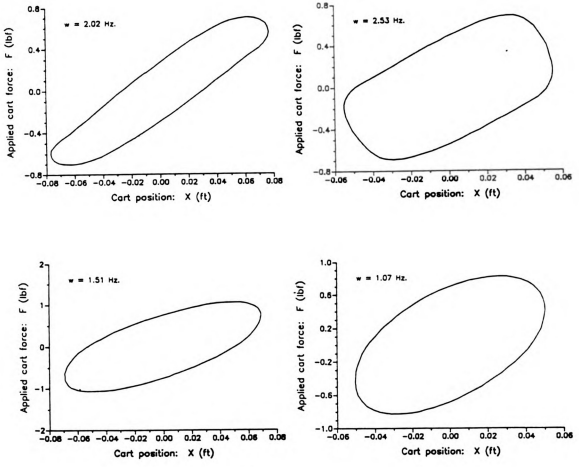


Figure I3: Experimental Hysteresis Loops

Rewriting the energy equation, using the results of the substitutions (I11) through (I14), a linear relation can be obtained where the slope is the viscous damping coefficient, and the y-intercept is the coulomb friction force:

$$\frac{W_i}{4X_i} = \left[\frac{\pi \omega_{f_i} X_i^2}{4X_i}\right] b' + f_x \qquad \text{(I15)}$$

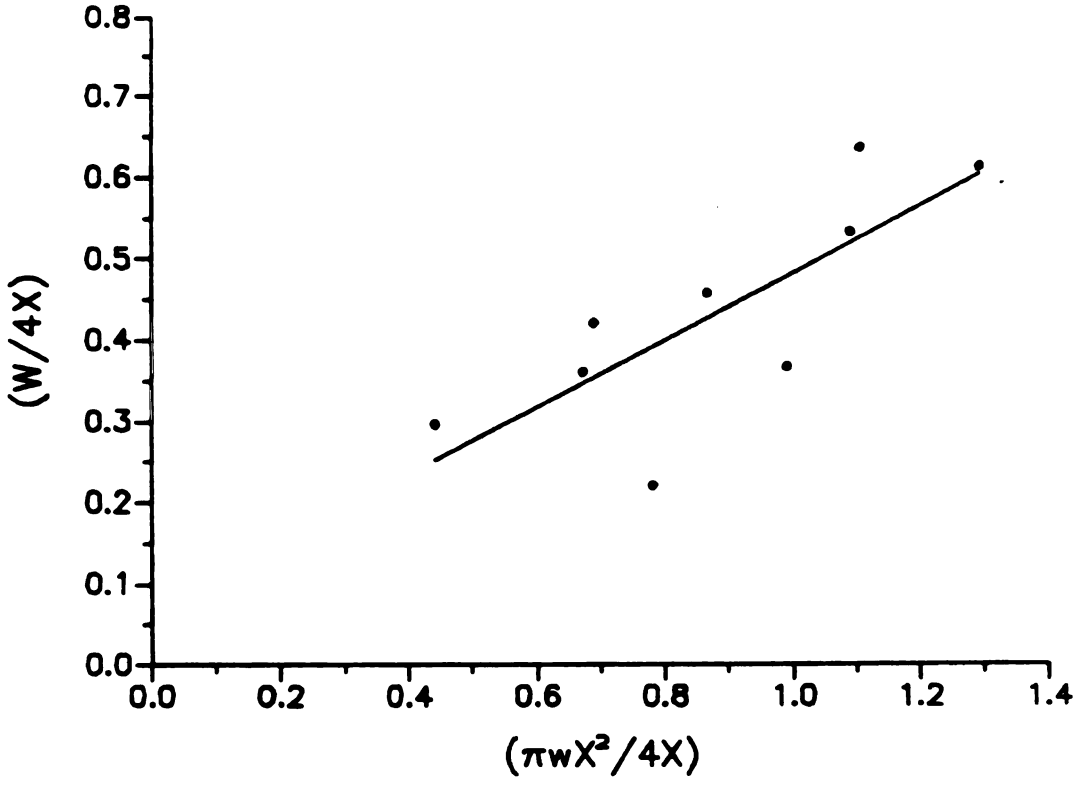


Figure I4: Damping terms from the energy equation

Figure I4 shows a plot of the data in the form of (I15), from nine hysteresis loop experiments at various frequencies and amplitudes. If the assumption of a linear combination of viscous and coulomb damping terms is valid, then the data should lie along a straight line. Since

this is not the case with the real data, the damping present in the cart is more complex than this model will predict. The slope and intercept from a least squares regression are:

$b' = 0.4119$ lb-s/ft

$f_x = 0.0698$ lb

Substituting the viscous damping result into the defining relation, equation (I8), the equivalent viscous cart damping coefficient is:

$b_2 = 0.025$ lb-s/ft

Sensor Calibration

The experimental system has two position sensors to measure the pendulum and cart positions. These sensors have a voltage output which is functionally related to the respective positions. To use the voltage information, the functional dependance must be determined. The voltage signals from the sensors are read by an A/D converter which digitizes the voltages. For simplicity, this calibration was made between the physical coordinates and the discrete digital representation.

The cart sensor is a ten turn linear rotational potentiometer connected to the drive train shaft. This allows a measurable cart motion of about three feet.

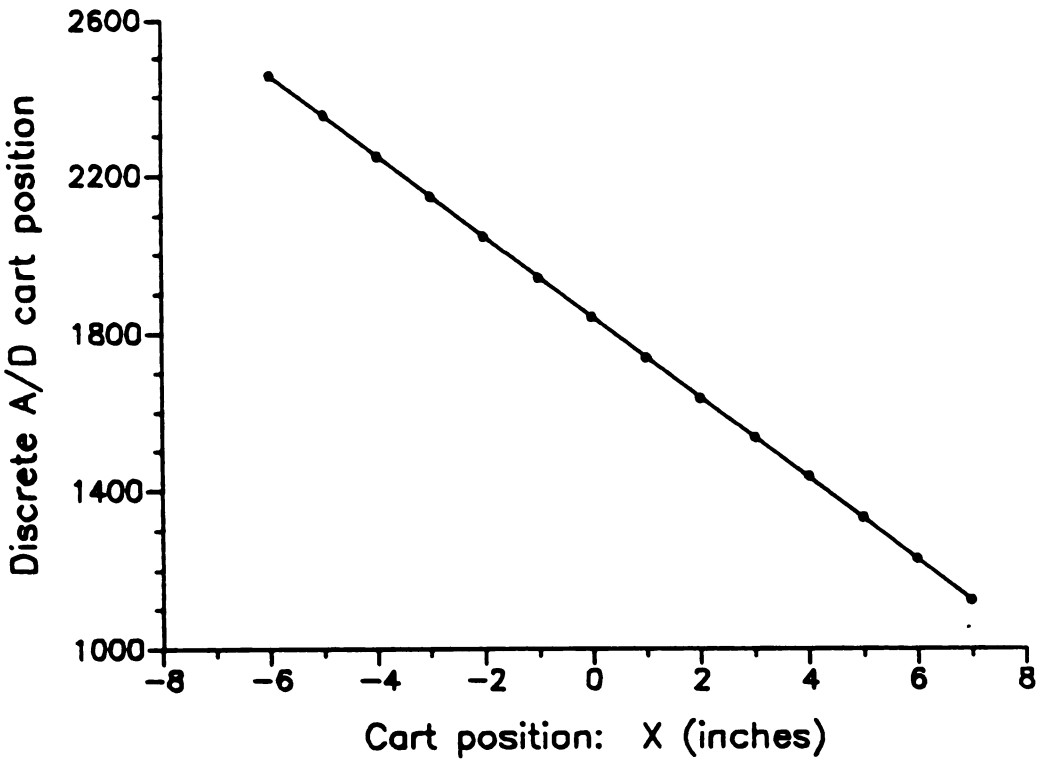


Figure I5: Calibration of linear cart potentiometer sensor

The slope of the line in Figure I5 is the calibration constant between the physical cart coordinate (x) and the digital computer state representation.

$$C_x = -1225.2 \text{ ft}^{-1}$$

A Hall Effect transducer is used for the pendulum position sensor. This device does not have a linear transfer function, as seen in Figure I6. In order to use the transducer, the function must be linearized around the operating point (θ=0). Since a stable pendulum will always be near the origin, this linearized approximation will be valid.

