Nonlinear Modeling and Simulation of a Hydrostatic Drive System

By

Erin E. Kruse

A THESIS
Submitted in partial fulfillment of the requirements
for the degree of
MASTER OF SCIENCE IN MECHANICAL ENGINEERING

MICHIGAN TECHNOLOGICAL UNIVERSITY
2001

This thesis, **"Nonlinear Modeling and Simulation of a Hydrostatic Drive System"** is hereby approved in partial fulfillment of the requirements of the degree of MASTER OF SCIENCE in the field of Mechanical Engineering.

Department: Mechanical Engineering - Engineering Mechanics

Thesis Advisor:

_____

Dr. Gordon G. Parker Date

Department Chair:

_____

Dr. William W. Predebon Date

Abstract

Sea conditions limit the safety and efficiency of crane maneuvers performed aboard U.S. Navy crane ships. Rough seas can make the large-payload maneuvers time-consuming, and dangerous. Operation of the cranes is currently not allowed above Sea State 2. An existing Navy initiative is to develop and test a swing-free controller which will allow operations to continue above this level. The work described in this thesis is the nonlinear model development, system identification and simulation of the Hagglunds TG3637 crane's existing hydrostatic drive system. The intended use of this model is to identify components that may limit crane performance, to develop advanced control strategies, and evaluate swing-free performance in simulation.

There are four main components developed in the drive system model: the control card, the pump directional spool valve actuated with two 24-volt solenoids, the pump stroker and swash plate assembly, and the hydraulic motor. Within the drive system, the voltage commanded by the operator is first passed through the control card which converts the voltage to a current. This current actuates a pair of solenoids that are in contact with the directional spool valve within the pump. The pump then supplies flow to the hydraulic motors which rotate the slewing turret, and luff and hoist winches. The control card model, as well as the pump/motor model, are developed using experimental data collected on board the Flickertail State using a Hagglunds TG3637 crane. Solenoid performance is characterized in the laboratory on a benchtop control card-solenoid subsystem. The modularity of the model allows for investigation of changes in performance due to system improvements.

The full dynamic equations of motion characterizing the drive system are developed, as well as a simplified, lumped parameter model. The latter is used to identify model parameters using operational data in conjunction with a numerical optimization code. Nonlinearities captured by the model include deadzone, speed saturation, and acceleration limiting. The resulting simulation is compared to operational data to assess its fidelity.

This work offers two contributions not found in existing literature. The first is the dynamic model of the pump and its mechanical feedback system for speed tracking. The

second is the system identification implementation that will facilitate rapid identification of other cranes, and be useful for tuning swing-free controller gains.

Acknowledgments

I would like to take the opportunity to thank those who helped make this work a success. Thank you to my advisor, Gordon Parker, whose constant encouragement to continue on for a Ph.D. helped me to think I must be doing *something* right! Mike Agostini and Becky Petteys also deserve special thanks for their continual assistance, smiling faces and great (academic, of course) conversation in the lab and office. Kenneth Groom at Sandia National Laboratories also deserves many thanks for supporting this work in more ways than one.

Dexter Bird was a great source of knowledge during the shipboard testing. Without his assistance we'd probably *still* be there, so I thank him for that. The cooperation and assistance of Scott, Bernie, and the rest of the crew aboard the Flickertail State was also indispensable, as well as enjoyable, during our time on the ship and in subsequent phone calls. Additionally, Greg Schwartz at Mannesmann-Rexroth was a wealth of information and showed great patience as I worked to understand the pump and control card-- thank you.

I would like to thank the folks kind enough to serve on my committee: Hanspeter Schaub, Chuck VanKarsen, and Richard Honrath. I certainly appreciate your willingness to peruse this document and offer your suggestions. Without a doubt I will also want to offer similar sentiments after your input at the defense. A big thanks to Chris Edlin for all sorts of support and advice throughout the final months, including the use of his house as a soup kitchen, help on the defense presentation, and for reminding me that it is possible to come out alive despite the sleep deprivation!

Finally, I would like to thank those involved in the less academic side of my life. My partner, Tony, has been a constant source of encouragement-- even going so far as to try to convince me that control cards are interesting enough to discuss over dinner. Thank you to my sister Brandi, in whom I have found increasing solace as we both grow up. Thanks also to my mother for making me who I am today and for her usual encouragement-- she'd make a great spokesperson for a certain shoe corporation! And thanks to my father and G'parents Kruse for their constant presence and creative forms of encouragement. And of course, where would I be without my support crew: Elly Bunzendahl, Jennifer Victory, Cindy Martineau, and number one neighbor, Jennifer Klipp. Thanks ladies.

I have never let my schooling interfere with my education.

~ Mark Twain

Table of Contents

List of Figures

List of Tables

1 Introduction

The United States Navy utilizes crane ships to transfer cargo from one ship to another at sea. A typical operation is depicted in Figure 1.1.

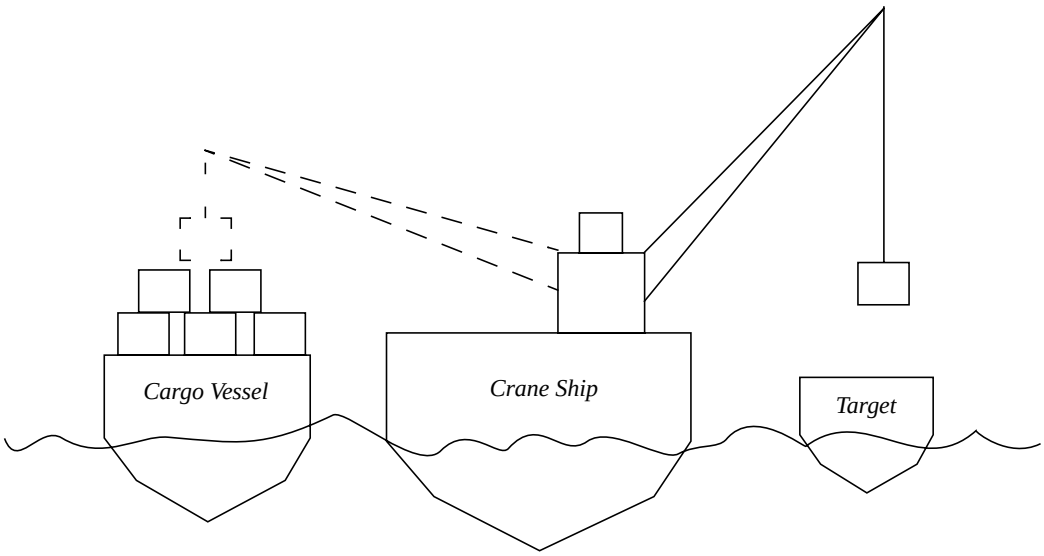

FIGURE 1.1 Typical Crane Ship Maneuver

The crane ship positions itself between the cargo vessel and the target ship. Payloads are picked up from the cargo ship, transferred over the deck of the crane ship and placed on the target. Moving the freight requires use of any of the four TG3637 cranes installed on the crane ships, each capable of lifting 35 tons.

Currently these cranes are operated in light to moderate sea conditions by operators with a wide range of expertise. Extended transfer time is required in moderate seas, but operations must be halted until high sea states diminish due to both difficulty and danger. The overall goal of the project is to fit the cranes with a swing-free controller which will allow operations to continue at sea state levels above that which is currently possible, and to

expedite transfer in moderate seas. The work described here is the system identification and modeling of the TG3637 crane's current hydrostatic drive system for use in design and evaluation of the controller. The model is developed using operational data taken the week of June 26, 2000 during shipboard testing of a TG3637 crane installed on The Flickertail State [1].

1.1 Drive System Overview

All four TG3637 cranes aboard each crane ship are supplied electrical power by two diesel generators that deliver 1,600 hp each. The individual cranes convert electrical power to mechanical power through 460 volt A.C. electric motors rated at 335 hp continuous duty, or 442 hp at 40% duty cycle. Power from the main electric motor is then transmitted to the crane's winches for luffing and hoisting operations, and to the slewing gears via a hydrostatic transmission. The crane operation terminology is described in Figure 1.2: raising or lowering the payload by changing the length of the lift line constitutes a *hoisting* maneuver; rotation about a line running vertically through the center of the crane cab is considered *slewing*; and *luffing* is defined as changing the angle between the crane's boom and horizontal.

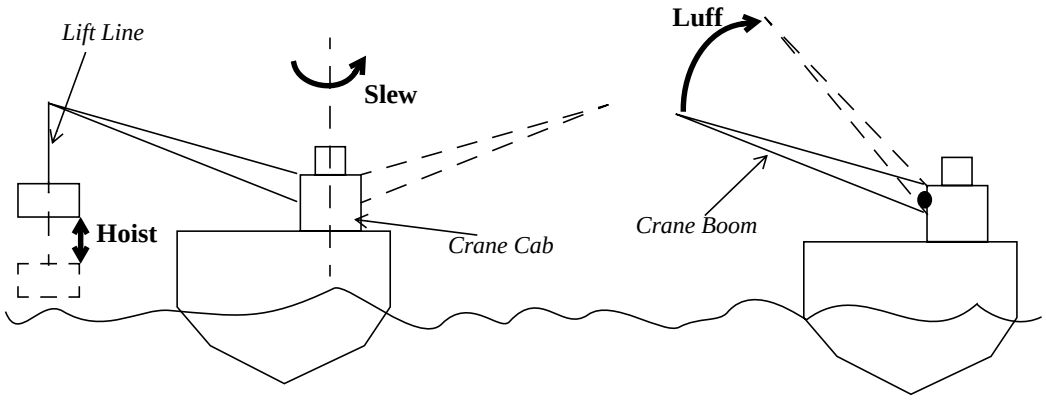

FIGURE 1.2 Depiction of the Cranes' Axes: Hoist, Slew, and Luff)

The main motor is allowed to run at a constant 1774 RPM with the hydraulic pump providing the speed and directional variations in the motors. The drive system described here is illustrated in Figure 1.3.

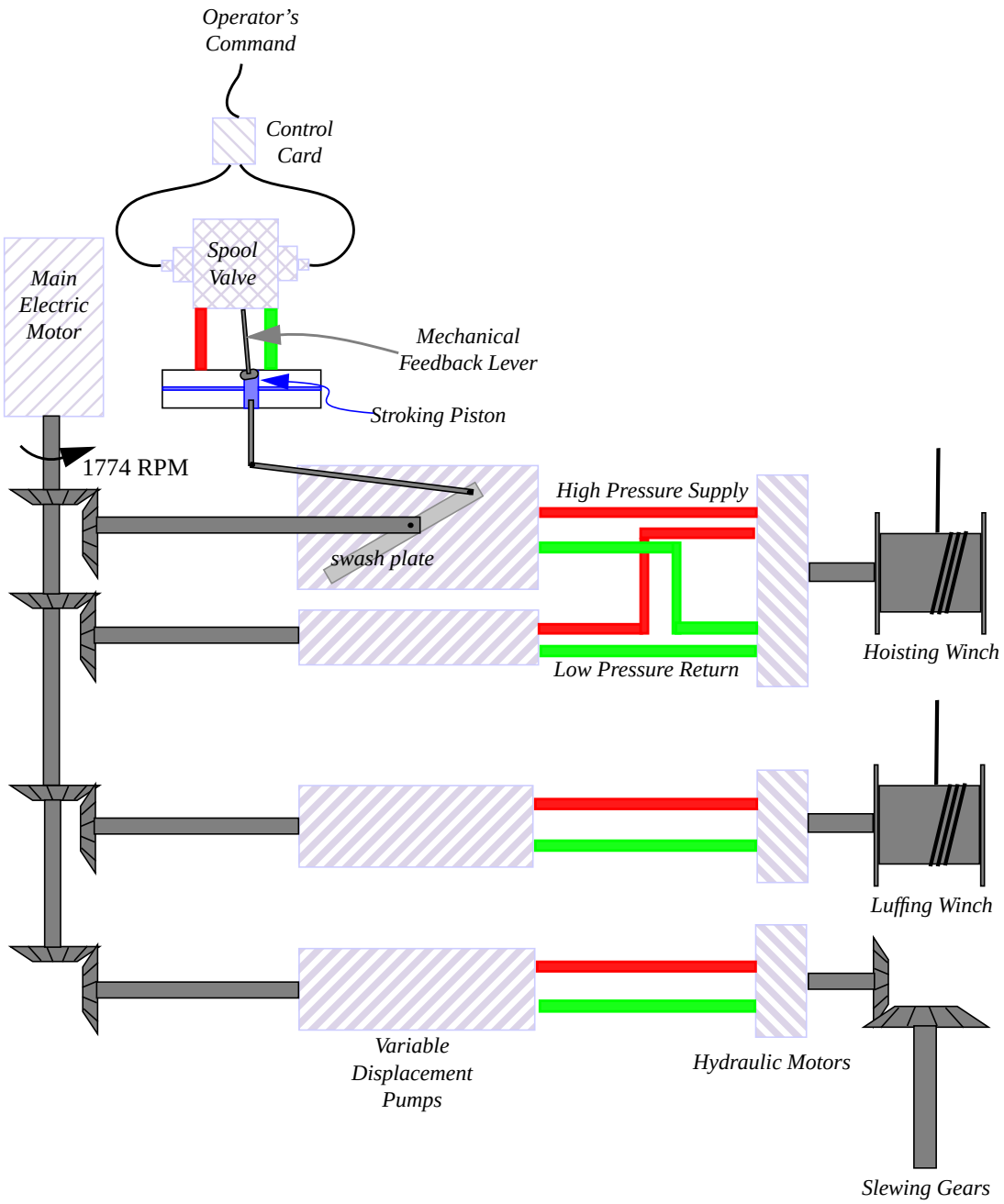

FIGURE 1.3 TG3637 Drive System

A block diagram of the drive system is shown below to facilitate understanding of the flow of information from one component to another.

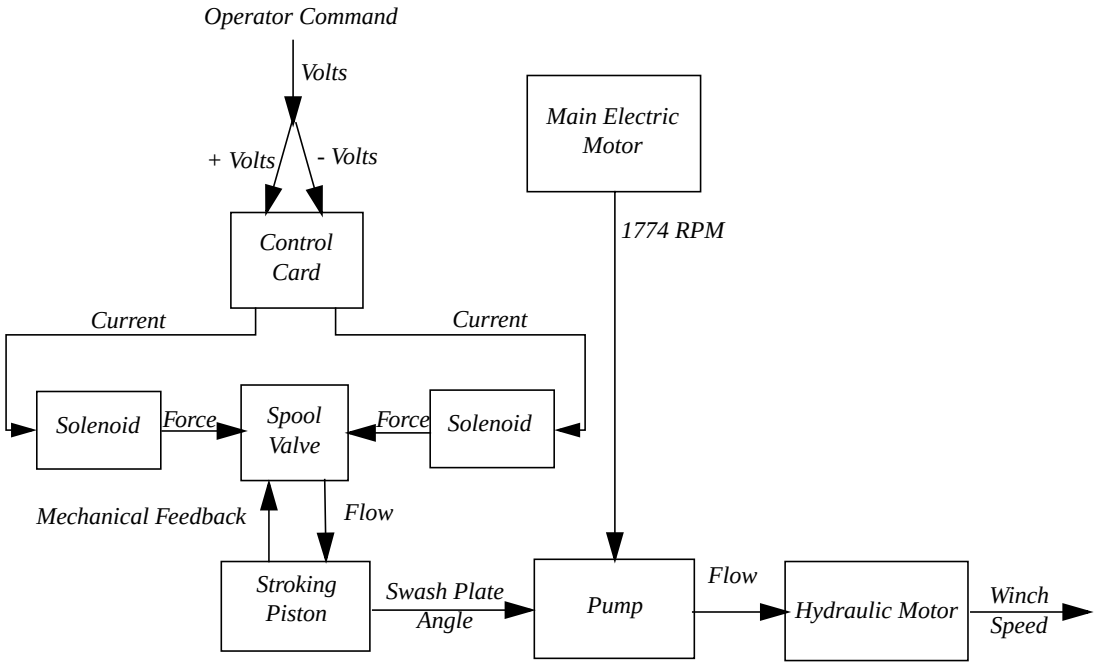

FIGURE 1.4 TG3637 Full Drive System Block Diagram

The hydrostatic transmissions for each axis consist of Rexroth variable displacement, closed circuit, hydraulic pumps and Hagglunds hydraulic motors. The pumps force the hydraulic fluid through the high pressure line to the hydraulic motor which then returns the fluid in a low pressure line back to the pump. The speed of the motor is controlled by a directional spool valve attached to the pump and the swash plate located within the pump. By varying the volume of flow to the stroking piston, the angle of the swash plate is adjusted, thus varying flow to the hydraulic motor. An illustration of the inner components of an axial piston pump with a swash plate can be seen in Figure 1.5.

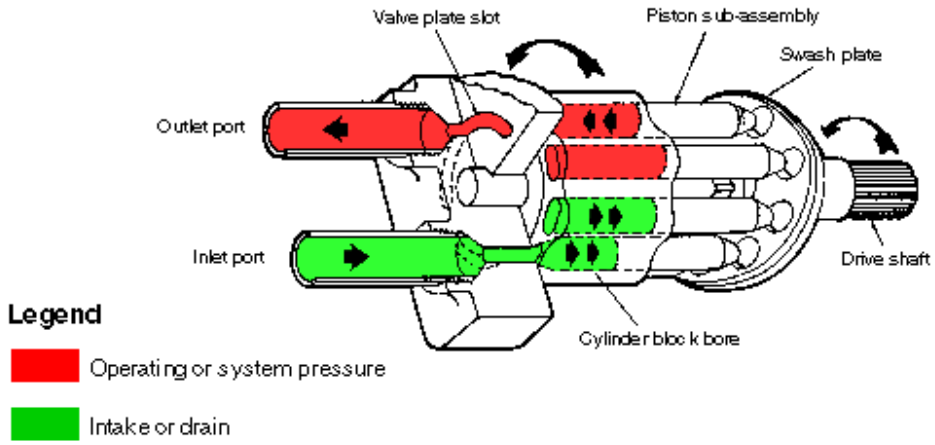

FIGURE 1.5 Axial Piston Pump with Swash Plate

The position of the swash plate affects the flow to the motor by altering the stroke of the pistons in the pump. At full stroke the pistons are able to travel their full range of motion and thus pump the maximum amount of fluid. As shown in Figure 1.6, when the swash plate is oriented vertically it is exactly perpendicular to the piston's range of motion. In this orientation the pistons are all held at a constant position, resulting in zero flow. The spool valve and swash plate are also responsible for the reversal of the hydraulic motor. By rotating the swash plate through the vertical position the high and low pressure lines to the hydraulic motor are switched. The direction of the motor is thus reversed.

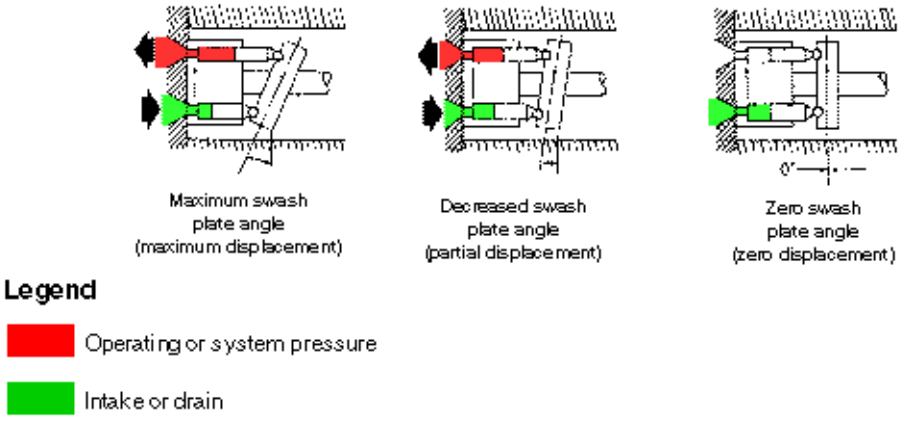

FIGURE 1.6 Axial Piston Pump Detail Showing Swash Plate Orientation vs. Flow

The directional spool valve is actuated by two 24 volt proportional solenoids positioned at either end of the spool. As the operator commands an increase in speed, one of the solenoids is energized; the opposite solenoid is energized when a decrease in speed is commanded. With the spool in the centered position, all flow to the stroker is blocked and the swash plate angle, and therefore motor speed, are held constant. A mechanical feedback lever and spring mechanism are used to return the spool to center when the desired swash plate angle is achieved.

The electrical current to actuate the solenoids is typically provided by a Hagglunds voltage to current card that is remotely operated by a joystick mounted potentiometer in the crane cab. For the operational tests performed on the Flickertail State, the standard control cards were replaced by a Rexroth MDSD-1 control cards with known factory presettings, and the joystick was replaced with a laptop computer. Due to different pump configurations on one of the crane's axes, two different voltage-to-current cards are used in the crane. The hoist axis uses a dual pump system, thus requiring a slightly different control card. The cards installed in the testing of the crane were the Rexroth MDSD-1K-2x/2 for the hoist axis, and the MDSD-1K-2x/4 for slew and luff. The only difference in the two cards being the factory pre-settings on the tunable potentiometers. Table 1.1 lists the part names, and numbers where available, as they pertain to each axis.

Table 1.1 Drive System Part Names and Numbers, by Axis

Part Name	Hoist	Slew	Luff
Control Card, Original	Hagglunds, unknown number	Hagglunds, unknown number	Hagglunds, unknown number
Control Card, Op. Testing only	Rexroth MDSD-1K-2x/2	Rexroth MDSD-1K-2x/4	Rexroth MDSD-1K-2x/4
Solenoid	Two Rexroth 24 Volt Proportional HU 09441692	Two Rexroth 24 Volt Proportional HU 09441692	Two Rexroth 24 Volt Proportional HU 09441692
Hydraulic Pump	Rexroth AA4V250 EL, and AA4V125 EL	Rexroth AA4V250 EL	Rexroth AA4V250 EL
Hydraulic Motor	Hagglunds 84-25100	Hagglunds unknown number	Hagglunds 64-16300

The letters “EL” in the pump name designate the type of control module [2] installed on the pump.

1.2 Model Overview

The model is sufficiently general to be applied to all three of the crane’s axes: hoist, slew, and luff as illustrated in Figure 1.2. There are four main model components: (1) control card, (2) pump directional spool valve and solenoid, (3) stroker and swash plate assembly, and (4) hydraulic motor. Although there are many parameters that define the model, the time history input is joystick voltage, and the output is the rotation speed of the winches on the hoist and luff axes, and the rotation speed of the turret on the slew axis.

The model has 14 parameters in the control card model and 22 parameters in the pump/motor model that must be identified. A numerical optimization approach was used in each case to determine their values for each axis. This could be applied to other drive systems

given their input voltages, solenoid currents, and output motor speeds. Through the examples included in this thesis the model shows good performance for both speed and acceleration saturated operation. A better input to output match could be achieved with additional data. Specifically, the internal states of the system are computed, but have not been checked against experimental data. Given independent solenoid currents, spool valve displacement, swash plate angle, stroker pressure, and the pressure drop across the motor, the model could be adjusted to predict the internal states with more resolution, and thereby creating a better input to output match.

Finally, the model does not capture the cam/cam roller effects seen on the motor speed data during loaded conditions on hoist. The cam rollers are attached to the end of each radial piston in the hydraulic motor and ride against a fixed cam ring. The cam ring is a scalloped cylinder which encircles the ring of pistons. As the pistons fill with hydraulic fluid the cam rollers are forced to roll into the valleys of the cam ring, thus creating the torque that rotates the motor. The spatial geometry of the cam ring causes an oscillation which has a frequency that is dependent on motor speed. As the load increases, the speed amplitude oscillation increases. The possibility that this could excite high frequency crane modes (e.g. boom bounce) may exist, and should be considered during all subsequent servo design studies.

1.3 Thesis Structure

A discussion of recent work in the system identification and modeling of hydraulic drive systems follows in Section 2. The control card model form, described in Section 3, is based on the manufacturer specifications supplied with the card, in addition to observed behavior of measured current from the operational data. The control card optimization

used to identify the model parameters is described in Section 4 alongside qualitative and quantitative illustrations of the model's success. Section 5 describes the proportional solenoid driven by the control card. The spool valve and the stroker/swash plate models, described in Section 6, are based on a lumped parameter representation. Dynamic equations are derived for completeness. However, a force equilibrium representation is sufficient for accurate matching to experimental data. Furthermore, a linearized version of the equations, for small motion, is included for possible use in servo design. The motor is represented as a gain between flow rate and rotation rate. Parameters in the pump/motor model are identified in Section 7, and hoist, slew and luff operational data is used to illustrate the model's ability to match experimental data. A complete summary of the model from voltage to rotation speed is found in Section 8.

2 Literature Review

Most modeling and system identification of electro-hydraulic drive systems is done for controller design, as are all of the articles referenced in this review of recent work. Likewise, the work contained in this thesis is also directed toward controller design. In contrast, the model developed here is also intended for controller evaluation. For successful evaluation of a proposed control strategy, a model must be of a higher fidelity than that required for controller design. The ability to develop a successful model lies almost exclusively in being able to handle the many dynamic effects within these complex systems, particularly those which are nonlinear. The fundamental equations representing pressure differentials and fluid flow through a valve are fairly standard. Five papers, [3], [4], [5], [6], and [7] simply refer the reader to the same text [8] for the derivation of their fundamental equations. It is at this point that the models begin to vary greatly in the degree to which they include linear and nonlinear dynamic performance characteristics.

Damping in the swash plate is addressed in both [9] and [5]. It is agreed that a study of the damping effects on the swash plate dynamics might have a positive effect on the simulation results in this work. To do this properly, however, information on internal states such as stroker pressure and swash plate angle would be required. Acceptable results are obtained without modeling any swash plate damping, therefore this parameter was not added to the already lengthy list. Along these same lines, the work done on modeling the torque on the swash plate angle of axial piston pumps by Zeiger and Akers, in [10], and that of Manring, in [11], is quite relevant to the work in this thesis. The effect of the load and the stroking piston on the swash plate torque is included in this thesis, but not

addressed in [10]. The time varying effect of the pistons as they transition between the high and low pressure ports is heavily considered in both [10] and [11]; the piston pressure profile is simply divided into two categories, high and low pressure, in this work. The inability to monitor the swash plate angle during operational testing prohibited the examination of this effect into the swash plate equation.

Often, the model presented captures only one or two of the more important nonlinearities present in the system. A four-way directional spool valve, similar to that used in the development of this work, is modeled in [12] as it actuates a cylinder an inertial load. This model is used to develop a force control system, and although it mentions a saturation reached when the valve attains its maximum opening, this is dismissed and only leakage in the cylinder is considered. Bobrow and Lum in [7], utilize the basic model developed in [8] with the addition of one term which lumps together the effects of all friction and hysteresis effects. The control law developed based on this model then indirectly identifies this lumped nonlinear term through online selection of controller parameters. In [3], the authors are emphatic about the importance of compensating for nonlinearities in hydraulic cylinders, but goes on to emphasize only deadband. Deadband in these systems can be attributed to many things including spool overlap and/or coulomb friction. The effects of friction in the lip seal of a hydraulic actuator cylinder is covered very thoroughly in [13]. Here, a successful model is developed using the Hammerstein model, where nonlinearities are assumed to be separable from the system's dynamics. Again, coulomb friction is the only nonlinearity encompassed by Halme's model of a hydraulic positioning servo in [4]. An in-depth analysis of solenoid performance in [14] successfully captures hysteresis and saturation. These phenomenon, however, are never exhibited in the solenoids in this

research due to their limited travel while in service. The remaining dynamics in the valve model are developed from basic equations similar to those found in the previous works cited. Both [6] and [15] address the effects of spool underlap in the valve. The model presented in this thesis assumes that the original design of a critically lapped spool still exists, however this could be an issue in other control modules or may need to be considered as the system wears.

Yao et al. handle nonlinearities caused by directional change of valve opening, friction, leakage, and valve overlap in their study of controlling electro-hydraulic servo systems in the presence of non-smooth and discontinous nonlinearities [16]. The model utilizes basic equations identical in form to those used in this work, however, the introduction of the nonlinearities is handled differently. The coulomb friction force is addressed individually in the dynamics of the inertial load, while all other "external disturbances and unmodeled friction forces" are lumped into one term. Although the model described in this work is simplified for lumped parameter optimization, the effect of each lumped parameter is clearly defined, not representing ambiguous nonlinearities. Leakage is addressed in a similar manner, the leakage term is presented in the cylinder dynamic equation. The issue of spool overlap is represented by an area gradient in the spool valve equation.

Jelali and Schwarz in [17] identify nonlinear models in observer canonical form for hydraulic drive systems in a similar parametric approach to that described in this document. The model's voltage input is converted to a linear output position, much like a swash plate angle. Using Bernoulli's equation for valve flow as the framework, the model addresses oil elasticity, and valve and cylinder friction.

Finally, a very thorough nonlinearity development is found in [18]. Hysteresis, saturation, orifice area relationships and pressure flow relations are all addressed by McLain et al. Unfortunately the application is a single-stage, four-way valve with characteristics that are stated and observed to be very different from those of the two-stage spool valves as dealt with in this work. This is the only paper found to address the issue of flow restriction due to the presence of orifices within the pump. However, the pump orifices described here are in a quite different orientation and thus a similar development is not appropriate.

All of the work described above is shown to be sufficient for controller design within a chosen bandwidth of the system of interest. To simulate the outer bounds of performance characteristics, as is necessary when exercising a projected control strategy, a very thorough model is critical. This work differs from that above in its attention to the nonlinear details presented by the complex electro-hydraulic drive system. Captured together in one model is leakage within the pump, at the swash plate in particular; the effect of reaching the physical limits of spool and swash plate travel; speed saturation due to maximum pump flow rates; acceleration limits caused by orifices between the valve and stroking piston; and limits in pressure. Additionally, each of these system parameters are handled in such a way that the model can be updated alongside system modifications or to preview the effects of such changes. Individually, the modeling of each nonlinearity may not rival the completeness of some of the other works, but the thorough inclusion of as many nonlinearities as appropriate provides a model which is expected, overall, to be more widely applicable and accurate.

3 Rexroth Control Card Model

The control card is used to convert the operator's commanded voltage into a current. The card outputs two channels which drive the two 24 volt solenoids attached to the pump's control module.

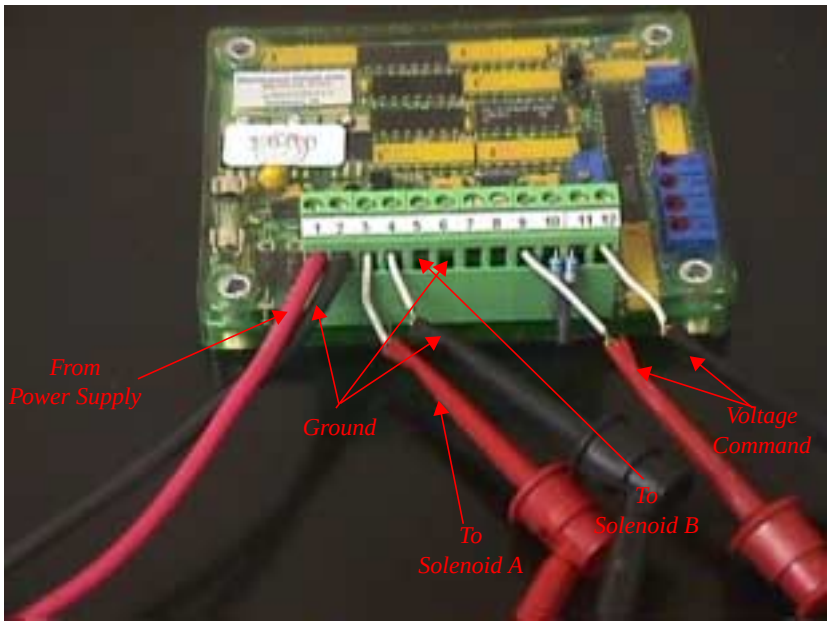

FIGURE 3.1 Control Card and Connections

Figure 3.1 illustrates the control card connections in a benchtop subsystem for the Rexroth MDSD-1K-2x/4 as set up for laboratory testing of the solenoid (described in Section 5). Note that only one solenoid (solenoid A) is connected to the card in this figure. The terminals for solenoid B connections are labeled. Terminals 7 and 8 are test terminals which were not used in either the laboratory or operational testing. Terminals 10 and 11 are connected with 200 KΩ of resistance as required for use with a ± 10 volt command signal.

Current produced by the control card is pulse width modulated according to the factory preset frequency of 100 Hz, for all three axes. An F.W. Bell, true RMS, AC non-contact

milliammeter is used to rectify the pulse-width modulated (PWM) signal and determine the effective output current before passing it to the A/D board in operational testing.

Given the manufacturer supplied specifications which accompany the control cards upon shipment, a model of the control card can be deduced with varying degrees of success. If taken literally, the specifications lead to a model of very little accuracy in neither the steady state nor the dynamic areas of performance. With a small amount of knowledge about the system and its performance in the operational data one can develop a model which is far more accurate in the steady state regime, but which still does not accurately capture the card's dynamics. The limitations experienced in the specifications-based model required the development of a more detailed model to capture the nuances of the steady-state features as well as the dynamic effects seen in the operational data.

The following model is intended to simulate the Rexroth control card's conversion of the operator commanded voltage to the current applied to the solenoids in the hydraulic pump. The regions within the control card's performance requiring particular attention are the dynamics as the signal departs from zero, and as the signal returns to zero. A general model is developed for all three of the crane's axes, where axis-dependent parameter values are identified in Section 4. The control card required on the hoist axis differs from the other two axes, which both use the same card, due to the fact that the hoist axis utilizes a tandem pump system. The model is therefore optimized for two sets of parameters; one set for the hoist axis, and one set for the slew and luff axes. Plots illustrating the match between measured and simulated current are provided in Section 4 for hoist and slew.

3.1 Model Development

A high level block diagram of the card model is provided in Figure 3.2. Due to the nature of the control card's circuitry, a qualitative block diagram provides the best overall representation of the model. The card's functionality can then be divided up into the three major blocks.

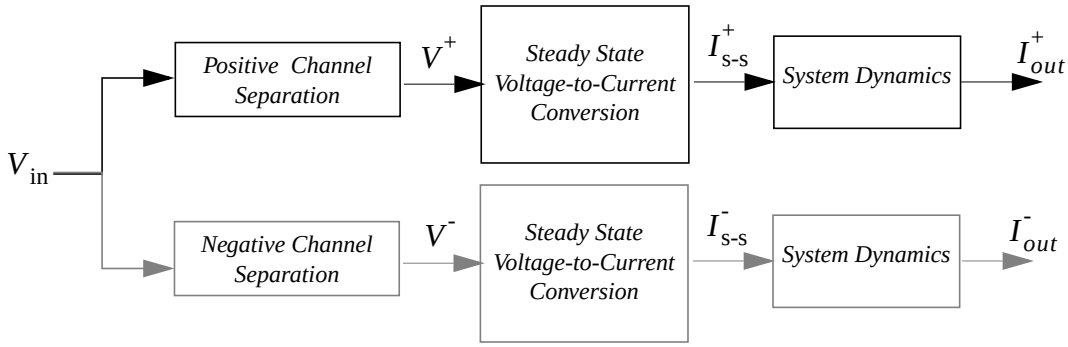

FIGURE 3.2 Control Card Model Block Diagram

In the following sections, these three fundamental functions are described in greater detail. The positive channel will be the focus in development, and for notational convenience the positive sign superscript on the parameters will be dropped. Using the same form, only a change in parameters is required to capture the performance of the negative channel. Identification of the model's parameters, based on experimental data, is handled in Section 4.

3.2 Channel Separation

The card is capable of receiving both positive and negative inputs. This input voltage is then routed in the control card to create an output current that has two channels, one for each solenoid. These channels will be described as the positive and negative channels as one responds only to positive voltage inputs and the other to negative. It is assumed that the performance characteristics for each channel are symmetrical, and the card model described in this document follows the current on the positive voltage input channel.

The first block in the diagram simply zeros all negative content in the voltage command. This produces a current which would power one solenoid.

$$V = \begin{cases} V_{in} & V_{in} \geq 0 \\ 0 & V_{in} < 0 \end{cases} \tag{3.1}$$

The card also includes another function similar to this, as seen in Figure 3.2, but which would respond only to the negative input, zeroing all of the positive input. This would produce the current supplied to the other solenoid.

3.3 Steady State Voltage to Current Conversion

The filtered input voltage, V_{in}, is converted to a steady-state current, $I_{s\text{-}s}$, in the second block of Figure 3.2. Based on information from the Rexroth control card specifications sheet [19], the relationship between input voltage and output current can be modeled as

$$I_{s\text{-}s} = \begin{cases} 0 & V < V_{dz} \\ J + G_1(V - V_{dz}) & V \geq V_{dz} \end{cases} \tag{3.2}$$

Figure 3.3 and Figure 3.4, both taken from the specifications sheet, show the card has trim potentiometers for adjusting the jump, J, (P5 and P6) and the gain, G_1, (P3 and P4).

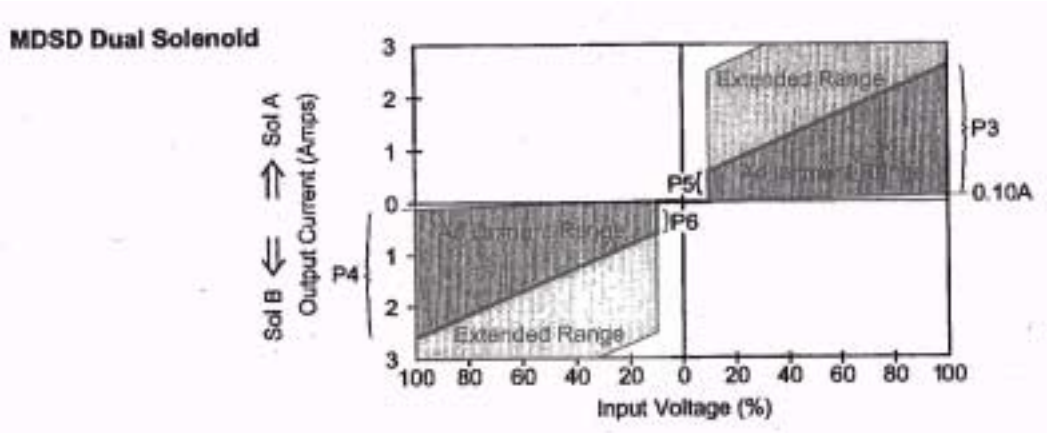

FIGURE 3.3 Rexroth Control Card Specifications, Adjustable Potentiometer Regions

P1	Ramp time (std.)	Sol A	0.2 10 sec.
P2	Ramp time (std.)	SolB	
P3	Max. current	Sol A	0 2.5 A
P4	Max. current	SolB	
P5	Min. current	Sol A	0.1 2.5 A
P6	Min. current	SolB	
P7	PWM frequency		75 275 Hz

FIGURE 3.4 Rexroth Control Card Specifications, Adjustable Range

The deadzone, V_{dz}, appears to be about 10% of full scale according to Figure 3.3, however it is not adjustable, nor given in the specifications. Based on experimental ramp data, the idealized steady state current expression (3.2) was modified to allow a third order polynomial behavior instead of linear. The final relationship used is

$$I_{s\text{-}s} = \begin{cases} 0 & V < V_{dz} \\ J + G_1(V - V_{dz}) + G_2(V - V_{dz})^2 + G_3(V - V_{dz})^3 & V \geq V_{dz} \end{cases} \tag{3.3}$$

As seen in Figure 3.4, the potentiometers for each solenoid (A and B) can be adjusted independently, indicating that the card's positive and negative channels could require different parameters. This is captured in Section 4, where parameters are determined for each channel.

3.4 Dynamic Features

The third block of Figure 3.2 incorporates several functions to capture all of the card's dynamic features not described in the specification sheets. A small time delay, second order dynamics, as well as the selective rate limiting observed in the measured current signal are all added here.

Although it is insignificant in terms of the crane's response time, a small time delay is needed to match the current output of the control card model with the measured current. This is important for synchronizing measured and simulated responses during the system identification process. The delay is described mathematically as

$$I_{\text{s-s,td}} = \begin{cases} 0 & t \le \tau_{td} \\ I_{\text{s-s}}(t - \tau_{td}) & t > \tau_{td} \end{cases} \tag{3.4}$$

where τ_{td} is the time delay in seconds, and $I_{\text{s-s,td}}$ is a state internal to the third block in Figure 3.2.