C_θ = -1241.3 rad^{-1}

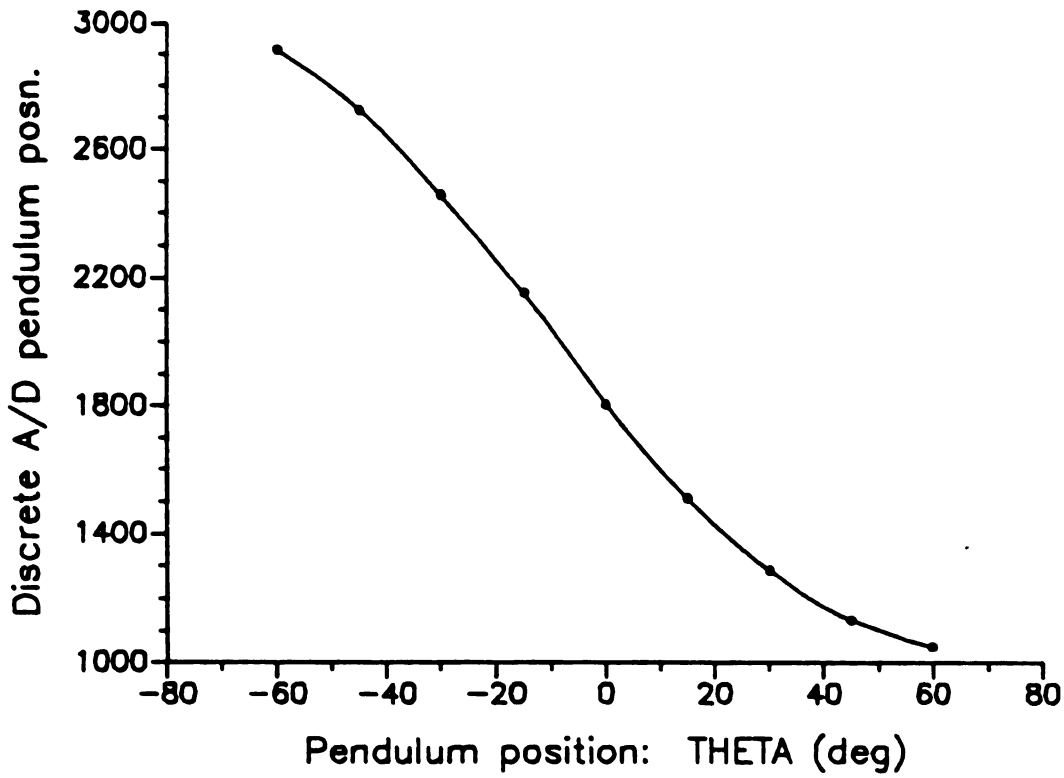


Figure I6: Calibration of Hall Effect transducer

Power Amplifier

The computer applied control voltages to the motor through a D/A output channel. The +10.0 to -10.0 volt D/A output signal is amplified through a power amplifier which has a constant linear gain of 1.58. It is this power amplifier which supplies the required current to drive the DC servo motor. The D/A converter has a linear gain of 204.75 V^{-1} between actual and discrete voltages. For simplicity, these two gains were lumped together to provide a linear relationship between the digital control voltage, and the actual voltage applied to the motor.

This linear constant is:

$$C_V = (4095./20.)/1.58$$

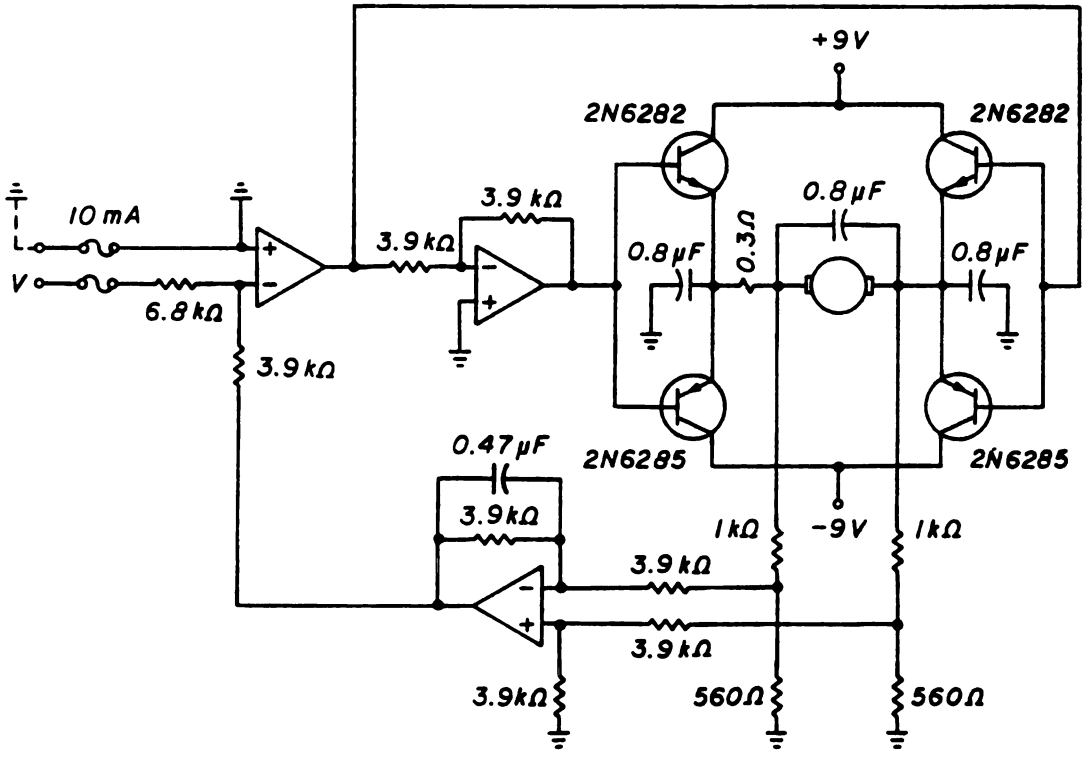


Figure I7: Power amplifier schematic diagram

```
      PROGRAM AREA
C
C  THIS PROGRAM READS IN A DATA FILE OF (X,Y) COORDINATE
C  PAIRS WHICH DEFINE A TWO DIMENSIONAL SURFACE.  AFTER
C  ALL THE POINTS ARE READ IN, THE AREA CONTAINED BY THE
C  CORDINATE BOUNDARY WILL BE COMPUTED.
C
C  -----DECLARE ALL VARIABLES
C
      REAL X(100),Y(100)
C
      INTEGER EROR,I,N,J
C
      CHARACTER FILNAM*32
C
C  -----GET THE FILENAME CONTAINING THE DATA
C
   10 WRITE(1,'(/,5X,A,$)') 'DATA FILENAME:  '
      READ(1,'(A)',ERR=10) FILNAM
C
      OPEN(5,FILE=FILNAM,ERR=10,STATUS='OLD')
C
C  -----READ IN ALL THE DATA
C
      I = 1
      EROR = 0
C
   20 READ(5,*,END=40,ERR=30) X(I),Y(I),J
C
      I = I + 1
      GOTO 20
C
   30 EROR = EROR + 1
      GOTO 20
C
   40 CLOSE(5)
C
      IF (EROR .GT. 0) THEN
        WRITE(1,*)'  '
        WRITE(1,*) 'THERE WERE',EROR,' READ ERRORS IN ',FILNAM
      ENDIF
C
      N = I - 1
C
      SUM = 0
C
C  -----CALCULATE THE AREA
C
      DO 50  I = 1,N
        IF (I .EQ. 1) THEN
          SUM = SUM + (X(N)-X(2))*Y(1)
        ELSE IF (I .EQ. N) THEN
          SUM = SUM + (X(N-1)-X(1))*Y(N)
        ELSE
```

```
            SUM = SUM + (X(I-1)-X(I+1))*Y(I)
          ENDIF
   50 CONTINUE
C
      SUM = ABS(SUM/2.0)
C
C  -----WRITE THE RESULTS TO THE TERMINAL
C
      WRITE(1,'(/,5X,A,F14.6)')'AREA = ',SUM
      WRITE(1,*)'  '
C
      END
```