$I_{\text{s-s,td}}$ is then passed through a second order transfer function to produce the oscillatory response observed when the current jumps from zero to any non-zero value. The system's dynamic response for the slew axis has an unusual aspect; the initial overshoot is large, while the oscillations damp out quickly. This feature cannot be captured with the second

order transfer function alone, but is successfully modeled with the addition of a coulomb-like nonlinear term. Using Euler integration, for example,

$$\begin{aligned} \dot{x} &= f(\tau) \\ x &= \int_0^t f(\tau)d\tau \\ x_{n+1} &= x_n + h f_n \end{aligned} \tag{3.5}$$

the discrete time representation is

$$I^*_{s\text{-}s,\,n+1} = 2(1-\zeta\omega_n h)I^*_{s\text{-}s,\,n} + (2\zeta\omega_n h - h^2\omega_n^2 - 1)I^*_{s\text{-}s,\,n-1} + \\ -Ch^2\text{sign}\left(\frac{1}{h}[I^*_{s\text{-}s,\,n} - I^*_{s\text{-}s,\,n-1}]\right) + (h^2\omega_n^2)I_{\text{s-s,td,n}} \tag{3.6}$$

where C is the coulomb-like parameter, and h is the integration time step. The differential equation for this is

$$\ddot{I}^*_{s\text{-}s} + 2\zeta\omega_n\dot{I}^*_{s\text{-}s} + C\text{sign}\dot{I}^*_{s\text{-}s} + \omega_n^2 I^*_{s\text{-}s} = \omega_n I_{\text{s-s,td}} \tag{3.7}$$

Finally, the data shows the effects of two different rate limits; one for signals departing zero and one for those approaching from zero. The first (A in Figure 3.5) is imposed above a threshold ($I_{t,\,1}$) as the signal departs from zero. The other (B in Figure 3.5) limits signals as they begin to approach zero and remains in place until reaching a lower threshold, $I_{t,\,2}$. Below $I_{t,\,2}$ the current decays as a first order system (C in Figure 3.5), however in all other nonrate-limited places (D in Figure 3.5) $I^*_{\text{s-s}}$ is simply passed through to the

output. An exaggerated illustration of a generic step response is included in Figure 3.5, and annotated with the rate limited areas, thresholds, and the exponential response.

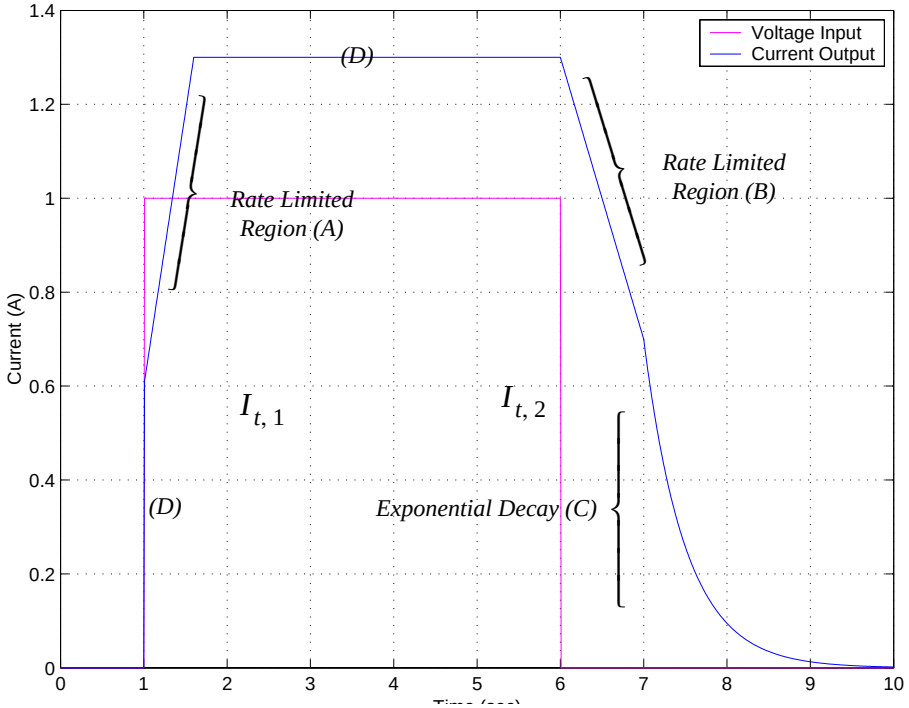

FIGURE 3.5 Illustration of Nonlinear Control Card Behavior

Equations 3.8, and 3.9 describe this relationship for the positive channel. An inverse relationship exists for the negative channel; parameters for both are shown in Section 4.

For voltage inputs which are increasing, the relationship is:

$$\dot{I}_{out} = \begin{cases} \dot{I}_{lim1} & I^*_{s\text{-}s} > I_{t,1} \text{ and } \dot{I}^*_{s\text{-}s} \geq \dot{I}_{lim1} \\ \dot{I}^*_{s\text{-}s} & I^*_{s\text{-}s} > I_{t,1} \text{ and } \dot{I}^*_{s\text{-}s} \leq \dot{I}_{lim1} \\ \dot{I}^*_{s\text{-}s} & I^*_{s\text{-}s} \leq I_{t,1} \end{cases} \tag{3.8}$$

and for decreasing voltage inputs this can be described mathematically as

$$\dot{I}_{out} = \begin{cases} \dot{I}_{lim2} & I^*_{s\text{-}s} > I_{t,\,2} \text{ and } -\dot{I}^*_{s\text{-}s} \geq \dot{I}_{lim2} \\ \dot{I}^*_{s\text{-}s} & I^*_{s\text{-}s} > I_{t,\,2} \text{ and } -\dot{I}^*_{s\text{-}s} \leq \dot{I}_{lim2} \\ \dfrac{d}{dt}\left(I^*_{s\text{-}s,\,i} e^{-a(t-\tau_i)}\right) & I^*_{s\text{-}s} \leq I_{t,\,2} \end{cases} \tag{3.9}$$

where $1/a$ is the time constant of the first order decay, and $I^*_{\text{s-s,i}}$ and τ_i are instantaneous values of the state and time, captured once as the current crosses $I_{t,\,2}$ to provide a smooth transition from the rate limited signal to the first order response. Note that the function which determines whether the voltage is increasing or decreasing will hold the previous state for durations where the voltage is constant. Initially, when the voltage input is zero, and no previous state of increasing or decreasing exists, I_{out} holds its initial conditions, typically zero.

Section 4 describes the optimization method used to parameterize the model as well as showing representative comparisons between measured and simulated current. All of the parameters used in the card model are tabulated in this section. Indirectly, Section 7 also illustrates the model's performance through representative plots comparing measured winch speed data and simulated winch speed data, where the simulated winch speed uses a simulated current.

4 Control Card Model Parameter Identification

Parameterization of both positive and negative channels of the control card was handled with a numerical optimization code. During shipboard testing both channels of measured output current were run through a single ammeter, as illustrated in Figure 4.1, creating a signal which was the summation of the signals to the individual solenoids.

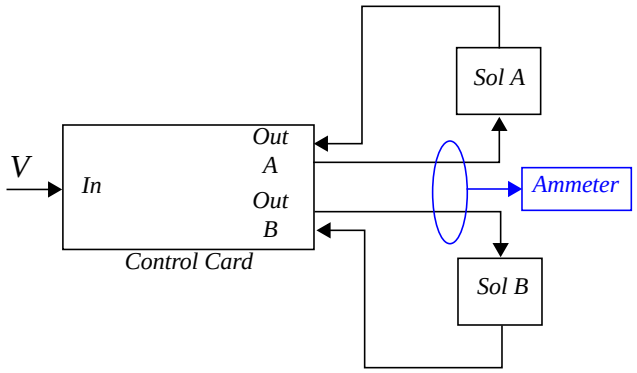

FIGURE 4.1 Solenoid/ Control Card Subsystem Showing Ammeter Placement

In order to create a comparable signal, the positive and negative channels of the simulated current are combined through a summation of the absolute value of each channel. The combined signal is used in the optimization code to identify the control card model parameters. The current which is used to identify the pump/motor model is in the original form, having both positive and negative components.

The cost function used in optimization was formed as the sum of the integral square error between simulated and measured current for ramp, step, and sine wave voltage inputs. Mathematically,

$$J = \sum_{i=1}^{n} J_i \tag{4.10}$$

where J is the cost, and i represents a particular data set (e.g. 1 = 4V step, 2 = 6.5V step, etc.). The optimization searches for the X^* that minimizes J, where X^* is the set of control card parameters. The block diagram in Figure 4.2 illustrates the optimization process.

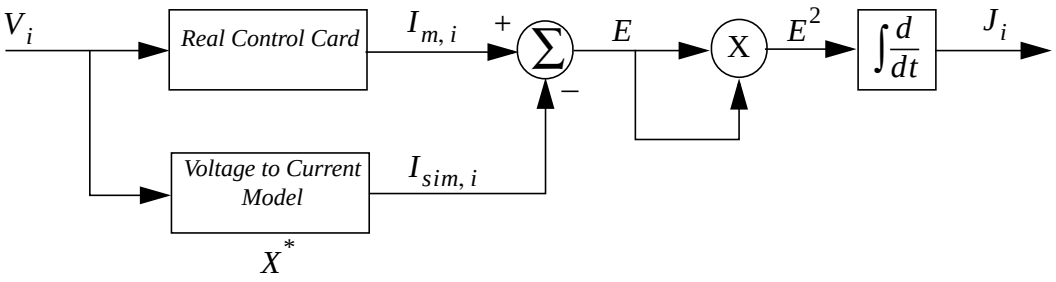

FIGURE 4.2 Block Diagram of Optimization

Initial estimates of the parameters were determined through extensive “manual tuning” prior to the numerical optimization, which chose each subsequent set of parameters using the recursive quadratic programming method. The cost function (curwrap.m, elwrap_hoist2.m, and elcost.c) and its setup file (curset.m) for the hoist axis are found in Appendix C and Appendix B respectively. Only the initial estimates differ in the files for the slew axis.

4.1 Parameterization, Positive Voltage Input

The optimized value of the parameters in the steady-state voltage-to-current conversion block of Figure 3.2, and Equation 3.3 for the positive channel of the control card are:

Table 4.1 Positive Channel Hoist, Slew and Luff Parameters, Equation 3.3

Parameter	Hoist Axis	Slew and Luff Axes
J (A)	0.27473	0.17033
V_{dz} (V)	0.12663	0.090948
G_1 (A/V)	0.049315	0.037892
G_2 (A/V^2)	0.00081975	0
G_3 (A/V^3)	-4.5317e-05	0

The parameters which were optimized for the dynamic response, Equation 3.4, 3.6, 3.8, 3.9, for the positive channel of the card are

Table 4.2 Positive Channel Hoist, Slew and Luff Parameters

Parameter	Hoist Axis	Slew and Luff Axes
τ_{td} (sec)	0.014	0.014
ζ (n.d.)	0.71887	1.4157
ω_n (rad/sec)	56.253	98.377
C (A)	0	4.8368
$\dot{I}_{lim1}$ (A/sec)	2.2841	2.2148
$\dot{I}_{lim2}$ (A/sec)	2.2615	2.2615
$I_{t,1}$ (A)	0.33628	0.33628
$I_{t,2}$ (A)	0.13724	0.12352
a (sec^{-1})	19.131	19.131

4.2 Parameterization, Negative Voltage Input

The optimized value of the parameters in the steady-state voltage-to-current conversion block of Figure 3.2, and Equation 3.3 for the negative channel of the control card are:

Table 4.3 Negative Channel Hoist, Slew and Luff Parameters, Equation 3.3

Parameter	Hoist Axis	Slew and Luff Axes
J (A)	-0.2951	-0.16474
V_{dz} (V)	-0.097857	0.088321
G_1 (A/V)	0.054706	0.036385
G_2 (A/V^2)	0.00042879	0
G_3 (A/V^3)	-9.7369e-05	0

The parameters which were optimized for the dynamic response, Equation 3.4, 3.6, 3.8, 3.9, for the positive channel of the card are

Table 4.4 Negative Channel Hoist, Slew and Luff Parameters

Parameter	Hoist Axis	Slew and Luff Axes
τ_{td} (sec)	0.014	0.014
ζ	0.72563	1.1014
ω_n (rad/sec)	61.833	135.39
C (A)	0	4.8368
$\dot{I}_{lim1}$ (A/sec)	2.2733	2.2733
$\dot{I}_{lim2}$ (A/sec)	2.6483	2.6483
$I_{t,1}$ (A)	-0.34863	-0.34863
$I_{t,2}$ (A)	-0.10683	-0.10683
a (sec)	-21.379	-21.379

4.3 Comparison of Current Simulation to Test Data

The figures in this section illustrate the accuracy of the Rexroth control card model. The data sets chosen are a representative subset of those used in the final verification of the model in Section 7. Note that because one current sensor recorded the signal to both solenoids, the measured current signal remains positive regardless of the voltage input. The simulated current is adjusted by summing the absolute value of the positive and negative channels to facilitate comparison. The scaled input voltage signal is also shown in the figures as a reference. The pulse width modulation, used for current output, is not modeled. This results in smooth, simulated current signals as compared to the true signal which includes the dither signal. It is assumed that the crane does not respond to this 100 Hz

phenomenon; its true purpose is to avoid static friction in the spool valve. Sampling at 512 Hz during operational testing meant that this dither effect was not accurately captured (nor was this the intention) and it often causes beating in the measured current signal. This is particularly evident in Figure 4.12. The actual current signal should not be mistaken as oscillatory in these cases. The measured current data is unfiltered.

4.3.1 Hoist Data

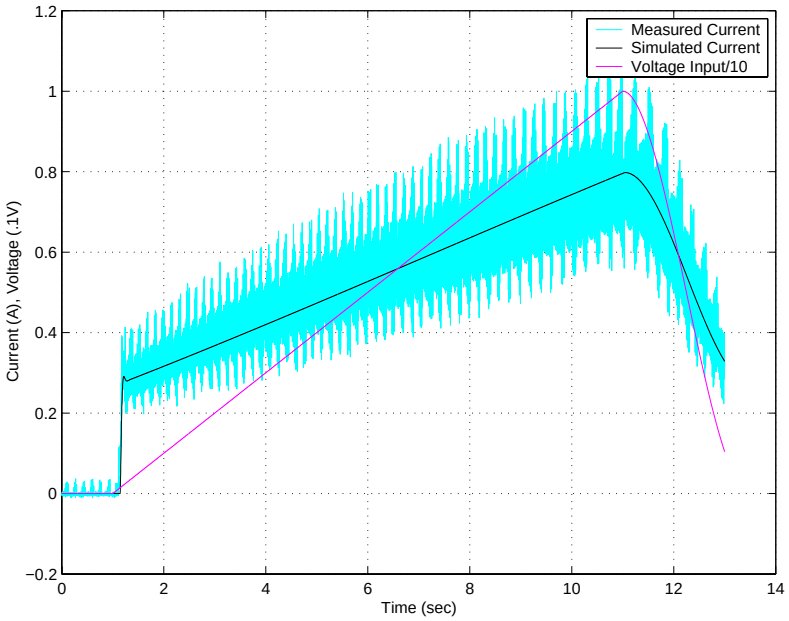

FIGURE 4.3 Hoist Current Data, 1V/sec Ramp (hoistr3.dat)

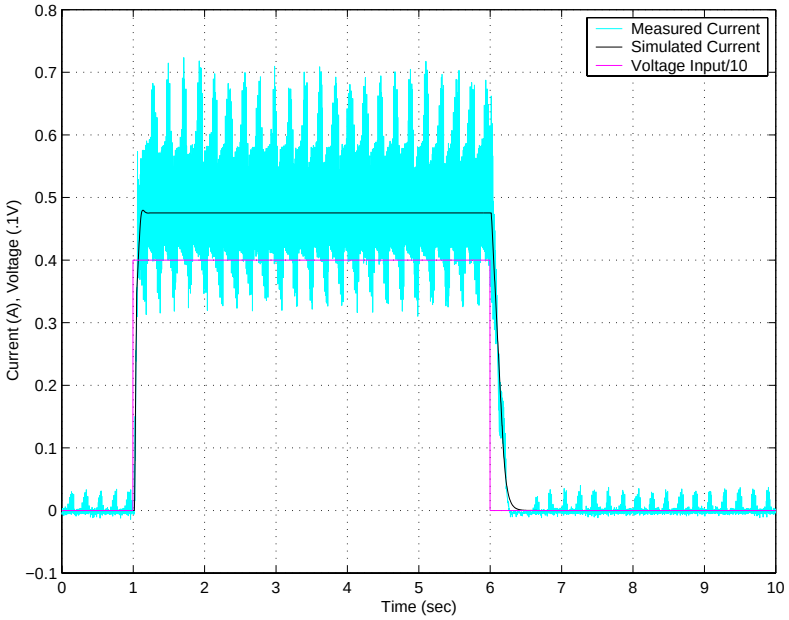

FIGURE 4.4 Hoist Current Data, 4V Step (hoistr6.dat)

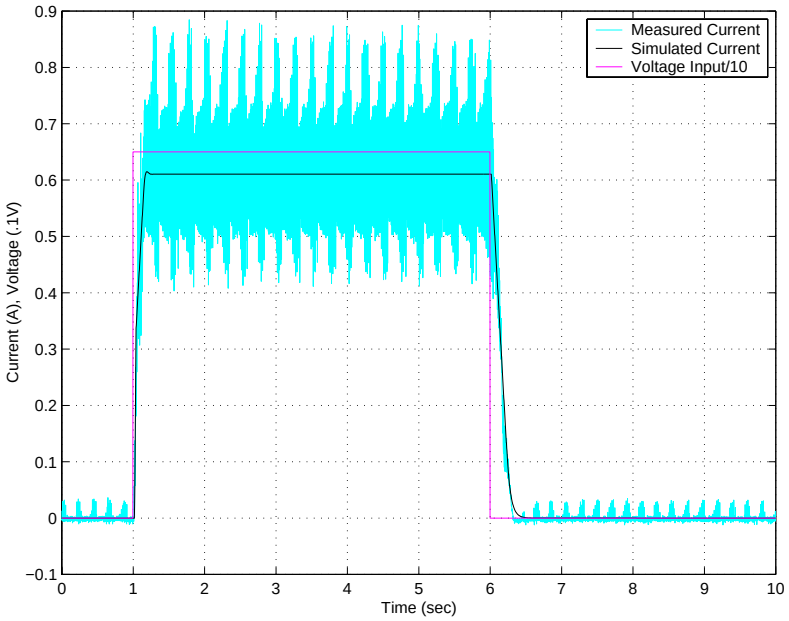

FIGURE 4.5 Hoist Current Data, 6.5V Step (hoistr8.dat)

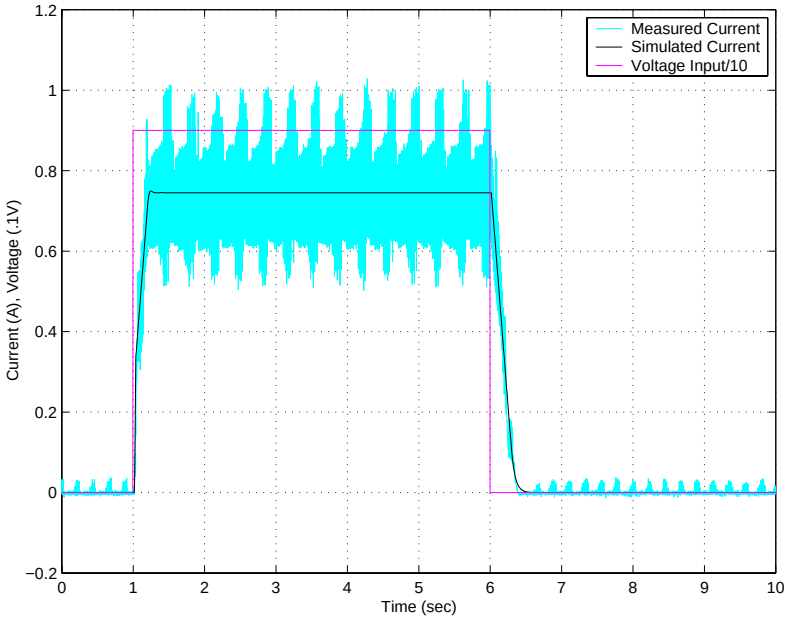

FIGURE 4.6 Hoist Current Data, 9V Step (hoistr10.dat)

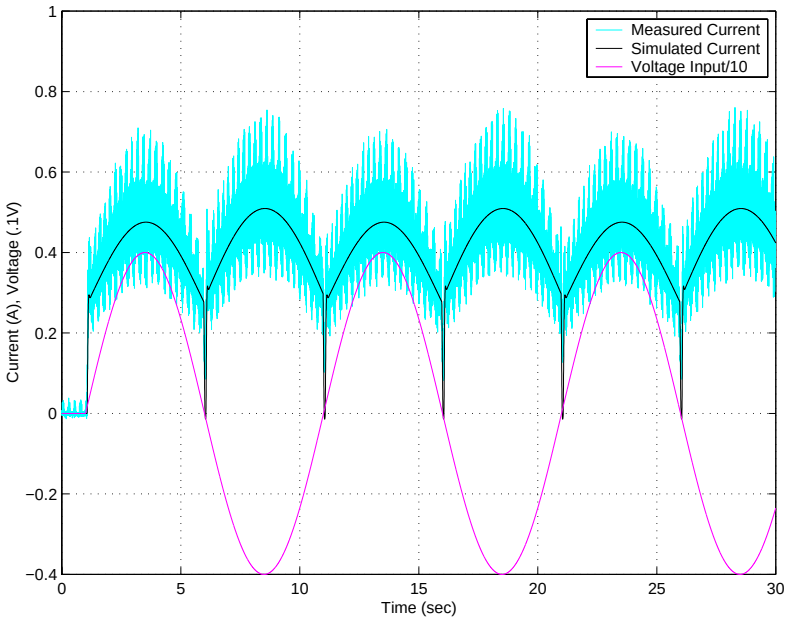

FIGURE 4.7 Hoist Current Data, 4V, 0.1Hz Sine (hoistr13.dat)

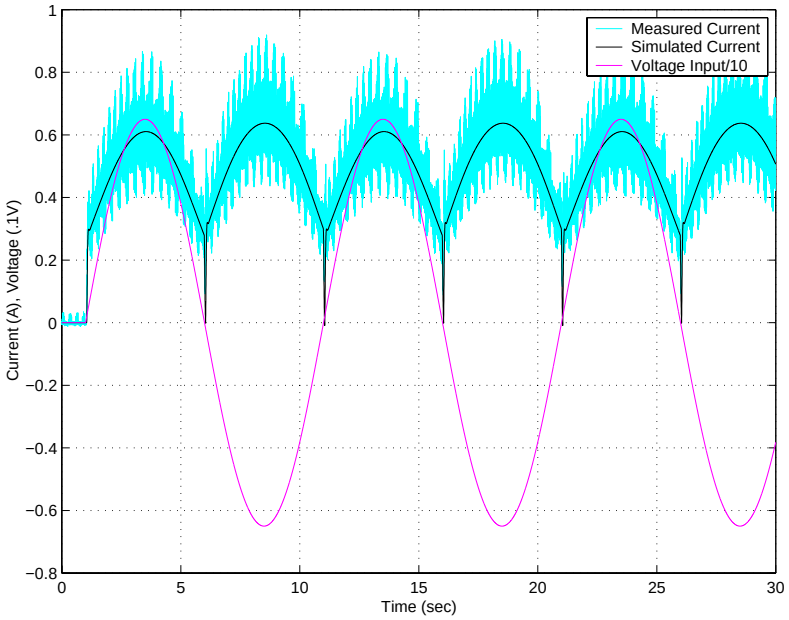

FIGURE 4.8 Hoist Current Data, 6.5V, 0.1Hz Sine (hoistr20.dat)

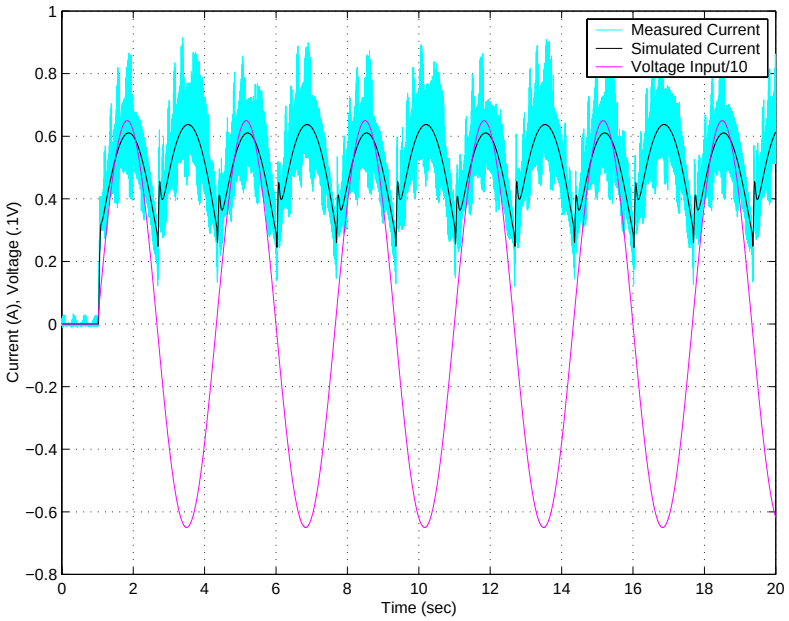

FIGURE 4.9 Hoist Current Data, 6.5V, 0.3Hz Sine (hoistr22.dat)

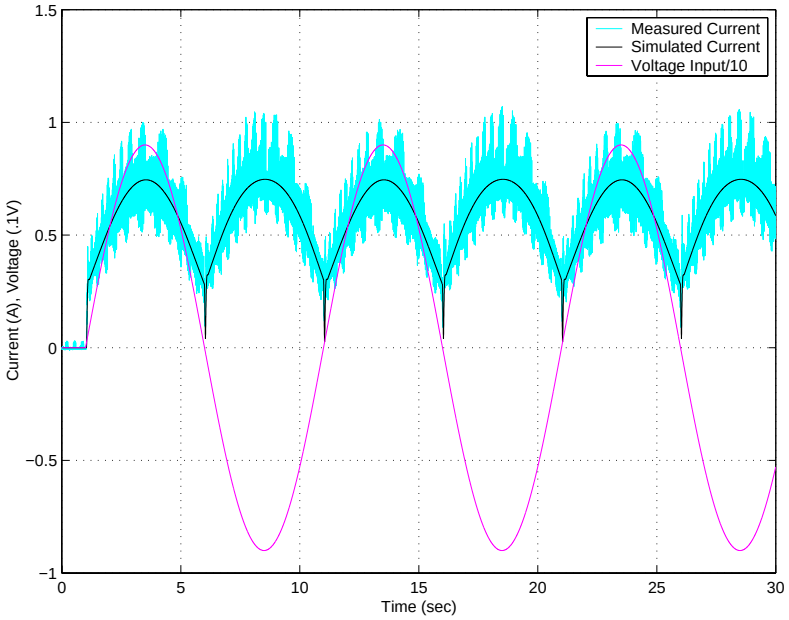

FIGURE 4.10 Hoist Current Data, 9V, 0.1Hz Sine (hoistr27.dat)

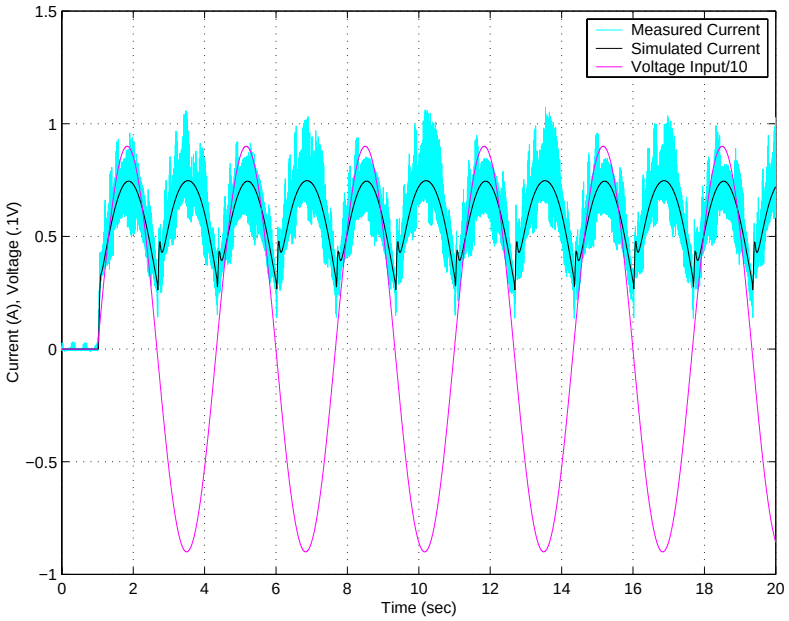

FIGURE 4.11 Hoist Current Data, 9V, 0.3Hz Sine (hoistr29.dat)

4.3.2 Slew Data

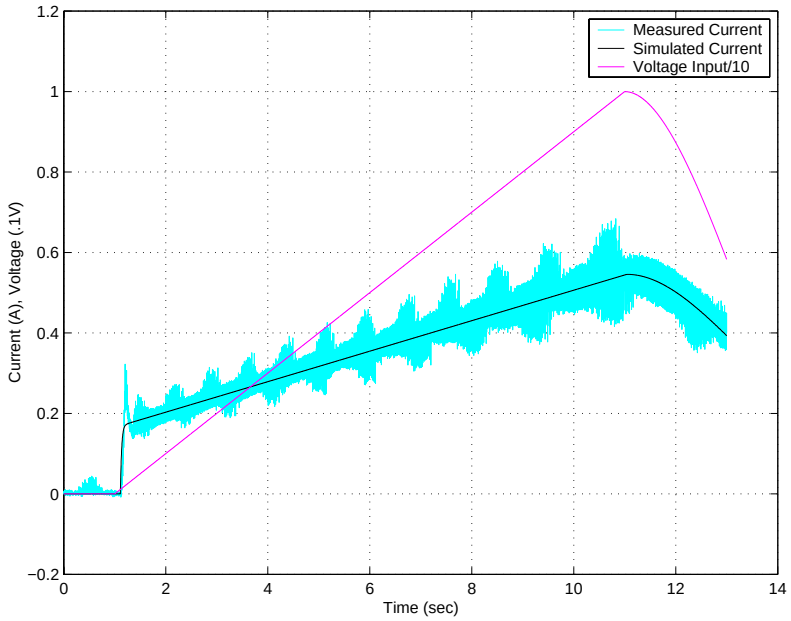

FIGURE 4.12 Slew Current Data, 1V/sec Ramp (slewr2.dat)

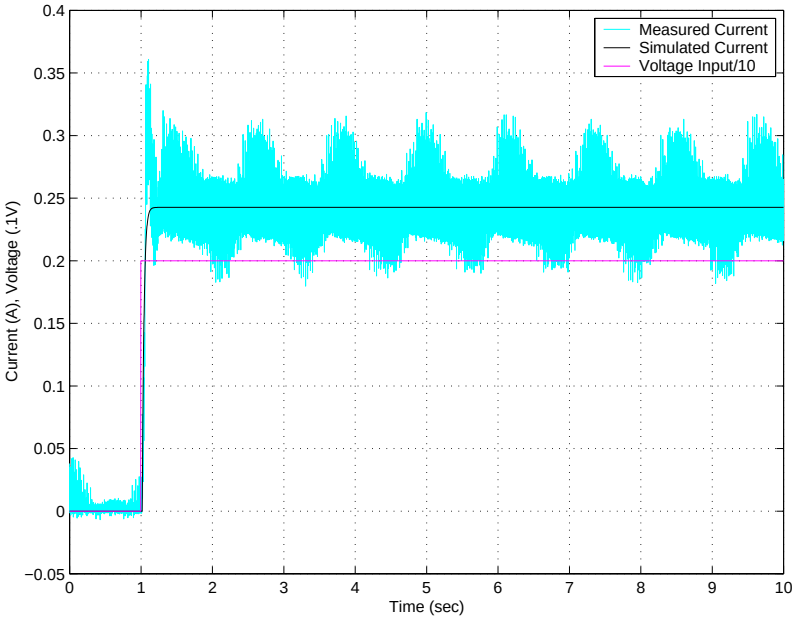

FIGURE 4.13 Slew Current Data, 2V Step (slewr10.dat)

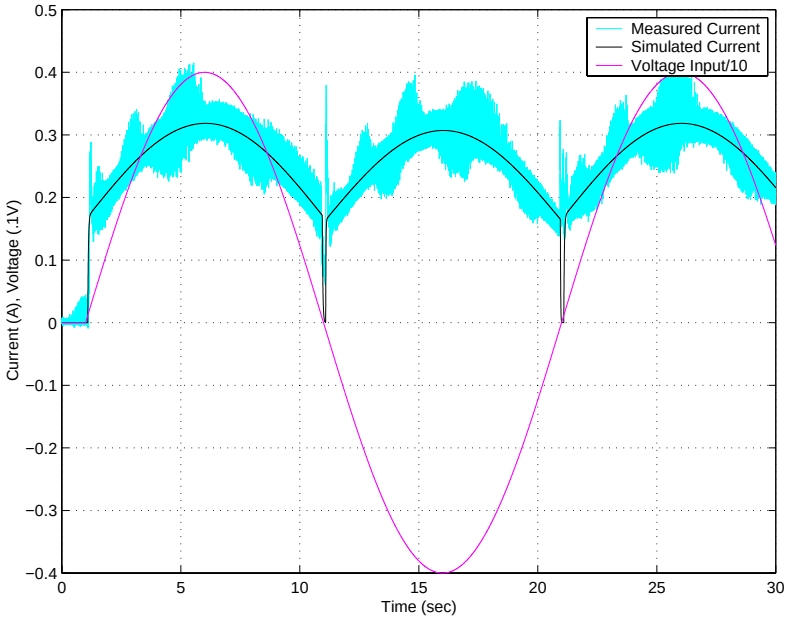

FIGURE 4.14 Slew Current Data, 4V, 0.05Hz Sine (slewr4.dat)

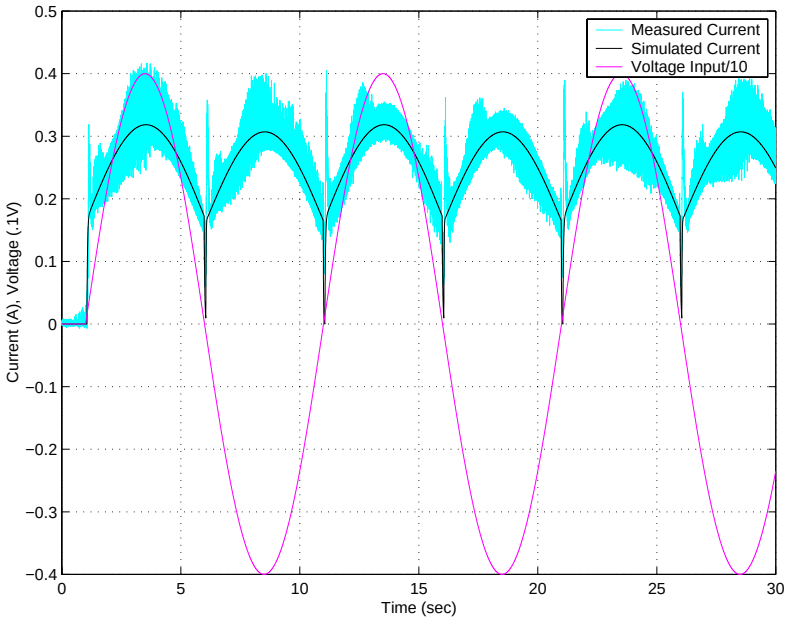

FIGURE 4.15 Slew Current Data, 4V, 0.1Hz Sine (slewr5.dat)

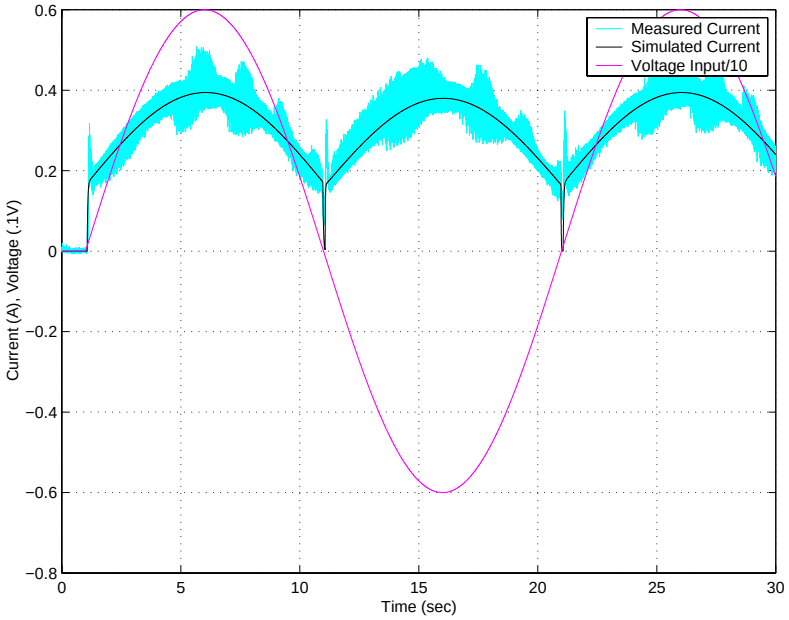

FIGURE 4.16 Slew Current Data, 6.0V, 0.05Hz Sine (slewr8.dat)

4.4 Error Quantification, Control Card Model

The figures above qualitatively illustrate the accuracy of the model. This section will quantify the maximum percent error experienced in each type of data set. The error is tabulated by axis, with the luff axis assumed to be similar to slew. The overshoot seen in some of the step and ramp data was deemed negligible during model development and therefore is not considered in the selection of the region with the maximum percent error. Likewise, the discrepancies in the regions of the sine wave tests which would ordinarily be the zero-crossings are ignored because the phenomenon is the result of data acquisition methods, not the card itself.

Table 4.5 Hoist Axis Control Card Maximum Percent Error

Test Type	Maximum % Error	Speed/ Amp/ Freq	Associated Figure
Ramp	5.34%	1 V/sec	Figure 4.3
Step	8.63%	4 V	Figure 4.4

Table 4.5 Hoist Axis Control Card Maximum Percent Error

Test Type	Maximum % Error	Speed/ Amp/ Freq	Associated Figure
Sine	8.79%	4 V, 0.1 Hz	Figure 4.7

Table 4.6 Slew Axis Control Card Maximum Percent Error

Test Type	Maximum % Error	Speed/ Amp/ Freq	Associated Figure
Ramp	0.55%	1 V/sec	Figure 4.12
Step	1.98%	2 V	Figure 4.13
Sine	2.63%	4 V, 0.05 Hz	Figure 4.14

5 Solenoid Performance

The EL Control Module installed on some versions of the Rexroth AA4V pumps use two 24 volt proportional solenoids to actuate the directional spool valve. Shown in Figure 5.1 is the control module with one solenoid removed and the spool pulled partially out the body of the valve.

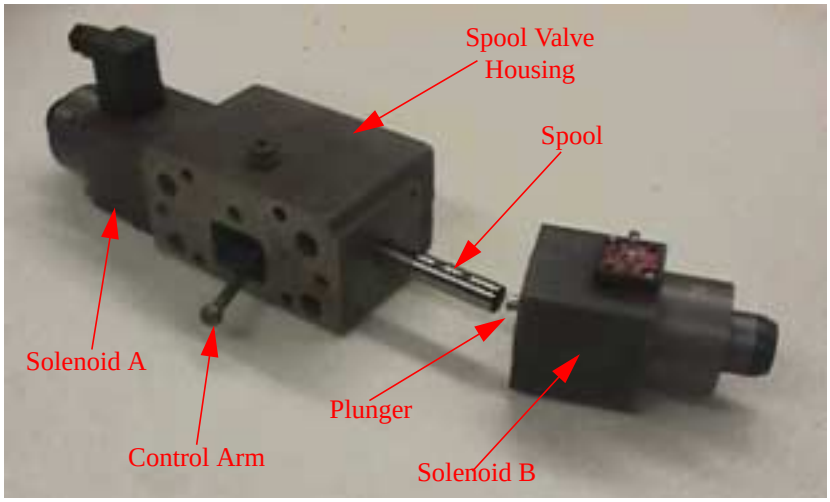

FIGURE 5.1 Photograph of EL Control Module

The force-displacement behavior of a proportional solenoid differs from a standard solenoid as the plunger reaches a fully extended position. Typically, a solenoid's force increases exponentially as the plunger retracts due to the diminishing air gap (annotated in Figure 5.2). The air in the gap provides a greater resistance to the flow of the magnetic

field than the iron of the C-frame or the plunger, so as it shrinks, the force capability of the solenoid increases.