APPENDIX J

The following control program was written in FORTRAN IV for use on a single user DEC minicomputer under the RT-11 operating system. Though this program is implemented on the DEC LSI-11/23+, it will run on most DEC computers under RT-11. Standard DEC vector addresses for the various system devices have been used [5], and due to the modular structure of the code, minor changes can easily be made for nonstandard vector addresses.

```
      PROGRAM PEND
C
C  This is the main program which accesses all the control
C  subroutines for the inverted pendulum.
C
C      [STEVE C. SOUTHWARD, B.S.M.E]
C
C  -----Overall program Logistics
C
C      PEND [Main program: PEND.FOR]
C
C            IPOSN [Subroutine: IPOSN.FOR]
C                  SAMPLE [Subroutine: SAMPLE.MAC]
C
C            CGAINS [Subroutine: CGAINS.FOR]
C
C            CONTRL [Subroutine: CONTRL.FOR]
C                  DISCRT [Subroutine: DISCRT.FOR]
C                  RUNLSF [Interrupt Service: RUNLSF.FOR]
C
C            PAUSE [Subroutine: CLOCK.FOR]
C
C            SETICK [Interrupt Service: CLOCK.FOR]
C
C  -----Format Note:   name [type: storage filename]
C
      REAL G(24),F(24)
C
      LOGICAL*1 IDATA
```

```
C
      DIMENSION IDLIST(21)
C
      EXTERNAL SETICK
C
      COMMON /ADDR1/ IKSR,IKDR,IPCH,ICCH,ITCK
      COMMON /ADDR2/ IPSR,IPDR,IBPR,IADVEC,IADERR
      COMMON /ADDR3/ ICSR,IADR,IDAR,IADBUF
      COMMON /MISC2/ F,G,IRCZ,IRPZ
      COMMON /TIME/  I,IEND
C
C  -----Initialize the variables
C
      DATA IRCZ,IRPZ/0,0/
C
C  -----These control gains place the poles at {-3,-4,-5,-6
C
      F(1) = 11.854                     !K1 /
      F(2) = 16.174                     !K2  LSF controller
      F(3) = 175.722                    !K3  gains
      F(4) = 33.0895                    !K4 /
C
C  -----These gains place the observer poles at {-8,-8,-9,-9}
C
      F(5) = 28.285                     !G1 /
      F(6) = 303.973                    !G2  Observer
      F(7) = -425.307                   !G3  gains
      F(8) = -2331.515                  !G4 /
C
      F(9) = 1.0/(1.0 + 3.0)            !position
      F(10)= 1.0 - F(9)                 !filter
C
      F(11)= 5.7107                     !A11
      F(12)= 8.276                      !A2
      F(13)= 99.357E-4                  !A3  /
      F(14)= 1.16369                    !A13  System
      F(15)= 4.6069                     !A12  parameters
      F(16)= 32.631                     !A6  /
      F(17)= 3.91757E-3                 !A7
      F(18)= 0.93876                    !A14
C
      F(19)= 1.0/(1.0 + 10.0)           !derivative
      F(20)= 1.0 - F(19)                !filter
C
      F(21)= 0.76078                    !C1 /
      F(22)= 1.75854E-2                 !C2  Coulomb damping
      F(23)= 0.61373                    !C3  coefficients
      F(24)= 6.93375E-2                 !C4 /
C
C  -----Set up address registers
C
      IPSW = "102                       !processor status word
      IKSR = "177560                    !keyboard status register
      IKDR = IKSR + 2                   !keyboard data register
```

```
      IPSR = IKSR + 4                  !printer status register
      IPDR = IKSR + 6                  !printer data register
      ICSR = "170420                   !real time clock status reg.
      IBPR = ICSR + 2                  !clock buffer preset reg.
      ICKV = "440                      !clock vector address
      IDAR = "170440                   !D/A base output addr.: ch.0
      IADR = "170400                   !A/D control status register
      IADBUF = IADR + 2                !A/D buffer register
      IADVEC = "400                    !A/D done interrupt vector
      IADERR = IADVEC + 4              !A/D error interrupt vector
      IBELL = "7                       !bell tone
      IZERO = "4000                    !zero value for D/A converter
C
C  -----Protect System from crash
C
      IDLIST(1) = IKSR                 !keyboard status
      IDLIST(2) = IPEEK(IKSR)
      IDLIST(3) = IPSR                 !printer status
      IDLIST(4) = IPEEK(IPSR)
      IDLIST(5) = IDAR                 !zero D/A channel 0
      IDLIST(6) = IZERO
      IDLIST(7) = IDAR + 2             !zero D/A channel 1
      IDLIST(8) = IZERO
      IDLIST(9) = IDAR + 4             !zero D/A channel 2
      IDLIST(10) = IZERO
      IDLIST(11) = IDAR + 6            !zero D/A channel 3
      IDLIST(12) = IZERO
      IDLIST(13) = IADR                !A/D status
      IDLIST(14) = 0
      IDLIST(15) = IADVEC              !A/D done interrupt vector
      IDLIST(16) = IPEEK(IADVEC)
      IDLIST(17) = ICSR                !clock status
      IDLIST(18) = IPEEK(ICSR)
      IDLIST(19) = IPSW                !processor status
      IDLIST(20) = IPEEK(IPSW)
      IDLIST(21) = 0                   !end of list marker
C
      CALL DEVICE(IDLIST)
C
C  -----Format Statements
C
  500 FORMAT(/,/,/,22X,'INVERTED PENDULUM CONTROL ROUTINES',
     +  /,22X,'--------------------------------')
  510 FORMAT(/,/,22X,'Enter the PENDULUM A/D channel: ',$)
  520 FORMAT(/,22X,'Enter the CART A/D channel: ',$)
  530 FORMAT(/,/,22X,'Enter the D/A output channel: ',$)
  540 FORMAT(I2)
  550 FORMAT(/,/,28X,'-----MAIN MENU-----',
     +  /,/,18X,'(I)nitialize the zero reference positions',
     +  /,/,18X,'(G)ain settings for LSF control β Observer',
     +  /,/,18X,'(S)tatus of Zero positions',
     +  /,/,18X,'(R)un the active controller',
     +  /,/,18X,'(Q)uit the program',
     +  /,/,/,22X,'Choose one of the above. . . ',$)
```

```
  560 FORMAT(/,/,22X,'STATUS of Zero References. . .',
     +  /,22X,'----------------------------')
  570 FORMAT(/,22X,'CART zero reference:    ',I6)
  580 FORMAT(/,22X,'PEND zero reference:    ',I6)
  590 FORMAT(/,/,22X,'Exiting Control Routine. . .',5(/))
  600 FORMAT(/,/,18X,'Zero reference positions NOT initialized')
C
C  -----Calibrate the PAUSE subroutine
C
      IEND = 10000                        !set initial value
      IPR = 7                             !highest priority interrupt
C
   10 I = INTSET(ICKV,IPR,1,SETICK)       !attach to RTC vector
      IF (I .EQ. 0) GOTO 20
C
      WRITE(7,*)'INTSET error -- RTC vector, CODE = ',I
      GOTO 10
C
   20 IRATE = "111                        !set up clock to generate
      ICOUNT = -16667                     !an interrupt after 1 tick
C
      CALL IPOKE(IBPR,ICOUNT)
      CALL IPOKE(ICSR,IRATE)              !start the clock
C
      DO 30  I = 1,10000                  !do nothing loop while
   30 CONTINUE                            !waiting for one tick
C
      ITCK = IEND + 1
D     WRITE(7,*)'ITICK = ',ITCK
C
C  -----Set zero (0) volts on the D/A output
C
      DO 40  I = 0,6,2                    !
        CALL IPOKE(IDAR+I,IZERO)          !zero out all 4 channels
   40 CONTINUE                            !
C
C  -----Get the cart and pendulum A/D channels
C
      WRITE(7,500)                        !initial header
   50 WRITE(7,510)                        !get PEND channel first
      READ(5,540,ERR=50) IPCH
      IF ((IPCH.GT.15).OR.(IPCH.LT.0)) GOTO 50
C
   60 WRITE(7,520)                        !get CART channel second
      READ(5,540,ERR=60) ICCH
      IF ((ICCH.GT.15).OR.(ICCH.LT.0)) GOTO 60
      IF (ICCH.EQ.IPCH) GOTO 60
C
   70 WRITE(7,530)                        !get D/A output channel
      READ(5,540,ERR=70) IDACH
      IF ((IDACH.LT.0).OR.(IDACH.GT.3)) GOTO 70
      IDAR = IDAR + IDACH * 2
C
C  -----Print out the options menu
```

```
C
   80 WRITE(7,550)                        !print the menu options
      CALL PAUSE(50,ITCK)                 !wait for printing to finish
C
C  -----Disable the keyboard
C
      I = IPEEK(IKSR) .AND. "177477       !clear keyboard interrupt
      CALL IPOKE(IKSR,I)
C
C  -----Wait for keyboard input
C
   90 CALL IPOKE(IPDR,IBELL)              !ring the bell
C
  100 I = IPEEK(IKSR) .AND. "200          !test bit 7 of IKSR
      IF (I .EQ. 0) GOTO 100
C
      IDATA = IPEEK(IKDR)                 !a key has been pressed
C
C  -----Check for a valid character
C
      IF ((IDATA.NE.'I').AND.(IDATA.NE.'G').AND.(IDATA.NE.'S').AND.
     +    (IDATA.NE.'R').AND.(IDATA.NE.'Q')) GOTO 90
C
      CALL IPOKE(IPDR,IDATA)              !print the proper key
C
C  -----Enable the keyboard
C
      I = IPEEK(IKSR) .OR. "100           !set bit 6 of IKSR
      CALL IPOKE(IKSR,I)
      WRITE(7,*)'  '
C
C  -----Go to the proper place
C
      IF (IDATA .EQ. 'I') GOTO 110
      IF (IDATA .EQ. 'G') GOTO 120
      IF (IDATA .EQ. 'S') GOTO 130
      IF (IDATA .EQ. 'R') GOTO 140
      IF (IDATA .EQ. 'Q') GOTO 150
C
C  -----INITIALIZE the zero reference positions
C
  110 CALL IPOSN
C
      GOTO 80                             !return to menu
C
C  -----GAIN settings for LSF control law
C
  120 CALL CGAINS
C
      GOTO 80                             !return to menu
C
C  -----STATUS of Zero positions
C
  130 WRITE(7,560)                        !initial header
```

```
      WRITE(7,570) IRCZ
      WRITE(7,580) IRPZ
C
      GOTO 80                              !return to menu
C
C  -----RUN the active control routine
C
  140 IF ((IRCZ.NE.0).AND.(IRPZ.NE.0)) GOTO 145
C
      WRITE(7,600)                         !zero references
      GOTO 80                              !not initialized
C
  145 CALL CONTRL
C
      GOTO 80                              !return to menu
C
C  -----QUIT the program
C
  150 WRITE(7,590)                         !exit message
      CALL PAUSE(60,ITCK)
      CALL EXIT
C
      END
```

```
      SUBROUTINE IPOSN
C
C  This subroutine allows the user to define the initial
C  reference positions of the cart and pendulum.
C
      REAL G(24),F(24)
C
      LOGICAL*1 IDATA
C
      COMMON /ADDR1/ IKSR,IKDR,IPCH,ICCH,ITCK
      COMMON /MISC2/ F,G,IRCZ,IRPZ
C
C  -----Format Statements
C
  100 FORMAT('+',26X,I4,12X,I4)
  110 FORMAT(/,/,/,20X,'INITIALIZE the Reference Positions',
     +  /,20X,'--------------------------------',
     +  /,/,/,15X,'Move the cart and pendulum to their respective',
     +  /,/,15X,'reference positions to define the ZEROES.',
     +  /,/,/,15X,'Press ''P'' to define PENDULUM reference . . .',
     +  /,/,15X,'Press ''C'' to define the CART reference . . .',
     +  /,/,15X,'Press ''0'' to RESTART . . .',
     +  /,/,/,/,27X,'CART',10X,'PENDULUM',/,/)
C
C  -----Main Routine
C
      WRITE(7,110)                      !print the instructions
      CALL PAUSE(100,ITCK)              !wait for CRT
C
C  -----Turn off the keyboard
C
      I = IPEEK(IKSR) .AND. "177477     !clear keyboard interrupt
      CALL IPOKE(IKSR,I)
C
    5 ICDUN = 0                         !reset the flags
      IPDUN = 0
C
C  -----Start Sampling
C
   10 IF (ICDUN .EQ. 1) GOTO 20
C
      CALL SAMPLE(ICCH,ICVAL,IFLAG)     !get CART A/D sample
      IF (IFLAG.NE.0) GOTO 40
      IRCZ = ICVAL
C
   20 IF (IPDUN .EQ. 1) GOTO 30
C
      CALL SAMPLE(IPCH,IPVAL,IFLAG)     !get PEND. A/D sample
      IF (IFLAG.NE.0) GOTO 40
      IRPZ = IPVAL
C
   30 WRITE(7,100) IRCZ,IRPZ
C
      CALL PAUSE(7,ITCK)                !wait for 7 ticks
```

```
C
C  -----Check for keyboard input
C
      I = IPEEK(IKSR) .AND. "200        !test bit 7 of IKSR
      IF (I .EQ. 0) GOTO 10
C
      IDATA = IPEEK(IKDR)               !a key has been pressed
C
      IF (IDATA .EQ. 'C') ICDUN = 1
      IF (IDATA .EQ. 'P') IPDUN = 1
      IF (IDATA .EQ. '0') GOTO 5
C
      IF ((ICDUN.EQ.1).AND.(IPDUN.EQ.1)) GOTO 40
      GOTO 10
C
C  -----Turn on the keyboard
C
   40 I = IPEEK(IKSR) .OR. "100
      CALL IPOKE(IKSR,I)
C
      RETURN
C
      END
```

```
        .TITLE SAMPLE
;
; This is a MACRO-11, FORTRAN compatible subroutine called with
;
;       CALL SAMPLE(ICHAN,IVAL,IFLG)
;
; This subroutine gets a sample from the specified A/D channel
; ICHAN, and the sampled data is passed back in IVAL. The flag
; IFLG is returned to the main program with the status:
;           /   0 = NO ERRORS DETECTED
;   IFLG = -    1 = CHANNEL NUMBER OUT OF RANGE (0-15)
;           /   2 = A/D SAMPLING ERROR
;
        .GLOBL SAMPLE           ;This makes SAMPLE a global symbol
        LC=.                    ;Find current assembly location
        .=1000+LC               ;Start assembling at location 1000
        CSR=170400              ;Define A/D status register address
        DBR=CSR+2               ;Define A/D buffer register address
;
SAMPLE:
;   R5 points to the first address of a table of arguments
;       J(R5) is the first data (= # of arguments passed)
;       J2(R5) is the first argument passed
;
        MOV     J2(R5),ICHAN    ;Put first argument in ICHAN
        MOV     J2(R5),TEMP     ;Put first argument in TEMP also
        BIC     #177760,ICHAN   ;Clear bits 4 to 15
        CLR     FLG             ;Clear the FLG
        CMP     TEMP,ICHAN      ;Check for ICHAN > 15
        BNE     ERR1            ;Branch to ERR1 if ICHAN > 15
START:  SWAB    ICHAN           ;Swap high byte for low byte
        MOV     ICHAN,J#CSR     ;Set A/D Control Status Register
        BIS     #1,J#CSR        ;Start the A/D conversion
;
WAIT:   BIT     #200,J#CSR      ;Test bit 7 for A/D done
        BEQ     WAIT            ;Wait if not set
;
        MOV     J#DBR,J4(R5)    ;Put sampled data into argument table
        BIT     #100000,J#CSR   ;Test bit 15 for A/D sampling error
        BNE     ERR2            ;Branch to ERR2 if bit 15 is set
DONE:   MOV     FLG,J6(R5)      ;Put flag back into argument table
        RTS     PC              ;Return from sub. to calling program
;
;       ERROR HANDLING ROUTINES
ERR1:   BIS     #1,FLG          ;Set bit 0 of FLG
        BR      START           ;Go back to start
ERR2:   BIS     #2,FLG          ;Set bit 1 of FLG
        BR      DONE            ;Go back to DONE
;
;       LOCAL VARIABLES
ICHAN:  .WORD   0
TEMP:   .WORD   0
FLG:    .WORD   0
        .END
```

```
      SUBROUTINE CGAINS
C
C  This subroutine allows the user to input the control
C  gains for the linear state feedback law, the luenberger
C  observer, or the first order filter parameters.
C
C  -----Definition of parameters F()
C
C       F(1) = LSF proportional cart gain
C       F(2) = LSF derivative cart gain
C       F(3) = LSF proportional pendulum gain
C       F(4) = LSF derivative pendulum gain
C       F(5) = observer gain #1
C       F(6) = observer gain #2
C       F(7) = observer gain #3
C       F(8) = observer gain #4
C       F(9) = filter parameter (alfa)
C       F(10)= filter parameter (1-alfa)
C       F(11)=
C         .    /
C         .      observer system parameters
C         .    /
C       F(18)=
C       F(19)= deriv. filter parameter (alfa)
C       F(20)= deriv. filter parameter (1-alfa)
C       F(21)= C1 /
C       F(22)= C2  Coulomb damping
C       F(23)= C3  coefficients
C       F(24)= C4 /
C
      REAL G(24),F(24),SN
C
      COMMON /MISC2/ F,G,IRCZ,IRPZ
C
C  -----Format Statements
C
  100 FORMAT(/,/,/,20X,'INPUT the System and Control Gains',
     +  /,20X,'-------------------------------')
  110 FORMAT(/,/,20X,'1.)  Input LSF Controller gains.',