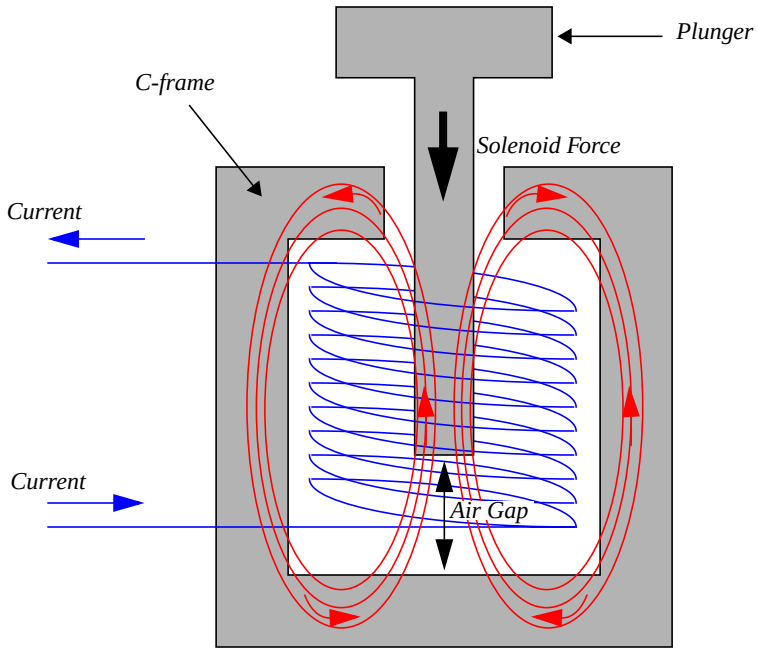

FIGURE 5.2 Standard Solenoid Configuration

A proportional solenoid eliminates this effect for a portion of the stroke. This can be done by utilizing design features that maintain and effectively constant air gap, or by using non-magnetic materials that cause it to appear to the solenoid that there is a constant air gap [20]. The result is a reshaping of the force-displacement curves to include a linear portion. When coupled with a carefully tuned spring that opposes plunger motion, solenoid force can be made proportional to input current. The key to choosing and calibrating the spring is ensuring its stiffness and the applied preload cause its force/displacement curve to lie along the linear portions of the solenoid force curves.

The same model solenoids used in the Rexroth pump were bench tested by fitting the solenoid with an Omega LCGC Series, Miniature Compression Disc Load Cell. The force

delivered by the solenoid was measured by the cell as it was compressed between the plunger and an adjustable brace. As shown in Figure 5.3, the brace is moved by adjusting the wing nuts. The load cell is held in place by a fitting clamped to the plunger in such a way that maximum plunger displacement is not affected.

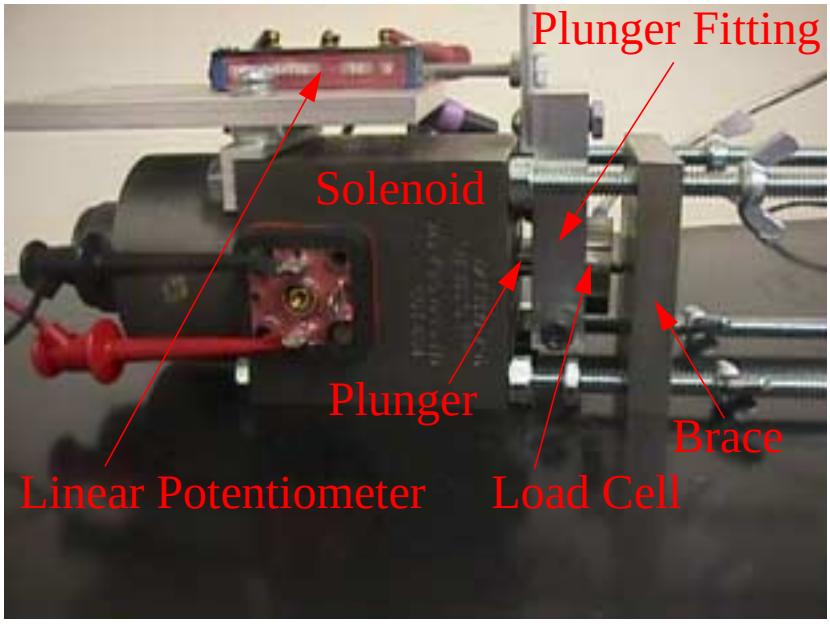

FIGURE 5.3 Solenoid fit with Load Cell, Linear Potentiometer and Brace

The distance (“d” in Figure 5.4) between the solenoid body and the brace was varied at a given current to obtain each constant-current, force-displacement curve.

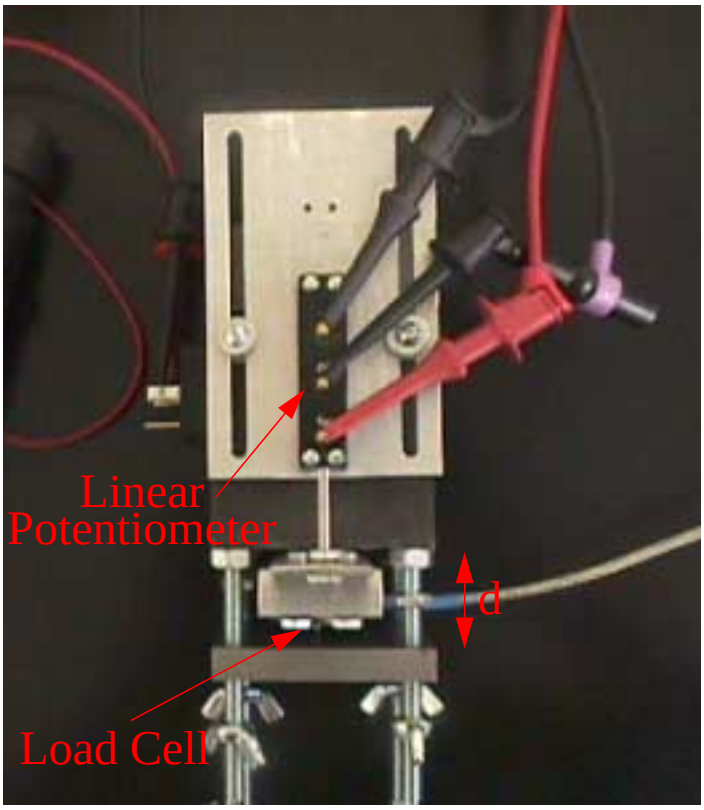

FIGURE 5.4 Top View of Solenoid Experimental Setup

The displacement in Figure 5.5 is the extension of the plunger as measured by the linear potentiometer in Figure 5.4. Note that the legend states the voltage commanded as opposed to the current. Due to the presence of the control card, the actual command

applied is a voltage. The corresponding currents, in ascending order from 1V to 10V, for the tested voltages shown in Figure 5.5 are: 0.2537A, 0.4090A, 0.5635A, 0.7185A.

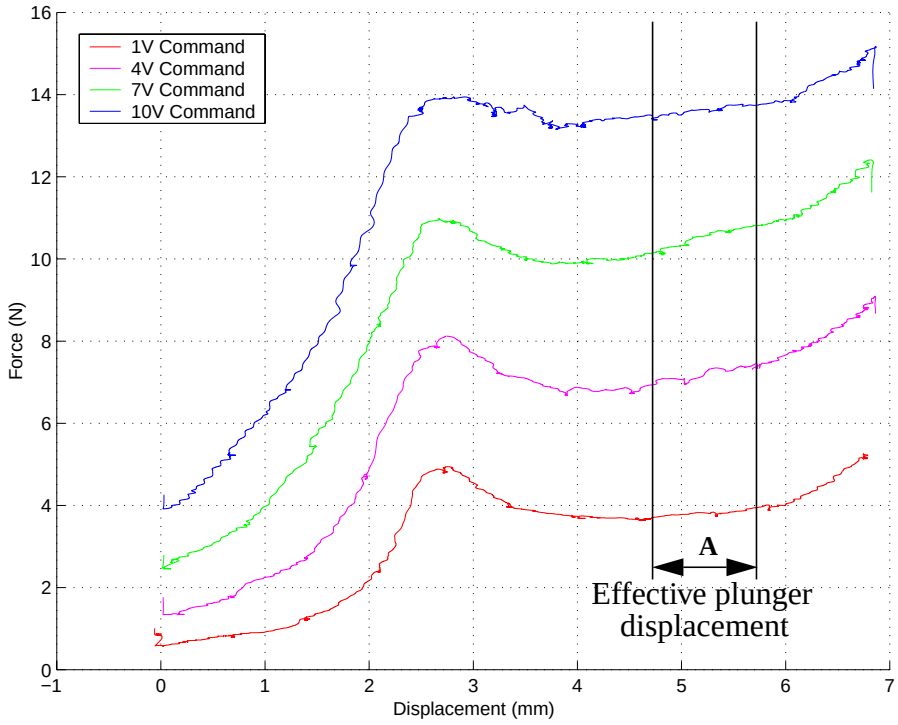

FIGURE 5.5 Force-Displacement Curves for Full Solenoid Plunger Travel

The travel of the solenoid plunger is actually so limited when attached to the spool valve that the functional region of the force curves are as marked by A, in Figure 5.6 above. Figure 5.6 plots voltage input versus solenoid force for the average displacement within the

operating range (A). Error bars indicate the maximum and minimum forces occurring at the extremities of A.

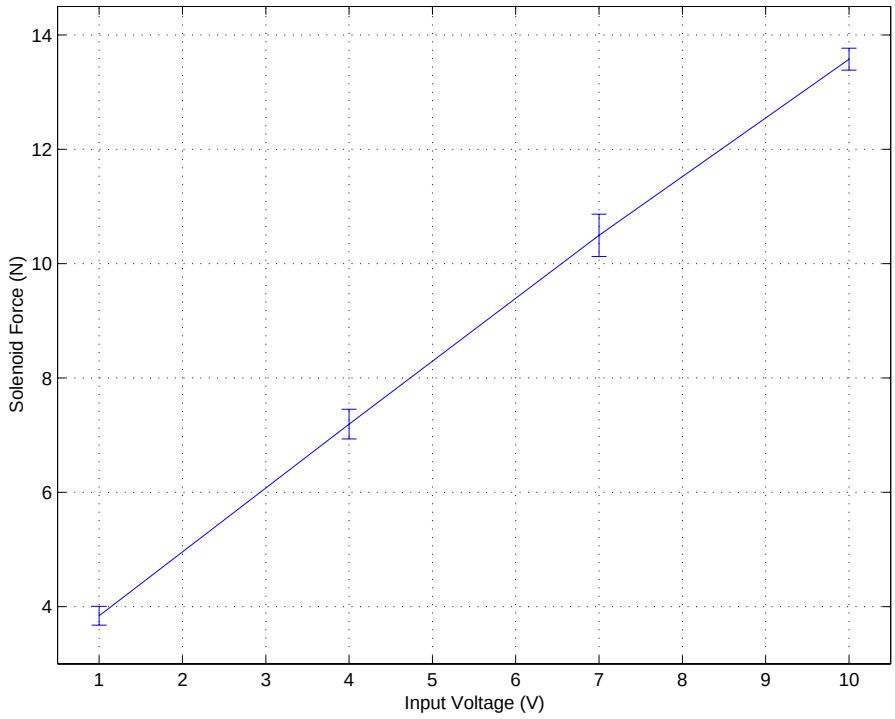

FIGURE 5.6 Voltage Input vs. Solenoid Force within Operating Range of Figure 5.5

This analysis shows that the solenoid force at the average displacements in the operating regime could accurately be modeled with a first order polynomial. Conservatively, to account for the error indicated by the bands in Figure 5.5, a fourth order polynomial is used to model the current-to-solenoid force function in the model. A higher order expression also allows flexibility in identification of different solenoids, should the pump or control module ever be upgraded. The values of the coefficients are chosen through the optimization described in Section 7.

6 Control Module, Rexroth Pump, and Hagglunds Motor

The model development for the hydro-mechanical portion of the drive train, consisting of a Rexroth pump and the EL control module, and the Hagglunds hydraulic motor, will be discussed in this section. The pump and its controls can be divided up into three sub-systems: (1) the spool valve assembly, (2) the stroking piston assembly, and (3) the swash plate assembly illustrated in Figure 6.1. A detailed view of the hydraulic ports surrounding the spool valve assembly can be found in Figure 6.6. The dashed line or lines labeled "neutral" in Figures 6.1 through 6.5, and in Figure 6.7, represent the neutral position for either the spool or swashplate. In depictions of the spool valve, when the spool is centered about this line it indicates that there is zero flow to the stroker; when the control arm is aligned with this line, the swashplate angle is zero. In illustrations of the swashplate, when the swashplate is centered about this neutral line this indicates that the motor speed is zero.

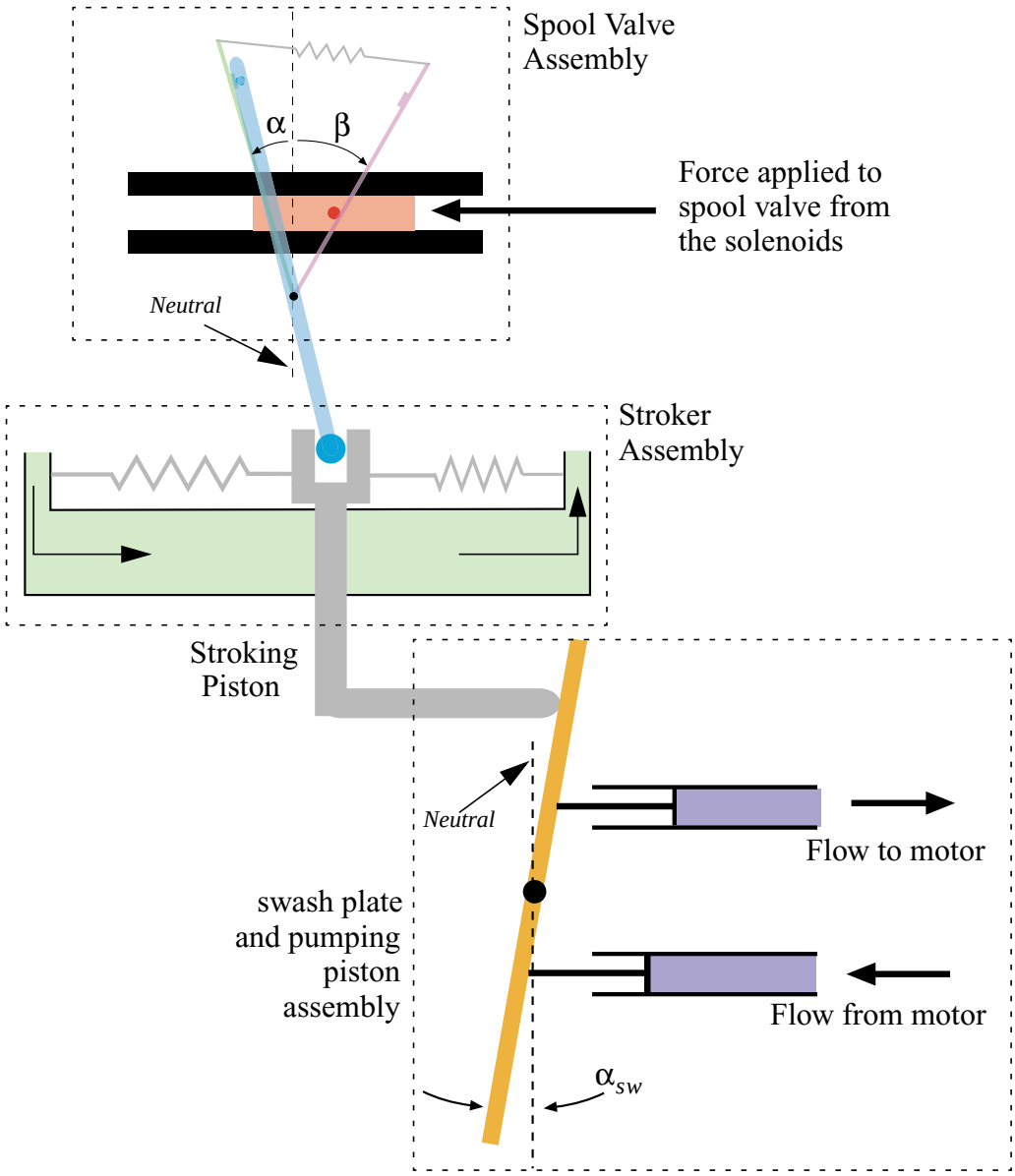

FIGURE 6.1 Rexroth Pump Diagram, Divided into the Three Assemblies

Prior to the development of equations, components within the pump and control module are defined and the general operation of the system is described.

6.1 Assembly Component Definition and Operation

The spool valve assembly, shown in Figure 6.2, is in direct contact with the solenoids described in Section 5. This assembly is made up of the spool valve and the mechanical feedback mechanism. This mechanism is composed of two feedback arms, a control arm

and a feedback spring. The ball at the end of the control arm is in contact with the stroking piston, and therefore, the swash plate as seen in Figure 6.3 and Figure 6.5. Although not evident from the planar representation of the spool valve and its mechanical feedback system drawing of Figure 6.2, the control arm does not lie in the same plane as the left feedback arm, right feedback arm, and feedback spring assembly. Its motion does not interfere with the spool pin motion.

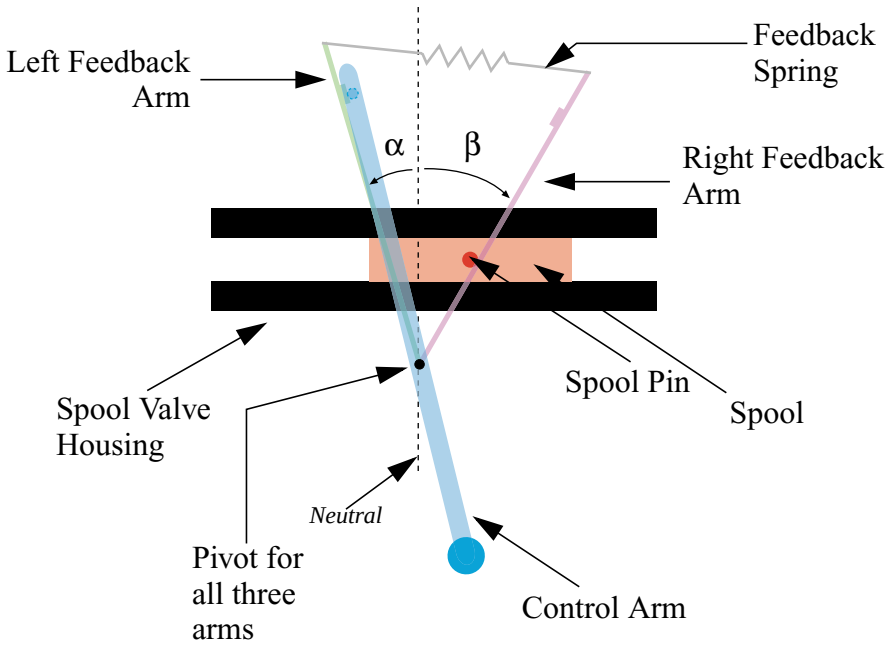

FIGURE 6.2 EL Control Module Spool Valve Assembly with Component Description

The purpose of the feedback mechanism is to communicate to the spool when the swash plate has achieved its commanded position, and thus that the motor has reached the desired speed. Note that the control arm is moved by the swash plate, thus α is assumed to be a prescribed displacement. The motor is now allowed to hold a constant speed as long as the commanded voltage remains constant. This effect is achieved through the control arm and the opposing forces of the solenoid and the feedback spring. Once the desired swash plate angle is achieved, an appropriate force is applied to the spool via the

stretch in the feedback spring. By taking advantage of the proportional aspect of the proportional solenoids, this balancing force allows the spool to hold various positions. This, in turn, varies the flow to the stroking piston, allowing a full range of swash plate angles. Figure 6.3 shows where the control arm interacts with the stroking piston and details the remainder of the system.