     +  /,20X,'2.)  Input Luenberger Observer gains.',
     +  /,20X,'3.)  Input Observer System Parameters.',
     +  /,20X,'4.)  Input Filter Parameters.',
     +  /,20X,'5.)  Check Status of all gains.',
     +  /,20X,'6.)  EXIT to main routine. .')
  120 FORMAT(/,/,25X,'Your Selection:  ',$)
  130 FORMAT(I1)
  140 FORMAT(/,27X,'K(',I1,') [',A10,'] = ',$)
  150 FORMAT(/,27X,'G(',I1,') [Observer] = ',$)
  160 FORMAT(/,30X,A3,' = ',$)
  170 FORMAT(/,18X,'Enter #SAMP./',A4,'. FILTER TIME CONST.: ',$)
  180 FORMAT(/,25X,'LSF Controller Gains:',/,
     +  4(/,30X,'K(',I1,') = ',F10.4))
  190 FORMAT(/,25X,'Observer Gains:',/,
     +  4(/,30X,'G(',I1,') = ',F10.4))
```

```
  200 FORMAT(/,25X,'Observer System Parameters:',/,
     +  12(/,30X,A3,' = ',F10.4))
  210 FORMAT(/,25X,A4,'. Filter Parameters:',/,
     +  /,30X,'#Samp./TC = ',F5.1,/,
     +  /,33X,' alfa  = ',F10.4,
     +  /,33X,'1-alfa = ',F10.4)
C
C  -----Main routine
C
      WRITE(7,100)
   10 WRITE(7,110)                        !display menu options
   20 WRITE(7,120)
      READ(5,130,ERR=20) IOP
      IF ((IOP.LT.1).OR.(IOP.GT.6)) GOTO 20
C
C  -----Goto the proper place
C
      GOTO(30,40,50,70,80,90) IOP
C
C  -----Input new LSF controller gains
C
   30 WRITE(7,140) 1,'Volt/Ft   '         !proportional cart gain
      READ(5,*,ERR=30) F(1)
C
   32 WRITE(7,140) 2,'Volt-S/Ft '         !derivative cart gain
      READ(5,*,ERR=32) F(2)
C
   34 WRITE(7,140) 3,'Volt/Rad  '         !proportional pend. gain
      READ(5,*,ERR=34) F(3)
C
   36 WRITE(7,140) 4,'Volt-S/Rad'         !derivative pend. gain
      READ(5,*,ERR=36) F(4)
C
      GOTO 10                             !return to cgains menu
C
C  -----Input new Luenberger Observer gains
C
   40 WRITE(7,150) 1                      !get observer gain #1
      READ(5,*,ERR=40) F(5)
C
   42 WRITE(7,150) 2                      !get observer gain #2
      READ(5,*,ERR=42) F(6)
C
   44 WRITE(7,150) 3                      !get observer gain #3
      READ(5,*,ERR=44) F(7)
C
   46 WRITE(7,150) 4                      !get observer gain #4
      READ(5,*,ERR=46) F(8)
C
      GOTO 10                             !return to cgains menu
C
C  -----Input new observer system parameters
C
C  50 GOTO 65
```

```
   50 WRITE(7,160) 'A11'                !get A11
      READ(5,*,ERR=50) F(11)
C
   52 WRITE(7,160) 'A2 '                !get A2
      READ(5,*,ERR=52) F(12)
C
   54 WRITE(7,160) 'A3 '                !get A3
      READ(5,*,ERR=54) F(13)
C
   56 WRITE(7,160) 'A13'                !get A13
      READ(5,*,ERR=56) F(14)
C
   58 WRITE(7,160) 'A12'                !get A12
      READ(5,*,ERR=58) F(15)
C
   60 WRITE(7,160) 'A6 '                !get A6
      READ(5,*,ERR=60) F(16)
C
   62 WRITE(7,160) 'A7 '                !get A7
      READ(5,*,ERR=62) F(17)
C
   64 WRITE(7,160) 'A14'                !get A14
      READ(5,*,ERR=64) F(18)
C
   65 WRITE(7,160) 'C1 '                !get C1
      READ(5,*,ERR=65) F(21)
C
   66 WRITE(7,160) 'C2 '                !get C2
      READ(5,*,ERR=66) F(22)
C
   67 WRITE(7,160) 'C3 '                !get C3
      READ(5,*,ERR=67) F(23)
C
   68 WRITE(7,160) 'C4 '                !get C4
      READ(5,*,ERR=68) F(24)
C
      GOTO 10                           !return to cgains menu
C
C  -----Input the filter parameters
C
   70 WRITE(7,170) 'POSN'               !get position filter first
      READ(5,*,ERR=70) F(9)
C
      F(9) = 1.0 / (1.0 + F(9))
      F(10) = 1.0 - F(9)
C
   72 WRITE(7,170) 'DERI'               !get derivative filter next
      READ(5,*,ERR=72) F(19)
C
      F(19) = 1.0 / (1.0 + F(19))
      F(20) = 1.0 - F(19)
C
      GOTO 10                           !return to cgains menu
C
```

```
C  -----Check the status of all gains
C
   80 WRITE(7,180) ((I,F(I)),I=1,4)
C
      WRITE(7,190) ((I-4,F(I)),I=5,8)
C
      WRITE(7,200) 'A11',F(11),'A2 ',F(12),'A3 ',F(13),'A13',
     +   F(14),'A12',F(15),'A6 ',F(16),'A7 ',F(17),'A14',F(18),
     +   'C1 ',F(21),'C2 ',F(22),'C3 ',F(23),'C4 ',F(24)
C
      SN = (1.0 / F(9)) - 1.0
      WRITE(7,210) 'POSN',SN,F(9),F(10)
C
      SN = (1.0 / F(19)) - 1.0
      WRITE(7,210) 'DERI',SN,F(19),F(20)
C
      GOTO 10                          !return to gains menu
C
C  -----EXIT to the main program
C
   90 RETURN
C
      END
```

```
      SUBROUTINE CONTRL
C
C  This is the control subroutine which sets up the
C  clocked interrupt service control routine, and
C  starts the control action.
C
      REAL G(24),F(24),A(4,4),B(4),C(4,4),D(4)
      REAL XN,VN,TN,WN,XO,TO,VO,WO,ERR,VOLT
C
      LOGICAL*1 IDATA,TI1(8),TI2(8)
C
      EXTERNAL RUNLSF                    !to use as subroutine arg.
C
      COMMON /ADDR1/ IKSR,IKDR,IPCH,ICCH,ITCK
      COMMON /ADDR2/ IPSR,IPDR,IBPR,IADVEC,IADERR
      COMMON /ADDR3/ ICSR,IADR,IDAR,IADBUF
      COMMON /MISC1/ ICH1,ICH2,PER,IFAST
      COMMON /MISC2/ F,G,IRCZ,IRPZ
      COMMON /MISC3/ C,D,XO,VO,TO,WO,NN
      COMMON /STATE/ X,T,XN,VN,TN,WN,VOLT,ERR
C
C  -----Format Statements
C
  100 FORMAT(/,/,/,22X,'RUN the Clocked Control Routine',
     +  /,22X,'-----------------------------')
  110 FORMAT(/,/,18X,'Enter the SAMPLING FREQUENCY (Hz.): ',$)
  120 FORMAT(/,/,18X,'ERROR. . .[Sampling frequency is too high.]')
  130 FORMAT(/,/,18X,'ERROR. . .[Sampling frequency is too low.]')
  140 FORMAT(/,/,18X,'ERROR. . .[Sampling rate TOO HIGH for system.]')
  150 FORMAT(/,/,18X,'Actual sampling freq. = ',E11.4,' Hz.')
  160 FORMAT(/,18X,'Actual sample period  = ',E11.4,' Sec.',/,/)
  170 FORMAT(/,/,18X,'Press ANY KEY to STOP active control. . .')
  180 FORMAT(/,/,18X,'Are you ready to begin control [Y,N]. . . ',$)
  190 FORMAT(/,/,20X,'1.)  Derivative State Approximation',
     +  /,20X,'2.)  Continuous Luenberger Observer',
     +  /,20X,'3.)  Discrete Luenberger Observer')
  200 FORMAT(/,/,25X,'Your Selection: ',$)
  210 FORMAT(I1)
  220 FORMAT('+',20X,'Turn OFF the Line Time Clock. . .')
  230 FORMAT(/,/,18X,'ERROR. . .[Pendulum out of controllable range]')
  240 FORMAT(/,/,18X,'Is system near the origin [Y,N]. . . ',$)
C
      IBELL = "7
      IPR = 7                            !highest priority interrupt
      IFAST = 0                          !reset too-high flag
      FMAX = 200.0                       !set max. sampling frequency
      VOLT = 0                           !initial voltage
      ERR = 0                            !initial error
      NN = 1
C
      CX = -1225.2                       !sensor calibration on x
      CT = -1241.3                       !sensor calibration on theta
      CV = (4095./20.)/1.58              !voltage output calibration
C
```

```
C  -----Convert real gains to controller gains
C
      G(1)  = F(1) * CV / CX            !K1  /
      G(2)  = F(2) * CV / CX            !K2   LSF controller
      G(3)  = F(3) * CV / CT            !K3   gains
      G(4)  = F(4) * CV / CT            !K4  /
C
      G(5)  = F(5)                      !G1  /
      G(6)  = F(6)                      !G2   Observer
      G(7)  = F(7) * CT / CX            !G3   gains
      G(8)  = F(8) * CT / CX            !G4  /
C
      G(9)  = F(9)                      !position
      G(10) = F(10)                     !filter
C
      G(11) = - F(11)                   !A11
      G(12) = - F(12) * CX / CT         !A2
      G(13) =   F(13) * CX / CT         !A3  /
      G(14) =   F(14) * CX / CV         !A13  Observer system
      G(15) =   F(15) * CT / CX         !A12  parameters
      G(16) =   F(16)                   !A6  /
      G(17) = - F(17)                   !A7
      G(18) = - F(18) * CT / CV         !A14
C
      G(19) = F(19)                     !derivative
      G(20) = F(20)                     !filter
C
      G(21) =   F(21) * CX              !
      G(22) = - F(22) * CX              !coulomb damping
      G(23) = - F(23) * CT              !coefficients
      G(24) =   F(24) * CT              !
C
C  -----Set up discrete observer matrix
C
      A(1,1) = - F(5)
      A(1,2) = 1.0
      A(1,3) = 0.0
      A(1,4) = 0.0
      A(2,1) = F(14) * F(1) - F(6)
      A(2,2) = F(14) * F(2) - F(11)
      A(2,3) = (F(14) * F(3) - F(12)) * CX / CT
      A(2,4) = (F(14) * F(4) + F(13)) * CX / CT
      A(3,1) = - F(7) * CT / CX
      A(3,2) = 0.0
      A(3,3) = 0.0
      A(3,4) = 1.0
      A(4,1) = - (F(18) * F(1) + F(8)) * CT / CX
      A(4,2) = - (F(18) * F(2) - F(15)) * CT / CX
      A(4,3) = - F(18) * F(3) + F(16)
      A(4,4) = - F(18) * F(4) - F(17)
C
      B(1) = F(5)
      B(2) = F(6)
      B(3) = F(7) * CT / CX
```

```
      B(4) = F(8) * CT / CX
C
C  -----Set up clocked ISR
C
      WRITE(7,100)                        !intial header
C
      WRITE(7,190)
    5 WRITE(7,200)                        !choose state estimation
      READ(5,210,ERR=5) IOP
      IF ((IOP.NE.1).AND.(IOP.NE.2).AND.(IOP.NE.3)) GOTO 5
C
      GOTO(10,15,20) IOP
C
   10 I = INTSET(IADVEC,IPR,1,RUNLSF)  !attach to A/D done vector
      IF (I .EQ. 0) GOTO 25
C
      WRITE(7,*)'INTSET error -- A/D Vector, CODE = ',I
      GOTO 10
C
   15 I = INTSET(IADVEC,IPR,2,RUNLSF)  !attach to A/D done vector
      IF (I .EQ. 0) GOTO 25
C
      WRITE(7,*)'INTSET error -- A/D Vector, CODE = ',I
      GOTO 15
C
   20 I = INTSET(IADVEC,IPR,3,RUNLSF)  !attach to A/D done vector
      IF (I .EQ. 0) GOTO 25
C
      WRITE(7,*)'INTSET error -- A/D Vector, CODE = ',I
      GOTO 20
C
   25 I = INTSET(IADERR,IPR,4,RUNLSF)  !attach to A/D error vector
      IF (I .EQ. 0) GOTO 30
C
      WRITE(7,*)'INTSET error -- A/D Error Vector, CODE = ',I
      GOTO 25
C
C  -----Input the sampling frequency
C
   30 WRITE(7,110)                        !get the sampling frequency
      READ(5,*,ERR=30) FREQ
C
      IF ((FREQ.LE.FMAX).AND.(FREQ.GT.0.)) GOTO 35
      WRITE(7,120)                        !sampling frequency too high
      GOTO 30
C
C  -----Calculate the best base clock rate
C
   35 IR = 1                              !start at highest clock rate
C
   40 TICK = (10.0**(7-IR))/FREQ
C
      IF (TICK .LT. 32767.) GOTO 45       !integer out of range
      IR = IR + 1                         !next lower base frequency
```

```
      IF (IR .LE. 7) GOTO 40
C
      WRITE(7,130)                        !sampling frequency too low
      GOTO 30
C
C  -----Calculate the ticks for IBPR
C
   45 ITICK = IFIX(-1.0*(TICK+0.5))       !nearest integer
C
C  -----Calculate the actual sampling frequency and period
C
      FREQ = (10.0**(7-IR))/FLOAT(-ITICK)
C
      PER = 1.0/FREQ
C
      WRITE(7,150) FREQ
      WRITE(7,160) PER
C
C  -----Calculate the discrete observer matrix system
C
      IF (IOP .NE. 3) GOTO 50
C
      CALL DISCRT(4,1,A,B,C,D,PER,10)
C
D     WRITE(7,*)'  '
D       DO 48  I = 1,4
D         WRITE(7,*) (C(I,J),J=1,4),D(I)
D  48   CONTINUE
D     WRITE(7,*)'  '
C
C  -----Set up the clock status and A/D status registers
C
   50 IRATE = (IR * 8) + 3
      ICH1 = "40140 + ICCH * (2**8)       !set up initial sample
      ICH2 = "40001 + IPCH * (2**8)       !set up second sample
C
C  -----Make sure the line time clock is turned off
C
   55 CALL TIME(TI1)                      !get the first time
      CALL PAUSE(60,ITCK)                 !wait for 1 second
      CALL TIME(TI2)                      !get the second time
C
      IF ((TI1(8).EQ.TI2(8)).AND.(TI1(7).EQ.TI2(7))) GOTO 60
C
      CALL IPOKE(IPDR,IBELL)
      WRITE(7,220)                        !message to user
      GOTO 55
C
C  -----Ready to begin active control
C
   60 WRITE(7,170)                        !user instructions
      WRITE(7,180)                        !stop message
      CALL PAUSE(40,ITCK)                 !wait for printer
C
```

```
C  -----Disable keyboard and wait for input [Y,N]
C
      I = IPEEK(IKSR) .AND. "177677     !reset bit 6 of IKSR
      CALL IPOKE(IKSR,I)
C
   65 I = IPEEK(IKSR) .AND. "200        !test bit 7 of IKSR
      IF (I .EQ. 0) GOTO 65
C
      IDATA = IPEEK(IKDR)               !a key has been pressed
C
      IF ((IDATA.EQ.'Y').OR.(IDATA.EQ.'N')) CALL IPOKE(IPDR,IDATA)
      IF (IDATA .EQ. 'Y') GOTO 70
      IF (IDATA .EQ. 'N') GOTO 85
C
      CALL IPOKE(IPDR,IBELL)
      GOTO 65
C
C  -----Disable the printer
C
   70 I = IPEEK(IPSR) .AND. "177677     !reset bit 6 of IPSR
      CALL IPOKE(IPSR,I)
C
D     WRITE(7,*)'  '
D     WRITE(7,*)'  '
C
C  -----Set up initial conditions for the system
C
   72 CALL SAMPLE(IPCH,IPVAL,IFLAG)     !get pend. position
      CALL SAMPLE(ICCH,ICVAL,IFLAG)     !get cart position
C
      X = (IRCZ - ICVAL)                !initial
      T = (IRPZ - IPVAL)                !positions
C
      XN = X
      TN = T
      VN = 0                            !initial
      WN = 0                            !velocities
C
C  -----Check for pendulum too far from stable origin
C
      IF ((ABS(X/CX).LE.0.5).AND.       !six inches from origin
     +   (ABS(T/CT).LE.0.175)) GOTO 74  !ten degrees from origin
C
      IDATA = IPEEK(IKDR)               !reset the keyboard status
C
      I = IPEEK(IPSR) .OR. "100         !set bit 6 of IPSR
      CALL IPOKE(IPSR,I)
C
      WRITE(7,*)'  '
      WRITE(7,240)
      GOTO 65
C
C  -----Start clock and ISR
C
```

```
   74 CALL IPOKE(IBPR,ITICK)
      CALL IPOKE(ICSR,IRATE)           !start clock ticking
      CALL IPOKE(IADR,ICH1)            !start A/D conversion
C
      IF (IOP .EQ. 1) GOTO 80
C
C  -----Wait for system to stabilize/keyboard input
C
   75 IF (IFAST .NE. 0) GOTO 85
      I = IPEEK(IKSR) .AND. "200       !test bit 7 of IKSR
      IF (I .EQ. 0) GOTO 75
C
      IDATA = IPEEK(IKDR)              !a key has been pressed
      NN = IOP                         !system is now stabilized
C
      IF (IDATA .NE. 'G') GOTO 85      !check for a go on observer
C
C  -----Wait for keyboard input or too-fast error
C
   80 IF (IFAST .NE. 0) GOTO 85
C
D1000 FORMAT('+',4(2X,E10.3))
D     WRITE(7,1000) XN,VN,TN,WN
D     CALL PAUSE(10,ITCK)
C
      I = IPEEK(IKSR) .AND. "200       !test bit 7 of IKSR
      IF (I .EQ. 0) GOTO 80
C
C  -----Enable the keyboard and printer
C
   85 I = IPEEK(IKSR) .OR. "100        !set bit 6 of IKSR
      CALL IPOKE(IKSR,I)
C
      I = IPEEK(IPSR) .OR. "100        !set bit 6 of IPSR
      CALL IPOKE(IPSR,I)
C
C  -----Check for too-fast error
C
      IF (IFAST .EQ. 0) GOTO 95
      IF (IFAST .EQ. 2) GOTO 90
C
      WRITE(7,*)'  '
      WRITE(7,140)                     !print an error message
      GOTO 95
C
   90 WRITE(7,*)'  '
      WRITE(7,230)                     !print an error message
C
C  -----Turn off clock and A/D converter
C
   95 CALL IPOKE(IADR,0)
      CALL IPOKE(ICSR,0)
      WRITE(7,*)'  '
C
```

```
C  -----Zero output voltage
C
      IZERO = "4000
      CALL IPOKE(IDAR,IZERO)
C
      RETURN
C
      END
```

```
      SUBROUTINE DISCRT(N,M,A,B,AD,BD,T,MAXIT)
C
C  This subroutine computes the discretized matrices
C  AD, and BD from the continuous system matrices.
C
C         DX(t)  =  A * X(t) +  B * U(t)
C
C         X(k+1) = AD * X(k) + BD * U(k)
C
C  -----Declare all variables
C
      DIMENSION A(N,N),AD(N,N),B(N,M),BD(N,M),
     +   A1(10,10),A2(10,10),C(10,10)
C