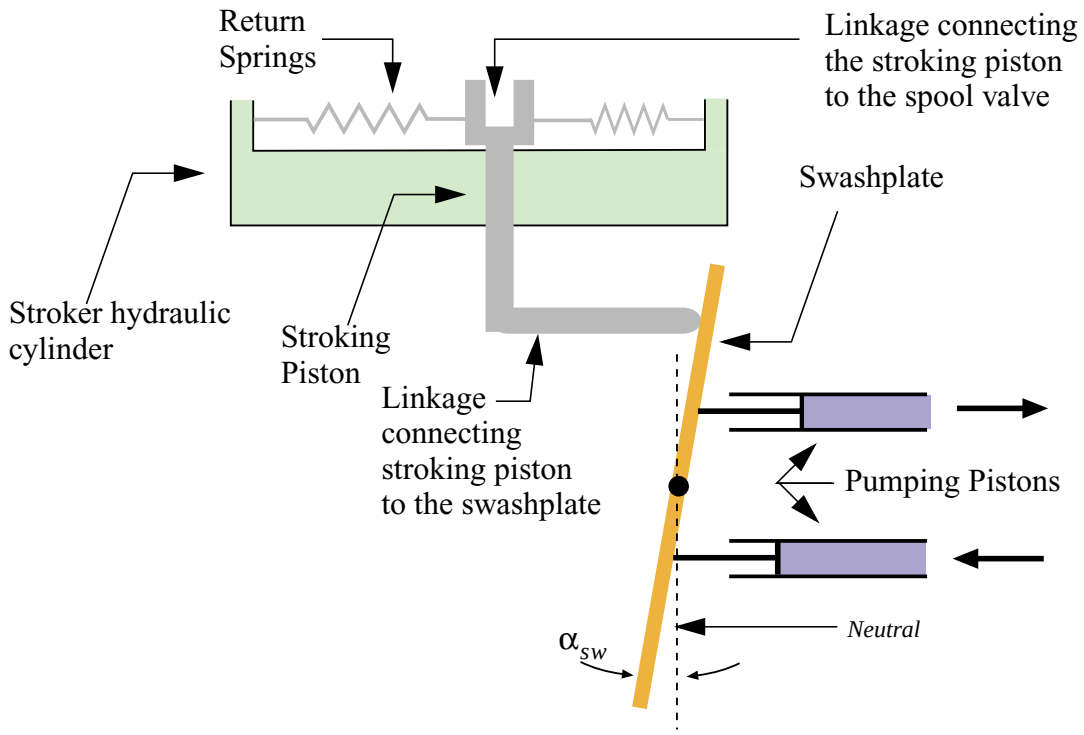

FIGURE 6.3 Stroker and Swash Plate Assemblies with Component Description

Consider the following example of the control module and pump's performance throughout a typical maneuver. When the spool is forced to the right due to a solenoid force, the right feedback arm rotates clockwise, and hydraulic fluid is allowed to flow to the stroking piston. As the piston moves it rotates the swash plate clockwise, as well as rotating the control arm counter-clockwise. The upper end of the control arm is in contact with the left feedback arm, causing it too to rotate counter-clockwise, thus resulting in additional stretch in the feedback spring. When the feedback spring force overcomes the solenoid

force, the spool valve begins moving left, back toward center. When the spool centers, flow is halted to the stroker leaving it, and the swash plate, in a fixed state, resulting in constant pump flow rate as long as the solenoid force remains constant.

If the operator zeros the solenoid force, the spool immediately moves to the left and the right feedback arm rotates counter-clockwise, maintaining contact with the spool pin. Flow to the stroking piston now reverses, causing the control arm, spool, and both feedback arms to rotate clockwise back to the neutral position. It should be noted that the dynamic behavior of the spool has two very different forms. When the spool is manipulated via a solenoid force, the dynamics of the stroker is effected by the combined effect of the stroker pressure dynamics and the feedback effect of the spool's feedback spring. When the solenoid force is zeroed, the dynamic behavior of the stroker is strictly due to the pressure dynamics.

The swash plate angle determines the output flow of the pump by setting the stroke of the pumping pistons. At full stroke, the swash plate angle is at its maximum, the pump is operating at full flow, and the hydraulic motor is at full speed. At zero stroke the swash plate is vertical, eliminating motion of the pistons, and stopping flow to the motor altogether. Thus, in a simplified model, motor speed could be considered proportional to the magnitude of the swash plate angle. As seen below, however, when dynamic effects and motor efficiency are considered, the relationship is more complex.

6.2 Dynamic Equations

The complete dynamic equations, where the states are the spool valve displacement, spool valve speed, stroker pressure, swash plate angle, and swash plate rate are derived below.

These are later simplified using the assumption that the forces acting on the spool (due to the solenoid and the feedback arm) and the forces acting on the swash plate (due to the stroker and the pumping pistons) dominate the dynamic behavior of the spool and the swash plate. Therefore, their dynamic equations are replaced by two force equilibrium equations.

6.2.1 Spool Valve

Included below for reference is the schematic of the spool valve assembly, labeled with the variables used in the model development.

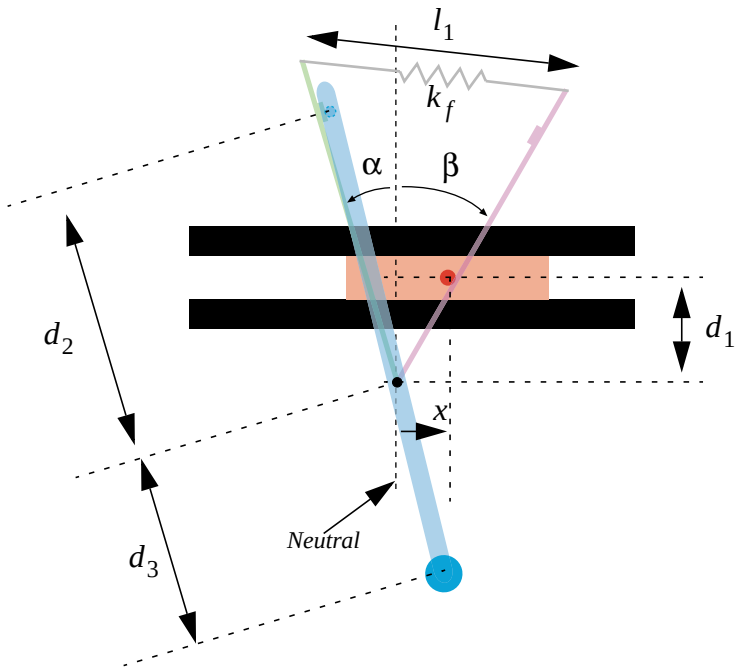

FIGURE 6.4 EL Control Module Showing Variables used for Model Development

The spool valve dynamic equation is developed using Lagrange's equations, assuming that the feedback arm has a prescribed displacement. In the following development, this means that α is prescribed by the stroker position and β is a degree of freedom. The equation of motion could also be derived with β as the prescribed input and α as the degree of freedom. However, when feedback arm variables are transformed to spool posi-

tion, the result is identical. Using the pivot as the origin of an inertial coordinate frame with the X-axis horizontal, and Y-axis vertical, the position vector to the center of the spool is

$$x = d_1 \tan\beta \tag{6.11}$$

and its speed is

$$\dot{x} = d_1 \dot{\beta}(1 + \tan^2\beta). \tag{6.12}$$

The kinetic energy is

$$T = \frac{1}{2} J_{fa} \dot{\alpha}^2 + \frac{1}{2} J_{fa} \dot{\beta}^2 + \frac{1}{2} m_{sp} \dot{x}^2 \tag{6.13}$$

where J_{fa} is the inertia of one feedback arm about the pivot and m_{sp} is the spool mass. The potential energy is due only to the stretch of the feedback spring and is

$$V = k_f d_3^2 [1 - \cos(\alpha + \beta)] \tag{6.14}$$

Applying Lagrange's equations:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\beta}}\right) - \frac{\partial L}{\partial \beta} = Q_\beta \tag{6.15}$$

where Q_β are the generalized forces, and the Lagrangian, L, is defined as

$$L = T - V \tag{6.16}$$

and expanding Equation 6.15 using Equation 6.13 and 6.14 gives

$$[J_{fa} + m_{sp} d_1^2 (1 + \tan^2\beta)^2]\ddot{\beta} + 2 m_{sp} d_1^2 \tan\beta (1 + \tan^2\beta)^2 \dot{\beta}^2 +$$

$$k_f d_3 \sin(\alpha + \beta) = d_1 F_{sol} (1 + \tan^2\beta) \tag{6.17}$$

where F_{sol} is the net force due to both solenoids. Using the relationship

$$\tan\beta = \frac{x}{d_1} \tag{6.18}$$

and writing the spool valve equation (6.17) in terms of the spool displacement, x, and the swash plate angle, α_{sw} gives

$$\frac{[m_{sp}(d_1^2 + x^2)^2 + d_1^2 J_{fa}]}{d_1(d_1^2 + x^2)}\ddot{x} - \frac{2J_{fa}d_1}{(d_1^2 + x^2)^2}x\dot{x}^2 + \\ + \frac{k_f d_3^2}{\sqrt{d_1^2 + x^2}}[d_1 \sin\alpha + x\cos\alpha] = \frac{(d_1^2 + x^2)}{d_1}F_{sol} \tag{6.19}$$

where the spool valve angle α is related to the swash plate angle α_{sw} by

$$\alpha = \sin^{-1}\left(\frac{d_4}{d_3}\tan\alpha_{sw}\right) \tag{6.20}$$

For reference, a schematic of the stroking piston and swash plate assemblies, labeled with the variables used in the model development, is included below.

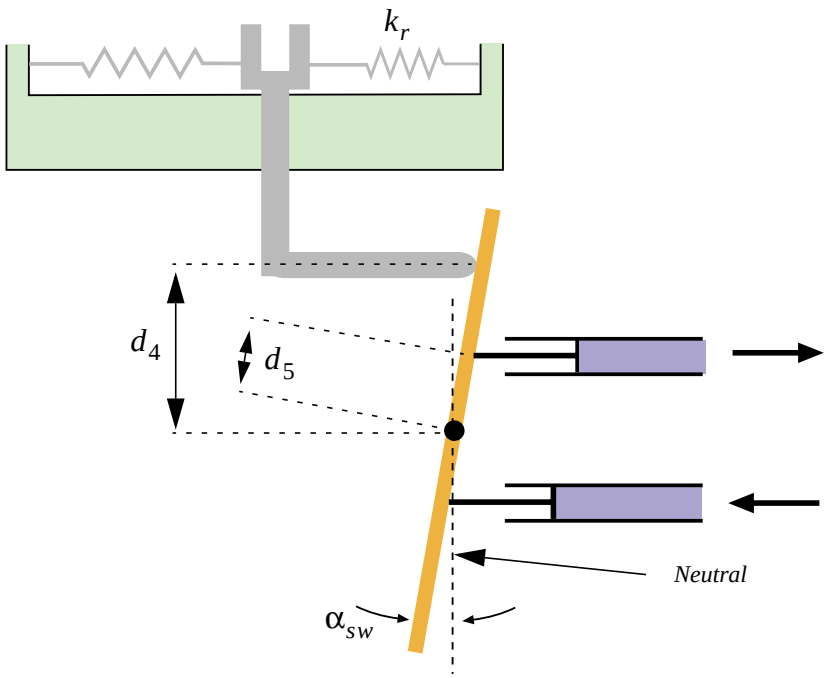

FIGURE 6.5 Stroker and swash plate Assemblies with Variables

6.2.2 Stroking Piston Pressure Equation

The pressure in the stroker is regulated by the spool valve. Modeling the spool valve requires an understanding of fluid flow through small orifices. The following development, including Equations through 6.30, can be found in texts covering hydraulic modeling, such as [8] and [21]. It is shown here, for completeness. Flow through hydraulic systems is generally considered to be turbulent flow at high Reynolds number [21]. There are two types of flow changes possible: a sudden expansion or a sudden contraction of the flow. In cases of expansion, the effect of flow separation is negligible. However, a sudden contraction of fluid flow can have a considerable effect and it is this situation which will be examined in terms of its effect on the stroker pressure equation. Figure 6.6 illustrates the hydraulic lines connecting the spool valve as can best be ascertained from the schematics, and the stroker cylinder, as well as a simplification of some of the mechanical connections.

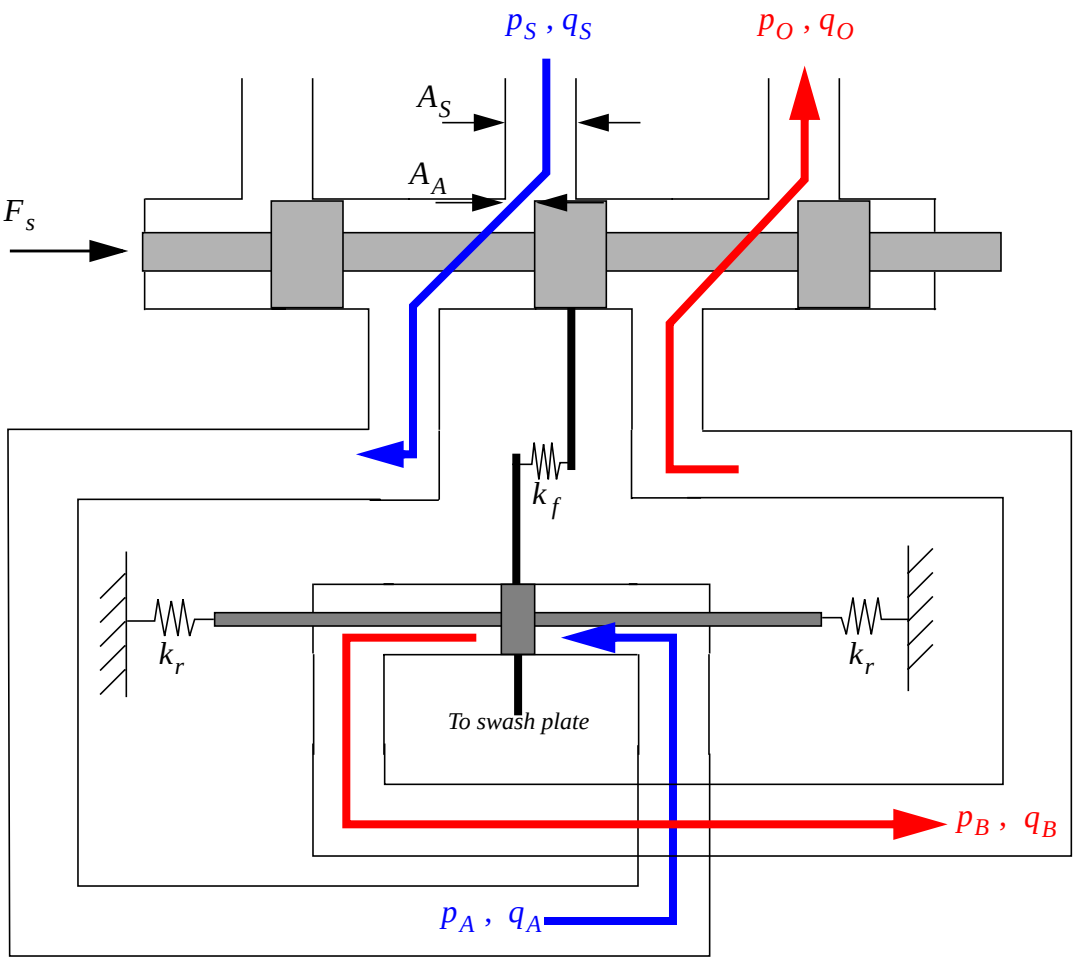

FIGURE 6.6 Schematic of Hydraulic Lines Between Valve and Stroking Piston

Two basic equations are used to relate volumetric supply flow rate q_S, and the volumetric flow rate after the orifice q_A. A conservation of energy equation can be written as,

$$\frac{1}{2}v_S^2 + \frac{1}{\rho}p_S = \frac{1}{2}v_A^2 + \frac{1}{\rho}p_A \tag{6.21}$$

and a flow continuity equation can be applied,

$$v_S A_S = v_A A_A \tag{6.22}$$

where p_S is the supply pressure, p_A and p_B are pressures acting on opposite sides of the stroker, and v_S and v_A are the corresponding velocities. Equation 6.21 assumes that the valve is level and operating at zero potential energy.

Solving Equation 6.21 and 6.22 simultaneously, an expression (6.23) is achieved for the volumetric flow rate after the orifice,

$$q_A = v_A A_A = \frac{A_A}{\sqrt{1-\left(\frac{A_A}{A_S}\right)^2}}\sqrt{\frac{2}{\rho}(p_S - p_A)} \tag{6.23}$$

The vena contracta, the cross-sectional flow area after the orifice, (A_A), is generally accepted to be smaller than the orifice area. For simplification, it will be assumed that A_A is equal to the size of the opening. Note, the term 'orifice' as it is used in this section describes the opening in the valve created by spool displacement. Future discussion of orifices is in reference to flow limiting orifices with constant diameters which restrict flow to the stroker.

To account for friction in duct flow, a dimensionless discharge coefficient, C_d is added

$$q_A = C_d\frac{A_A}{\sqrt{1-\left(\frac{A_A}{A_S}\right)^2}}\sqrt{\frac{2}{\rho}(p_S - p_A)} \tag{6.24}$$

similarly,

$$q_B = C_d\frac{A_S}{\sqrt{1-\left(\frac{A_A}{A_S}\right)^2}}\sqrt{\frac{2}{\rho}(p_B - p_O)} \tag{6.25}$$

Assuming C_d on the supply side is equal to that of the return side and equating q_A to q_B gives

$$p_s - p_A = p_B - p_O \tag{6.26}$$

Defining the pressure differentials as

$$\Delta P \equiv p_s - p_O$$

$$P \equiv p_B - p_A \tag{6.27}$$

and solving for the two return pressures gives

$$p_O = -\Delta P + p_S$$

$$p_B = p_A - P \tag{6.28}$$

Assuming the supply and reservoir pressures to be constant, Equation 6.28 is substituted into 6.25

$$q_B = \frac{C_d A_S \sqrt{\frac{2}{\rho}(p_A - P + \Delta P - p_S)}}{\sqrt{1 - \left(\frac{A_A}{A_S}\right)^2}} \tag{6.29}$$

recalling $q_A \equiv q_B$, the total flow rate through the spool valve is defined as

$$q = \frac{1}{2}(q_A + q_B) \tag{6.30}$$

Relating the flow equation, 6.30, directly to the spool valve displacement with the assumption that the flow area is proportional to spool displacement and expanding gives

$$q = \frac{C_d k x \sqrt{\frac{1}{2\rho}(\Delta P - P)}}{\sqrt{1 - \left(\frac{kx}{A_S}\right)^2}} \tag{6.31}$$

where q is now the total volumetric flow rate across the spool valve.