      REAL A,B,AD,BD,A1,A2,C,KF,T
C
      INTEGER N,M,MAXIT,I,J,K,L
C
C  -----Initialize the matrices
C
      KF = 1.0
C
      DO 20  I = 1,N
        DO 10  J = 1,N
          IF (I .NE. J) GOTO 5
            A1(I,J) = 1.0
            AD(I,J) = 1.0
            C(I,J) = T
          GOTO 10
    5       A1(I,J) = 0.0
            AD(I,J) = 0.0
            C(I,J) = 0.0
C         ENDIF
   10   CONTINUE
   20 CONTINUE
C
C  -----Compute the matrix exponential
C
      DO 80  K = 1,MAXIT
        DO 50  I = 1,N
          DO 40  J = 1,N
            A2(I,J) = 0.0
            DO 30  L = 1,N
              A2(I,J) = A2(I,J) + A1(I,L) * A(L,J)
   30       CONTINUE
   40     CONTINUE
   50   CONTINUE
C
        KF = KF * K
C
        DO 70  I = 1,N
          DO 60  J = 1,N
            AD(I,J) = AD(I,J) + A2(I,J)*(T**K)/KF
            C(I,J) = C(I,J) + A2(I,J)*(T**(K+1))/(KF*(K+1))
```

```
            A1(I,J) = A2(I,J)
   60     CONTINUE
   70   CONTINUE
   80 CONTINUE
C
      DO 110  I = 1,N
        DO 100  J = 1,M
          BD(I,J) = 0.0
          DO 90  L = 1,N
            BD(I,J) = BD(I,J) + C(I,L) * B(L,J)
   90     CONTINUE
  100   CONTINUE
  110 CONTINUE
C
      RETURN
C
      END
```

```
      SUBROUTINE RUNLSF(ID)
C
C  This is an interrupt service routine to get clocked samples
C  of the cart and pendulum positions.  The system states are
C  calculated in the observer equations, or by derivative ap-
C  proximations.  The linear state feedback control voltage is
C  calculated and applied to the motor.
C
C  -----Definition of parameters G()
C
C       G(1) = proportional cart gain
C       G(2) = derivative cart gain
C       G(3) = proportional pendulum gain
C       G(4) = derivative pendulum gain
C       G(5) = observer gain #1
C       G(6) = observer gain #2
C       G(7) = observer gain #3
C       G(8) = observer gain #4
C       G(9) = filter parameter (alfa)
C       G(10)= filter parameter (1-alfa)
C       G(11)=
C         .    /
C         .      observer system parameters
C         .    /
C       G(18)=
C       G(19)= deriv. filter parameter (alfa)
C       G(20)= deriv. filter parameter (1-alfa)
C       G(21)=  /
C       G(22)=   coulomb damping
C       G(23)=   coefficients
C       G(24)=  /
C
      REAL G(24),F(24),VOLT,PER,ERR,X,T
      REAL DX,DV,DT,DW,XN,VN,TN,WN,TMAX,VMAX
      REAL XO,VO,TO,WO,C(4,4),D(4)
      INTEGER IA(2),IB(2),IVS,IWS,IV,IW
C
      EQUIVALENCE (VN,IA),(IA(1),IV),(WN,IB),(IB(1),IW)
C
      COMMON /ADDR3/ ICSR,IADR,IDAR,IADBUF
      COMMON /MISC1/ ICH1,ICH2,PER,IFAST
      COMMON /MISC2/ F,G,IRCZ,IRPZ
      COMMON /MISC3/ C,D,XO,VO,TO,WO,NN
      COMMON /STATE/ X,T,XN,VN,TN,WN,VOLT,ERR
C
      DATA TMAX,VMAX/541.6,2047.5/
C
C  -----Select entry:  ID = 1 . . . A/D Sample Done [Deri. approx.]
C                      ID = 2 . . . A/D Sample Done [Cont. Obs.]
C                      ID = 3 . . . A/D Sample Done [Disc. Obs.]
C                      ID = 4 . . . A/D Sample Error
C
      GOTO (10,5,5,50) ID
      WRITE(7,*)'RUNLSF entry error, ID = ',ID
```

```
      CALL EXIT
C
    5 GOTO (10,20,30) NN
C
C        <<<<< DERIVATIVE STATE APPROXIMATIONS >>>>>
C
C  -----Get a new X position for the cart
C
   10 X = G(9)*(IRCZ - IPEEK(IADBUF)) + G(10)*XN
C
      CALL IPOKE(IADR,ICH2)              !start next sample
C
C  -----Calculate a new velocity
C
      VN = G(19)*(X - XN)/PER + G(20)*VN
C
C  -----Get a new theta value for pendulum
C
      T = G(9)*(IRPZ - IPEEK(IADBUF)) + G(10)*TN
C
      IF (ABS(T) .GT. TMAX) GOTO 50      !pend. out of 25 deg. range
C
      CALL IPOKE(IADR,ICH1)              !reset A/D status register
C
C  -----Calculate a new angular velocity
C
      WN = G(19)*(T - TN)/PER + G(20)*WN
C
C  -----Shift new variables into the old variables
C
      XN = X
      TN = T
C
      XO = XN
      TO = TN
      VO = VN
      WO = WN
C
C  -----Use these approximated states for LSF control law
C
      GOTO 40
C
C       <<<<< CONTINUOUS LUENBERGER OBSERVER >>>>>
C
C  -----Get a new X position for the cart
C
   20 X = (IRCZ - IPEEK(IADBUF))
C  20 X = G(9)*(IRCZ - IPEEK(IADBUF)) + G(10)*X
C
      CALL IPOKE(IADR,ICH2)              !start next sample
C
      ERR = X - XN
C
C  -----Get the sign of the two velocities
```

```
C
      IVS = 1 + 2 * ( (IV .AND. "100000) .EQ. "100000)
      IWS = 1 + 2 * ( (IW .AND. "100000) .EQ. "100000)
C
C  -----Calculate the observer derivatives
C
      DX = VN + G(5)*ERR
      DV = G(11)*VN + G(12)*TN + G(13)*WN + G(14)*VOLT + G(6)*ERR
     +      + G(21)*IVS + G(22)*IWS
      DT = WN + G(7)*ERR
      DW = G(15)*VN + G(16)*TN + G(17)*WN + G(18)*VOLT + G(8)*ERR
     +      + G(23)*IVS + G(24)*IWS
C
C  -----Approximate states from derivatives (Euler)
C
      XN = XN + DX * PER
      VN = VN + DV * PER
      TN = TN + DT * PER
      WN = WN + DW * PER
C
C  -----Get a new Theta value for pendulum
C
      T = (IRPZ - IPEEK(IADBUF))
C
      IF (ABS(T) .GT. TMAX) GOTO 50     !pend. out of 25 deg. range
C
      CALL IPOKE(IADR,ICH1)             !reset A/D status register
C
C  -----Use these approximated states for LSF control law
C
      GOTO 40
C
C      <<<<< DISCRETE LUENBERGER OBSERVER >>>>>
C
C  -----Get a new X position for the cart
C
   30 X = G(9)*(IRCZ - IPEEK(IADBUF)) + G(10)*X
C
      CALL IPOKE(IADR,ICH2)             !start next sample
C
C  -----Calculate the observed states
C
      XN = C(1,1)*XO + C(1,2)*VO + C(1,3)*TO + C(1,4)*WO + D(1)*X
      VN = C(2,1)*XO + C(2,2)*VO + C(2,3)*TO + C(2,4)*WO + D(2)*X
      TN = C(3,1)*XO + C(3,2)*VO + C(3,3)*TO + C(3,4)*WO + D(3)*X
      WN = C(4,1)*XO + C(4,2)*VO + C(4,3)*TO + C(4,4)*WO + D(4)*X
C
C  -----Shift new variables into the old variables
C
      XO = XN
      VO = VN
      TO = TN
      WO = WN
C
```

```
C  -----Get a new Theta value for pendulum
C
      T = (IRPZ - IPEEK(IADBUF))
C
      IF (ABS(T) .GT. TMAX) GOTO 50     !pend. out of 25 deg. range
C
      CALL IPOKE(IADR,ICH1)             !reset A/D status register
C
C  -----Use these approximated states for LSF control law
C
C      <<<<< LINEAR STATE FEEDBACK >>>>>
C
C  -----Calculate the LSF control law
C
   40 VOLT = G(1)*XN + G(2)*VN + G(3)*TN + G(4)*WN
C
C  -----Check for voltage out of range
C
      IF (VOLT .LT. -VMAX) VOLT = -VMAX
      IF (VOLT .GT.  VMAX) VOLT =  VMAX
C
      IVOLT = IFIX(VOLT + 2048.0)
C
      CALL IPOKE(IDAR,IVOLT)            !apply the voltage
C
      RETURN
C
C  -----A/D Sample error
C
   50 CALL IPOKE(IADR,0)                !turn off A/D
      CALL IPOKE(ICSR,0)                !turn off RTC
C
      IFAST = 1                         !set the too-fast flag
C
      IF (ABS(T) .GT. TMAX) IFAST = 2   !pendulum out of range
C
      RETURN
C
      END
```

```
      SUBROUTINE PAUSE(NTICKS,ITCK)
C
C  This subroutine is approximately calibrated to pause
C  in ITCK intervals, as specified on input.  There are
C  60 ticks in one second.
C
      DO 20  I = 1,NTICKS
C
         DO 10  J = 1,ITCK              !one tick per outer
   10    CONTINUE                       !do loop step on (I)
C
   20 CONTINUE
C
      RETURN
C
      END

      SUBROUTINE SETICK(ID)
C
C  This is an interrupt service routine that is called
C  when the real time programmable clock has generated
C  an interrupt after one tick (1/60 sec.).
C
      COMMON /TIME/ I,IEND
C
      IF (ID .EQ. 1) GOTO 10            !check for entry error
C
      WRITE(7,*)'SETICK entry error, ID = ',ID
      CALL EXIT
C
   10 IEND = I
      I = 10000                         !reset do loop counter
C
      RETURN
C
      END
```