The total flow, q, can be broken down into three additive components:

$$q = q_v + q_c + q_L \tag{6.32}$$

These three components; (1) flow due to volume change, (2) the fluid compression contribution, and (3) flow due to leakage around the stroking piston, respectively, are defined as

$$\begin{aligned} q_v &= A_p \dot{x}_{st} \\ q_c &= \frac{V}{B}\dot{P} \\ q_L &= K_{swL} P \end{aligned} \tag{6.33}$$

where A_p is the cross sectional area of the stroking piston, V is the volume of the fluid under pressure in the stroking cylinder, B is the bulk modulus of the hydraulic fluid, and K_{swL} is the coefficient of leakage around the stroking piston. The rate of stroking piston displacement, $\dot{x}_{st}$, is related to the rate of change of the swash plate angle $\dot{\alpha}$ through the relationship derived from Figure 6.5

$$\dot{x}_{st} = d_4 \dot{\alpha}_{sw}(1 + \tan^2 \alpha_{sw}) \tag{6.34}$$

Equating 6.31 and 6.32, and incorporating 6.34 gives the final equation for the pressure in the stroker, P,

$$\dot{P} + \frac{K_{swL} B}{V} P + \frac{A_p B}{V} d_4 (1 + \tan^2 \alpha_{sw}) \dot{\alpha}_{sw} = \frac{C_o B}{V}\left(\sqrt{\frac{1}{2}(\Delta P - P)}\right) x_{eff} \tag{6.35}$$

where x_{eff} is the effective displacement of the spool valve from the centered position. The effective spool displacement is related to the true spool displacement by

$$x_{eff} = \begin{cases} x_{max} & x \geq x_{max} \\ x & -x_{max} > x > x_{max} \\ -x_{max} & x \leq -x_{max} \end{cases} \tag{6.36}$$

and captures the possibility that the spool valve moves beyond the port width in the spool valve body. Spool widths are assumed to be equal to port widths with no overlap.

C_o is an expanded flow coefficient,

$$C_o = \frac{C_d k}{\sqrt{\rho\left\{1 - \left(\frac{kx}{A_S}\right)^2\right\}}} \tag{6.37}$$

where the term containing spool displacement in the denominator, $\left(\frac{kx}{A_S}\right)^2$, is assumed to be small compared to one thereby allowing C_o to be modeled as a constant.

6.2.3 Swash Plate Dynamic Equation

The swash plate is acted on by the stroking piston force F_{st} and the load pressure. The free body diagram of the swash plate (Figure 6.7) shows the load forces where A_p is the effective pumping piston area acting on the swash plate, P_{hi} is the pumping pressure, and P_{lo} is the return side, which captures the pressure drop across the hydraulic motor.

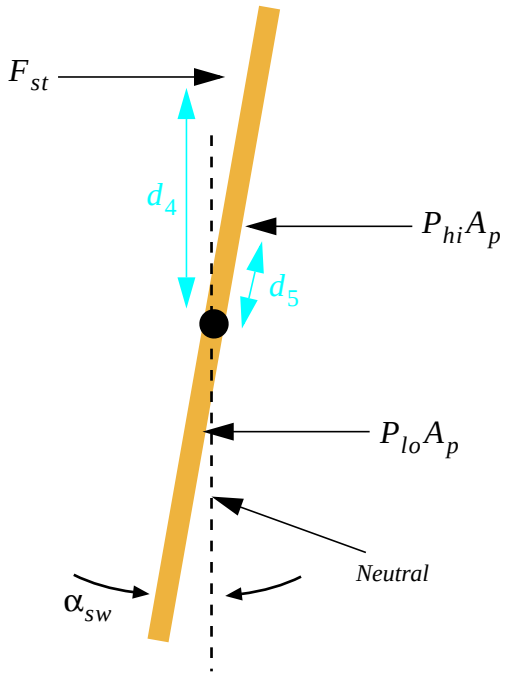

FIGURE 6.7 Free Body Diagram of Swash Plate

Considering the free body diagram of the swash plate, the dynamic equation is readily obtained as

$$J_{sw}\ddot{\alpha}_{sw} + (P_{hi} - P_{lo})A_p d_5 \cos\alpha_{sw} = F_{st} d_4 \tag{6.38}$$

The speed of the Hagglunds hydraulic motors (84-25100 and 64-16300) are modeled as simply being proportional to the pump flow rate

$$\Omega_m = K_{mf} Q \tag{6.39}$$

where Ω_m is the motor speed, Q is the fluid flow rate from the pump, and K_{mf} is the hydraulic motor constant. The motor pressure equation is obtained by considering the work done by the motor

$$QP_{hi} = QP_{lo} + \tau\Omega_m \tag{6.40}$$

where τ is the load torque.

Combining the two motor equations, Equations 6.39 and 6.40, gives

$$P_{hi} - P_{lo} = K_{mf}\tau \tag{6.41}$$

assuming τ is approximately constant.

The expression for the load torque is axis dependent. In general, the load torque will be expressed as

$$\tau = C_i\dot{\Omega}_m + C_g \tag{6.42}$$

where the values of C_i and C_q depend on the axis being considered. For slew there is no gravitational effect and therefore C_g is zero.

Substituting Equation 6.42 into 6.41 yields

$$P_{hi} - P_{lo} = K_{mf}(C_i\dot{\Omega}_m + C_g) \tag{6.43}$$

Next we assume that the flow from the pump is related to the swash plate angle according to

$$Q = C_q \sin\alpha_{sw} \tag{6.44}$$

where C_q is a constant capturing the pumping cylinder size and geometry.

Substituting Equation 6.44 into 6.39, the swash plate angle can be related to the motor speed as

$$\begin{aligned} \Omega_m &= K_{mf}C_q \sin\alpha_{sw} \\ \dot{\Omega}_m &= K_{mf}C_q\dot{\alpha}_{sw}\cos\alpha_{sw} \end{aligned} \tag{6.45}$$

Substituting 6.45 into Equation 6.43 gives

$$P_{hi} - P_{lo} = K_{mf}[C_iC_qK_{mf}\dot{\alpha}_{sw}\cos\alpha_{sw} + C_g] \tag{6.46}$$

Substituting Equation 6.46 into Equation 6.38 gives a new version of the swash plate equation where the pressure drop across the motor has been resolved out

$$J_{sw}\ddot{\alpha}_{sw} + K_{mf}[C_i C_q K_{mf}\dot{\alpha}_{sw}\cos\alpha_{sw} + C_g]A_p d_5 \cos\alpha_{sw} = F_{st} d_4 \tag{6.47}$$

Looking at the free body diagram of the stroking piston in Figure 6.8,

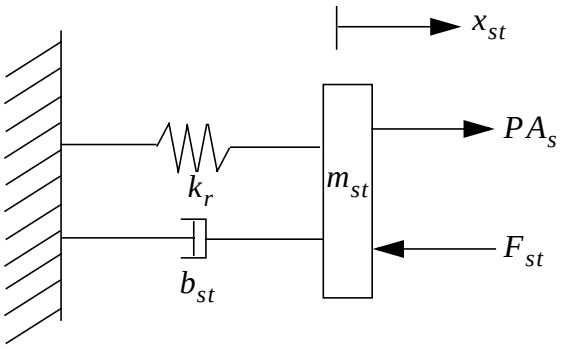

FIGURE 6.8 Free Body Diagram of Stroking Piston

the stroking piston equation is simply written as

$$m_{st}\ddot{x}_{st} + b_{st}\dot{x}_{st} + k_r x_{st} = PA_s - F_{st} \tag{6.48}$$

where some viscous damping has been added using the coefficient b_{st}. Recall that k_r is the return spring stiffness. Using the relationships between the swash plate angle, α_{sw}, and the stroking piston displacement, x_{st}, obtained from Figure 6.5,

$$\begin{aligned} \frac{x_{st}}{d_4} &= \tan\alpha_{sw} \\ \dot{x}_{st} &= d_4\dot{\alpha}_{sw}(1 + \tan^2\alpha_{sw}) \\ \ddot{x}_{st} &= d_4\ddot{\alpha}_{sw}(1 + \tan^2\alpha_{sw}) + 2d_4\tan\alpha_{sw}(1 + \tan^2\alpha_{sw})\dot{\alpha}_{sw}^2 \end{aligned} \tag{6.49}$$

the stroker equation, 6.48, can be written in terms of the swash plate angle and the stroker pressure:

$$m_{st}d_4(1+\tan^2\alpha_{sw})\ddot{\alpha}_{sw} + 2m_{st}d_4\tan\alpha_{sw}(1+\tan^2\alpha_{sw})\dot{\alpha}_{sw}^2 + b_{st}d_4(1+\tan^2\alpha_{sw})\dot{\alpha}_{sw} + k_r d_4\tan\alpha_{sw} - PA_s = -F_{st} \quad (6.50)$$

Finally, substituting Equation 6.50 into the swash plate equation, 6.47, gives

$$[J_{sw} + m_{st}d_4^2(1+\tan^2\alpha_{sw})]\ddot{\alpha}_{sw} + 2m_{st}d_4^2\tan\alpha_{sw}(1+\tan^2\alpha_{sw})\dot{\alpha}_{sw}^2 + [K_{mf}^2 C_i C_q A_p d_s\cos^2\alpha_{sw} + b_{st}d_4^2(1+\tan^2\alpha_{sw})]\dot{\alpha}_{sw} + k_r d_4^2\tan\alpha_{sw} = PA_s d_4 - K_{mf}C_g A_p d_5\cos\alpha_{sw} \quad (6.51)$$

6.2.4 Model Simplification through Force Equilibrium

One approach for simulating the pumps would be to use the three dynamic Equations 6.35, and 6.51, resulting in a 5 state model of the system. Due to the small mass and high forces acting on the both the spool and swash plate, the x and α_{sw} degrees of freedom have high frequency content (approximately 40 Hz) compared to the low frequency motion of the winches (roll off at approximately 0.3 Hz). Therefore, both the spool and swash plate dynamic equations are replaced with force equilibrium equations. This leaves only the stroker pressure equation (1 state) which can be simulated with greater speed and efficiency. In addition, the stroker leakage will be assumed to be negligible and the pressure equation will be expanded about the static equilibrium pressure, P_0:

$$P = P_0 + \delta P \quad (6.52)$$

From Equation 6.51, the swash plate force equilibrium expression is

$$k_r d_4^2\tan\alpha_{sw} = (P_0 + \delta P)A_s d_4 - K_{mf}C_g A_p d_5\cos\alpha_{sw} \quad (6.53)$$

and from the static equilibrium equation at $\delta P = 0$

$$P_0 \approx \frac{K_{mf} C_g A_p d_5}{A_s d_4} \tag{6.54}$$

for $\alpha_{sw_0} \ll 1$, the static equilibrium swash plate angle, Equation 6.46 can be solved for α_{sw}

$$\alpha_{sw} = \tan^{-1}(Z_1 \delta P - Z_2) \tag{6.55}$$

where

$$\begin{aligned} Z_1 &= \frac{A_s}{k_r d_4} \\ Z_2 &= \frac{K_{mf} C_g A_p d_5}{k_r d_4^2} - P_0 = 0 \end{aligned} \tag{6.56}$$

From Equation the spool force equilibrium expression is

$$x = \frac{-Z_5 Z_4 \sin\alpha\cos\alpha + Z_5^2 F_{sol}\sqrt{Z_4 - Z_5^2 F_{sol}^2}}{Z_4 \cos^2\alpha - Z_5^2 F_{sol}^2} + Z_{18} \tag{6.57}$$

where it is assumed that $\tan^2\beta \ll 1$ and defining

$$\begin{aligned} Z_4 &= k_f^2 d_3^4 \\ Z_5 &= d_1 \end{aligned} \tag{6.58}$$

and Z_{18} as the plussing. Plussing is an external adjustment on the control module that attempts to compensate for the gravitational effects on the hoist and luff axes. In a properly adjusted system, the plussing will keep the payload or boom stationary even when a moderate load is applied. This adjustment is modeled by giving the spool valve a nonzero displacement when the solenoid force is zero. Although the solution of the quadratic for *x*

yields two solutions, the positive solution was selected as it is the only physically consistent one.

Introducing the lumped parameter, Z_3, Equation 6.20 is written as

$$\alpha = \sin^{-1}(Z_3 \tan\alpha_{sw}) \tag{6.59}$$

where

$$Z_3 = \frac{d_4}{d_3} \tag{6.60}$$

F_{sol} is related to the input current as described in Section 5. This relationship will be written as

$$F_{sol} = \begin{cases} 0 & i \le Z_{17} \\ Z_{22}(i - Z_{17})^4 + Z_{21}(i - Z_{17})^3 + Z_{20}(i - Z_{17})^2 + Z_{19}(i - Z_{17}) & i > Z_{17} \end{cases} \tag{6.61}$$

where use of Z_{19}, Z_{20}, Z_{21}, and Z_{22} is the nomenclature used in the parameterization. It should be noted that there are two sets of coefficients, one set for positive currents and the other for negative currents. Section 6.2.5 defines Z_{17}.

The pressure dynamic equation requires the swash plate angle states. Normally, this will require an analytical expression for $\dot{\alpha}_{sw}$, unless the swash plate hits a hard stop. To accommodate this case consider the swash plate force equilibrium Equation 6.55 where $Z_2 = 0$

$$\tan\alpha_{sw} = Z_1 \delta P \tag{6.62}$$

where solving for the swash plate rate gives

$$\dot{\alpha}_{sw} = \frac{Z_1}{1 + \tan^2\alpha_{sw}} \delta \dot{P} \tag{6.63}$$

which is valid when the swash plate is not at a hardstop. Of course, $\dot{\alpha}_{sw} = 0$ when the swash plate hits either the positive or negative maximum. Substituting this expression into the pressure equation (6.35) along with Equation 6.52 gives

$$\delta\dot{P} = -\frac{K_{swL}B}{V}(P_0 + \delta P) - \frac{A_s B d_4 Z_1}{V}\delta\dot{P} + \frac{C_o B}{V}\left(\sqrt{\frac{1}{2}(\Delta P - P_0 - \delta P)}\right)x_{eff} \tag{6.64}$$

In static equilibrium, at $\delta P = 0$

$$\frac{K_{swL}}{V}P_0 = \frac{C_o\beta}{V}\sqrt{\frac{1}{2}(\Delta P - P_0)}x_0 \tag{6.65}$$

Assuming K_{swL} is zero implies $x_0 = 0$ at static equilibrium.

Solving Equation 6.64 for the pressure rate gives

$$\left(1 + \frac{A_s B d_4 Z_1}{V}\right)\delta\dot{P} = \frac{C_o B}{V}\left(\sqrt{\frac{1}{2}(\Delta P - P_0 - \delta P)}\right)x_{eff} \tag{6.66}$$

The various unknown parameters can be lumped as before to give

$$\delta\dot{P} = \frac{Z_8}{1 + Z_6}\left(\sqrt{\frac{1}{2}(Z_9 - \delta P)}\right)x_{eff} \tag{6.67}$$

where

$$Z_6 = \begin{cases} \dfrac{A_s^2 B}{k_r V} & -\alpha_{sw,\,max} < \alpha_{sw} < \alpha_{sw,\,max} \\ 0 & -\alpha_{sw,\,max} \geq \alpha_{sw} \geq \alpha_{sw,\,max} \end{cases}$$

$$Z_7 = \frac{K_{swL}B}{V} \qquad Z_8 = \frac{C_o B}{V} \qquad Z_9 = \Delta P - P_0 \tag{6.68}$$

The output equation, relating swash plate angle to motor speed is

$$\Omega = Z_{10}\alpha_{sw} - Z_{11} - Z_{12}\cos(\alpha_{sw})\delta\dot{P} \tag{6.69}$$

where Z_{10} is a constant associated with the motor displacement, and Z_{11} and Z_{12} capture load dependent leakage derived from the relationship between the pressure drop across the motor and the stroker pressure and are given by

$$Z_{11} = K_{mL} C_g \; Z_{12} = K_{mL} K_{mf} C_i C_q \tag{6.70}$$

The final version of the dynamic equations allows the model to be run at a speed sufficient for use in an optimization code to ascertain the numerous lumped parameters in the model. Optimization of these parameters is described in Section 7.

6.2.5 Model Nonlinearities

Although the pressure dynamic equations are nonlinear in themselves, more dramatic nonlinear effects are introduced by the limits imposed upon the model variables. The swash plate is limited based on a nominal value of 15 degrees, resulting in a motor speed limit. The stroker pressure is assumed to be limited via a relief valve, which is engaged whenever the swash plate limits. Without this feature in the model, the stroker pressure will continue to build after the swash plate limits. This pressure build-up causes the swash plate to remain at a maximum position long after any current command has gone to zero. The spool valve is displacement limited, but the effects of the in-line orifices between the valve and the stroker are seen before this happens. The flow limiting orifices are incorporated as an effective limit that occurs prior to the spool hitting its hard limit.

$$x_{eff} = \begin{cases} Z_{16} & x > Z_{16} \\ x & x \le Z_{16} \end{cases} \tag{6.71}$$

where Z_{16} represents the displacement of x which would produce a valve opening equal to the orifice size. Further increase in the valve opening results in no greater flow due to the orifice restriction. Modeling these flow limiting orifices is an important aspect of this

model's flexibility. They are designed to be easily switched out for orifices of different diameters by the operator should different performance characteristics ever be desired. A larger orifice would result in the swash plate's capability to move from zero to maximum deflection more quickly. This would be reflected by a significant increase in the motor acceleration limit. The capability to model the hard limit on spool displacement is left in the model to capture the ultimate acceleration should the flow limiting orifices be removed entirely. Likewise, as expressed in Equation 6.36, the possibility that the port width is the limiting factor is also retained. Finally, a deadzone in the input current had an affect on the solenoid motion.

The same Z nomenclature for parameterization is used to tabulate and define these limits in Table 6.1.

Table 6.1 Nonlinear Parameter Definitions

Parameter	Description
Z_{13}	Swash plate limit
Z_{14}	Spool valve hard limit
Z_{15}	Pressure limit
Z_{16}	Effective orifice length
Z_{17}	Solenoid deadzone

It should be noted that Z_{13} and Z_{16} will have 2 different values depending on the sign of the swash plate angle and spool valve respectively.

6.2.6 Small Motion Linearization

Although the equations used in the model identification are those described in Section 6.2.4, it may be beneficial to use equations linearized for small values of the swash plate angle α_{sw} for initial control design. The effort limiting nonlinearities, including swash plate limits, spool valve limits, and pressure limits, of Section 6.2.5 would still need to be considered in such an analysis.

The linearized equations are

$$x = \frac{-d_1 d_3^4 k_f^2 \alpha_{sw} + d_1 F_{sol}\sqrt{k_f^2 d_3^4 - d_1^2 F_{sol}^2}}{k_f^2 d_3^4 - d_1^2 F_{sol}} \tag{6.72}$$

$$\alpha_{sw} = \frac{A_s}{k_r d_4}\delta P \tag{6.73}$$

$$\delta\dot{P} = \frac{C_o B}{V}\sqrt{\frac{1}{2}(\Delta P - P_o)}\alpha_{sw} \tag{6.74}$$

The output equation, relating swashplate angle to winch speed of Equation 6.69.

The output equation, relating swash plate angle to winch speed is still that of Equation 6.69

When none of the nonlinearities of Section 6.2.5 are present, then a first order transfer function relating amplifier current to motor speed can be formed, which may be helpful for designing an outer servo loop on speed.

7 Parameter Identification, Pump/Motor Model

The parameters Z_1 through Z_{22} were determined for hoist, slew, and luff using an optimization approach. The cost function was formed as the sum of the integral square error between simulated and measured motor speed for ramp, step, and sine wave voltage inputs. Mathematically,

$$J = \sum_{i=1}^{n} J_i \tag{7.1}$$

where J is the cost, and i represents a particular data set (e.g. 1 = 4V step, 2 = 6.5V step, etc.). The optimization searches for the Z^* that minimizes J. The block diagram in Figure 7.1 illustrates the optimization process.

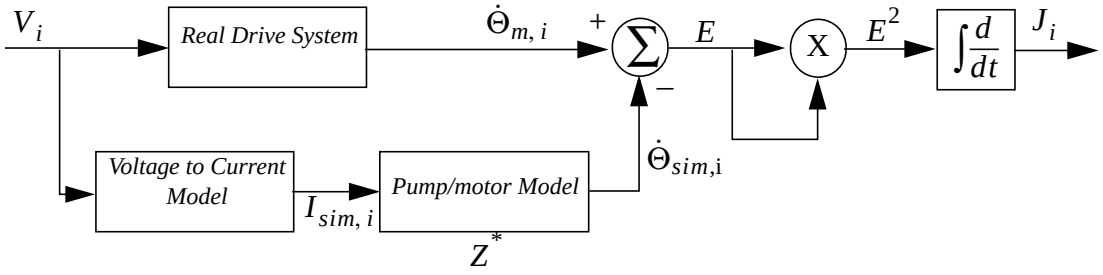

FIGURE 7.1 Block Diagram of Optimization

The initial estimate of Z^* was based on measurements or specifications where available and each subsequent choice was determined by an optimization code using the recursive quadratic programming method. Section 7.1 describes the process of converting the measured motor position to a motor speed. Section 7.3 through 7.5 show the results of the model output for each axis. The cost function code (elwrap_hoist2.m and elcost.c) and its setup file (elset.m) for the hoist axis are included in Appendix E and Appendix D, respectively. Only the initial estimates differ in the files for the other axes.