APPENDIX K

A fifth order linear matrix representation of the inverted pendulum system equations was developed in Appendix B. This linear system model can equivalently be written as:

$$\frac{d}{dt}\begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -A1 & -A2 & A3 \\ 0 & 0 & 0 & 1 \\ 0 & A5 & A6 & -A7 \end{bmatrix} \begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ A4 \\ 0 \\ -A8 \end{bmatrix} i + \begin{bmatrix} 0 & 0 \\ -C1 & C2 \\ 0 & 0 \\ C3 & -C4 \end{bmatrix} \begin{bmatrix} sgn(v) \\ sgn(\omega) \end{bmatrix} \quad (K1)$$

$$(di/dt) = [-A9]\, v + [-A10]\, i + [P1]\, V \quad (K2)$$

The fifth order system has been partitioned into a fourth order system (K1) coupled to a first order system (K2). The fourth order model represents the cart/pendulum subsystem, and the first order equation represents the DC servo motor dynamics. It is well known that the motor dynamics are much faster than the rest of the system. An eigenvalue study based on the parameters in Appendix I, has shown the motor eigenvalue to be three orders of magnitude greater than the pendulum system eigenvalues.

Because of the fast eigenvalue of the motor, the armature current will approach a steady state value much faster than any other state [4,9]. The derivative of the current state may therefore be assumed zero, leaving a constraint relation:

$$[P1]\, V = [A9]\, v + [A10]\, i \quad (K3)$$

The fifth order system has now been reduced to a fourth order model plus a constraint. Substitution of this constraint equation into the fourth order model (K1), yields a new fourth order model. This new model is given by:

$$\frac{d}{dt}\begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -A11 & -A2 & A3 \\ 0 & 0 & 0 & 1 \\ 0 & A12 & A6 & -A7 \end{bmatrix} \begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ A13 \\ 0 \\ -A14 \end{bmatrix} V + \begin{bmatrix} 0 & 0 \\ -C1 & C2 \\ 0 & 0 \\ C3 & -C4 \end{bmatrix} \begin{bmatrix} sgn(v) \\ sgn(\omega) \end{bmatrix}$$

where the new matrix elements are defined to be:

$$A11 = A1 + A4\ A9\ /\ A10 \qquad (K4)$$

$$A12 = A5 + A8\ A9\ /\ A10 \qquad (K5)$$

$$A13 = A4\ P1\ /\ A10 \qquad (K6)$$

$$A14 = A8\ P1\ /\ A10 \qquad (K7)$$

This new system model is an approximation based on fast motor dynamics. The fifth order model can be replaced by this fourth order model which indirectly includes information about the motor. The new model has a voltage input just as the real system does, yet the motor state, (armature current), is not directly included in the model. Because of this, the approximate model facilitates easier design of a Luenberger observer.

APPENDIX L

When implementing a real digital controller, the measured quantities are not the actual physical states of the system. The measured states are discrete, or quantized integers, which are proportional to the actual states at the time of sampling. This linear transformation can be expressed by:

$$\underline{X} = \underline{L}\,\underline{IX} \tag{L1}$$

where:

$\underline{X}$ = the real measured state (physical coordinates)

$\underline{IX}$ = the discrete measured state (integer numbers)

The complete matrix transformation (L1), is given by:

$$\begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} = \begin{bmatrix} C_x^{-1} & 0 & 0 & 0 \\ 0 & C_x^{-1} & 0 & 0 \\ 0 & 0 & C_\theta^{-1} & 0 \\ 0 & 0 & 0 & C_\theta^{-1} \end{bmatrix} \begin{bmatrix} Ix \\ Iv \\ I\theta \\ I\omega \end{bmatrix} \tag{L2}$$

where the two constants are the sensitivities of the measurement sensors, and are defined in Appendix I.

The Linear State Feedback control law can now be transformed to operate on the discrete coordinates. This will allow a computer algorithm to directly use the discrete coordinates. Using equation (L1), this transformation is given by:

$$V = \underline{K}\,\underline{X} = \underline{K}^{*}\,\underline{IX} \tag{L3}$$

where:

$$\underline{K}^{*} = [\underline{K}\ \underline{L}] \tag{L4}$$

For implementation, the actual voltage applied to the DC servo motor must also be discretized by:

$$VOLT = V\,C_V \tag{L5}$$

where C_V is also defined in Appendix I.

The Luenberger observer has been designed to estimate the actual system states. Since the LSF law has been transformed, in (L3), to utilize the available discrete coordinates, the observer must also be transformed to provide the LSF control law with estimated discrete states. The original observer equations are given by:

$$\dot{\hat{\underline{X}}} = \underline{A}\,\hat{\underline{X}} + \underline{B}\,V + \underline{G}\,\underline{D}(\underline{X} - \hat{\underline{X}}) + \underline{C}\,\mathrm{sgn}(\hat{\underline{X}}') \tag{L6}$$

Using the transformation (L1), on both the actual and the observed states, (L6) can be written as:

$$\dot{\hat{\underline{IX}}} = \underline{A}^{*}\,\hat{\underline{IX}} + \underline{B}^{*}\,VOLT + \mathrm{col}_1(\underline{G}^{*})(\underline{IX} - \hat{\underline{IX}}) + \underline{C}^{*}\,\mathrm{sgn}(\hat{\underline{IX}}') \tag{L7}$$

The matrices in (L7) have absorbed the transformation of coordinates, and are given by:

$$\underline{A}^* = [\underline{L}^{-1}\ \underline{A}\ \underline{L}] \tag{L8}$$

$$\underline{B}^* = C_V^{-1}\ [\underline{L}^{-1}\ \underline{B}] \tag{L9}$$

$$\underline{G}^* = [\underline{L}^{-1}\ \underline{G}\ \underline{D}\ \underline{L}] \tag{L10}$$

$$\underline{C}^* = -\ [\underline{L}^{-1}\ \underline{C}] \tag{L11}$$

The leading negative sign in (L11) comes from the transformation:

$$\text{sgn}(\underline{X}) = \text{sgn}(\underline{L}\ \underline{IX}) \tag{L12}$$

For this particular laboratory setup, the sensitivity parameters associated with the transformation matrix $\underline{L}$ are all negative. Therefore equation (L12) can be written:

$$\text{sgn}(\underline{L}\ \underline{IX}) = -\ \text{sgn}(\underline{IX}) \tag{L13}$$

The transformed observer system (L7), is now suitable for implementation on the computer. The observer has the discretized voltage input VOLT, and calculates the discretized output states. These states are then used directly by the transformed LSF control law to calculate the new discretized voltage.