The optimized pump/motor parameters are given in Table 7.2, 7.3, and 7.4. This system showed several performance characteristics which differed when the motor was run in forward versus when run in reverse. For instance, the swash plate hard stop (Z13) requires two values to characterize performance in both directions. The coefficients in the equation relating current to solenoid force (Z19-22) are also fit by two sets of parameters. For each of these cases when dual parameterization is necessary, the parameter which goes with a positive voltage input is listed first. The notation in the table is positive/negative. If the parameter is the same for both positive and negative voltage inputs, then only one value is listed.

The Rexroth control card model of Section 3 was used to generate simulated currents based on the same voltage histories used during the operational testing. The model results compare simulated motor speed output to measured motor speeds using ramp, step, and sine wave voltage inputs. The voltage signal is included to illustrate the lags and roll off features.

7.1 Encoder Calibration

For slew, the 81,000 count (with quadrature) turret encoder signal was used to estimate turret speed,

$$\Omega_{turret} = \frac{\frac{d}{dt}E_{slew}}{81000(4)N_{slew}} \tag{7.2}$$

where N_{slew} is the gear ratio between the encoder and the turret, and E_{slew} is the count time history. The differentiation of encoder count time histories is discussed in the following section.

The encoders for hoist and luff were used to extract their motor speeds as well

$$\Omega_{luff} = \frac{\frac{d}{dt}E_{luff}}{100(4)N_{luff}} \tag{7.3}$$

$$\Omega_{hoist} = \frac{\frac{d}{dt}E_{hoist}}{635(4)N_{hoist}} \tag{7.4}$$

where E_{hoist} and E_{luff} are encoder time histories and N_{hoist} and N_{luff} are the gear ratios between the motors and their encoders. The gear ratios for all three axes are given in Table 7.1. It should be noted that the slew drive model generates the turret speed. The slew motor speed can be related to the slew turret speed by

$$\Omega_{slew,\,motor} = 3150.1\Omega_{turret} \tag{7.5}$$

Table 7.1 Axis Dependent Gear Ratios

Axis	N
slew	7
luff	120/19
hoist	1

7.2 Encoder Differentiation

Differentiation of encoder data is not a simple task due to the discontinuities occurring at each new pulse.

Extracting a smooth speed signal from encoder data can be especially difficult when the sample rate of the data acquisition system is near, or less than, the frequency of the pulse changes. Mathematically,

$$h < \frac{N_{enc}\omega}{2\pi} \tag{7.6}$$

where h is the data acquisition sample period, N_{enc} is the encoder resolution in counts/rev, and ω is the angular rate of the encoder in rad/sec. In this situation a new position is only recorded after several samples. Unfortunately, the sample rate is less than the pulse change frequency for low motor speeds. This requires the encoder data to be interpolated to facilitate smoother derivative calculations. A sample comparison between the interpolated or 'smoothed' data and the raw encoder data is included in Figure 7.2.

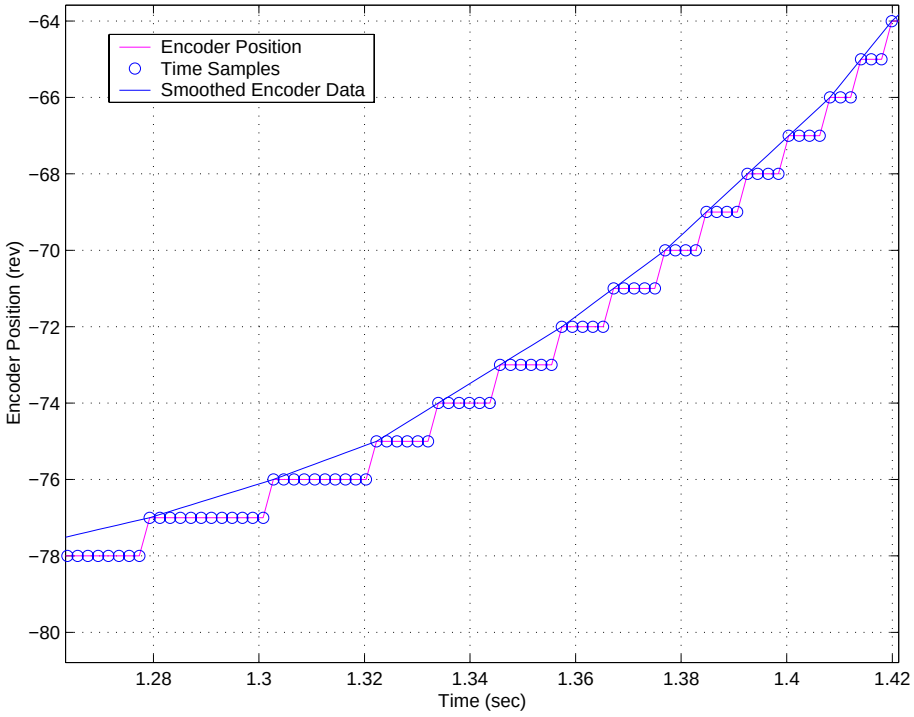

FIGURE 7.2 Comparison between Raw and Smoothed Encoder Data

The smoothing function (smthenc.m-- MATLAB code included in Appendix A) simply monitors the encoder count time history for a change in position, then calculates the slope created by this point and the point marked by the previous change in position. The interpolated values for the intermediate time steps are then filled in.

After being smoothed, the encoder data was low pass filtered, and then run through a band limited differentiating filter as seen in the block diagram below.

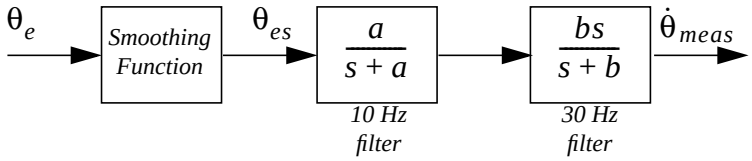

FIGURE 7.3 Encoder Differentiation Block Diagram

where

$$a = 10(2\pi)$$
$$b = 30(2\pi) \tag{7.7}$$

The parameter identification strategy is sensitive to phase shift errors between measured and simulated motor speed time histories. To avoid this, all simulated motor speeds are also run through the low pass filters with the same 10 and 30 Hz cut-off frequencies. However, the output of the pump/motor model is already a rotational speed and therefore the zero is removed from the 30 Hertz differentiating filter as shown in Figure 7.4.

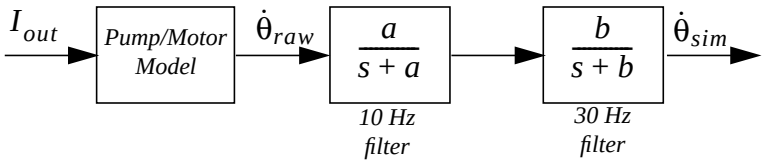

FIGURE 7.4 Block Diagram of Simulated Motor Speed Filtration

7.3 Hoist Results

The data set used for optimization and shown in the figures below were performed with no load attached to the hook block. Each maneuver is brought to rest through a cubic spline with a duration of 2.5 seconds if necessary. A positive voltage input indicates hoisting up.

Table 7.2 Optimized Parameters, Hoist Axis

Parameter	Optimized Value	Parameter	Optimized Value	Parameter	Optimized Value
Z_1	4.6441e-07	Z_8	1.6514e+06	Z_{15}	4.1191e+06
Z_2	0	Z_9	1.0743e+07	Z_{16} +/-	1.1634e-04/ -1.5126e-04
Z_3	0.62542	Z_{10}	12.396	Z_{17}	0.27867
Z_4	0.053604	Z_{11}	0	Z_{18}	2.1667e-05
Z_5	0.0079568	Z_{12}	0	Z_{19} +/-	12.349/ 10.019
Z_6	0.073926	Z_{13} +/-	0.27596/ -0.26047	Z_{20} +/-	0.27028/ 3.7297
Z_7	0	Z_{14}	0.0036461	Z_{21} +/-	0.22421/ 1.6581
				Z_{22} +/-	0/ -54.153

The following figures show simulated winch speeds compared with measured winch speeds. The voltage input signal is also included for reference.

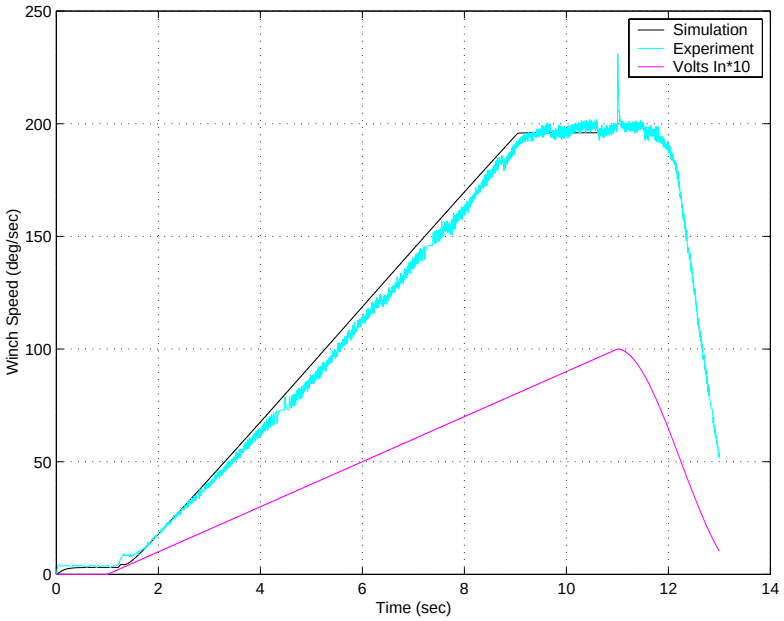

FIGURE 7.5 Hoist Winch Speed Data, 1V/sec Ramp (hoistr3.dat)

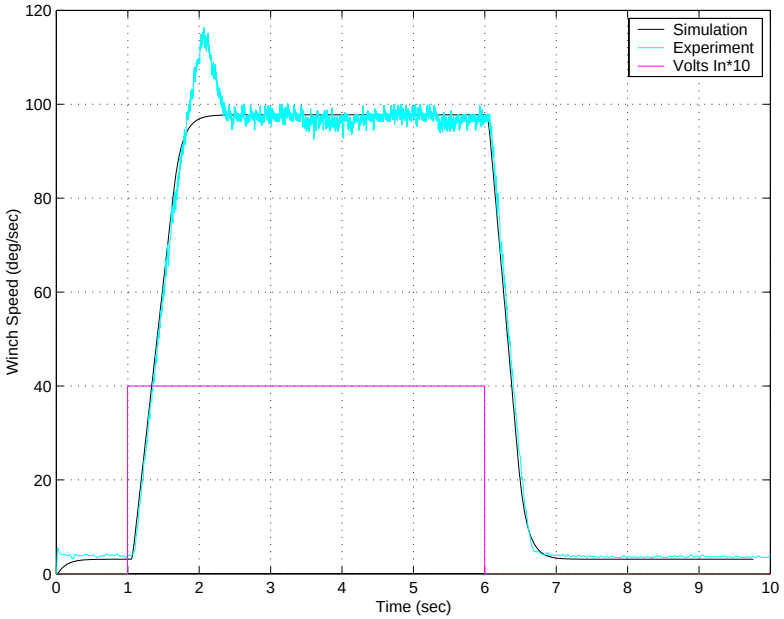

FIGURE 7.6 Hoist Winch Speed Data, 4V Step (hoistr6.dat)

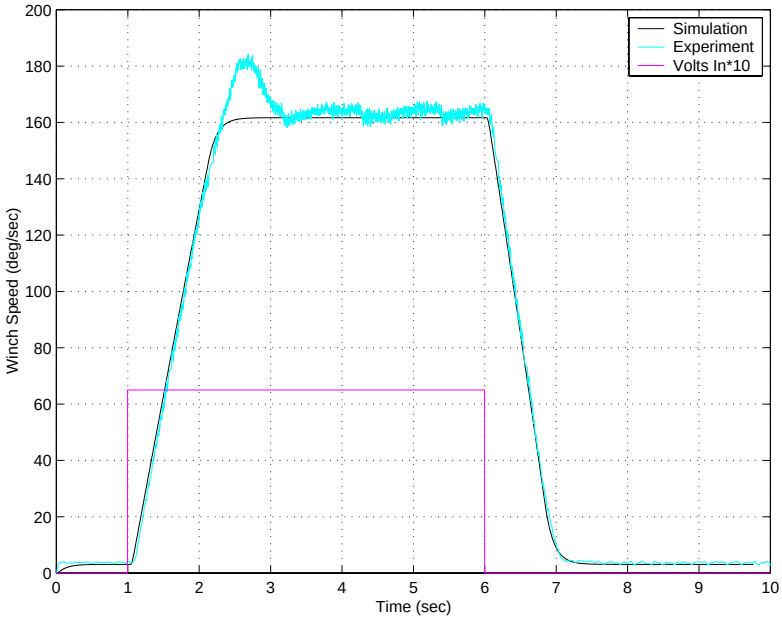

FIGURE 7.7 Hoist Winch Speed Data, 6.5V Step (hoistr8.dat)

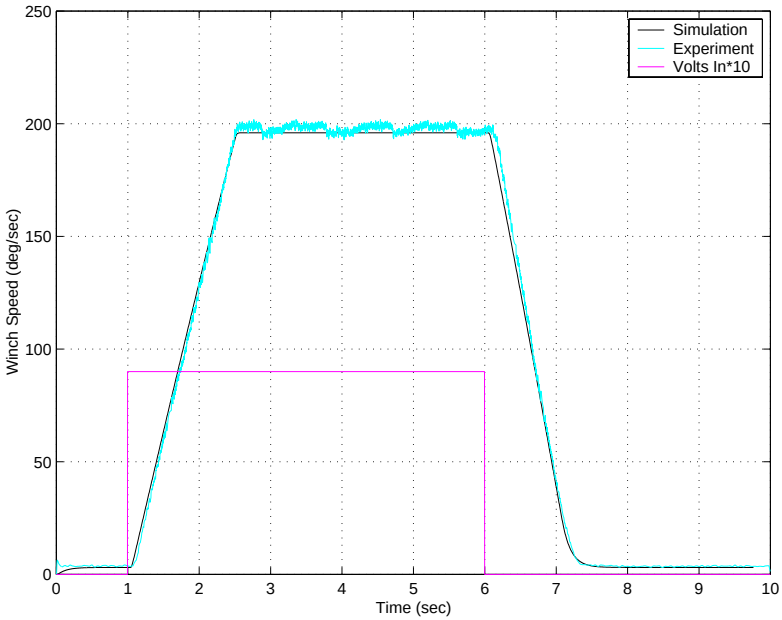

FIGURE 7.8 Hoist Winch Speed Data, 9V Step (hoistr10.dat)

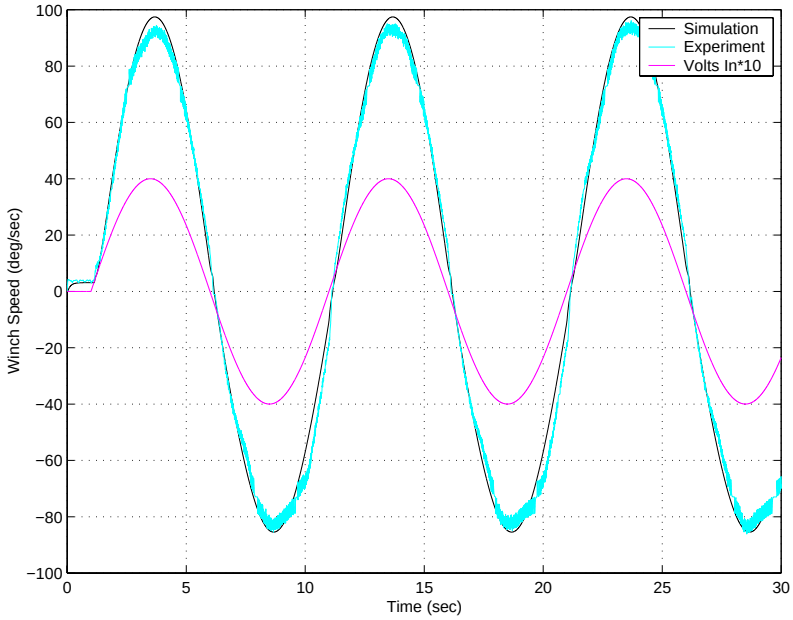

FIGURE 7.9 Hoist Winch Speed Data, 4V, 0.1Hz Sine (hoistr13.dat)

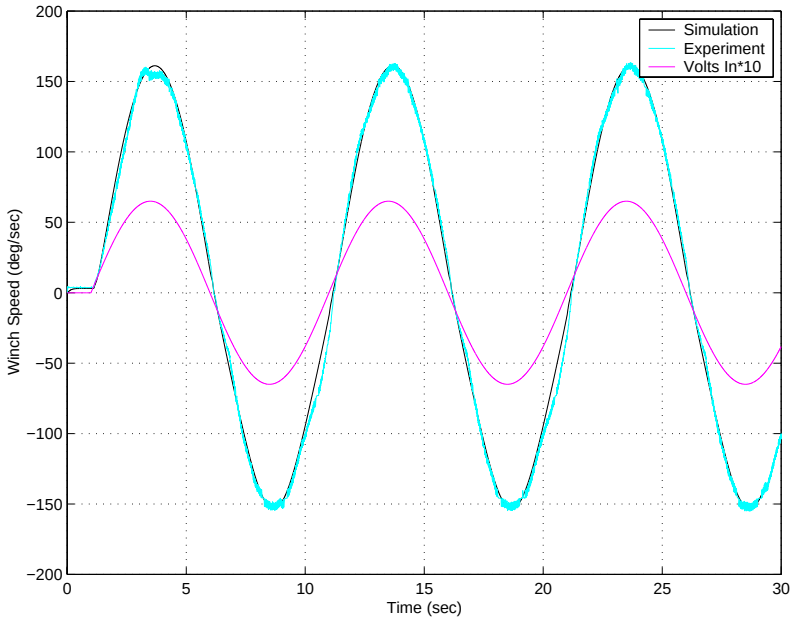

FIGURE 7.10 Hoist Winch Speed Data, 6.5V, 0.1Hz Sine (hoistr20.dat)

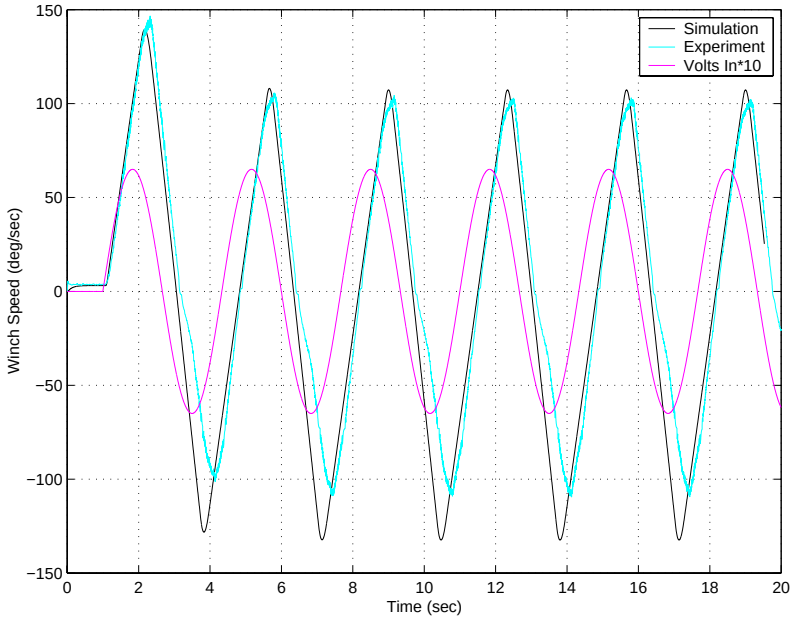

FIGURE 7.11 Hoist Winch Speed Data, 6.5V, 0.3Hz Sine (hoistr22.dat)

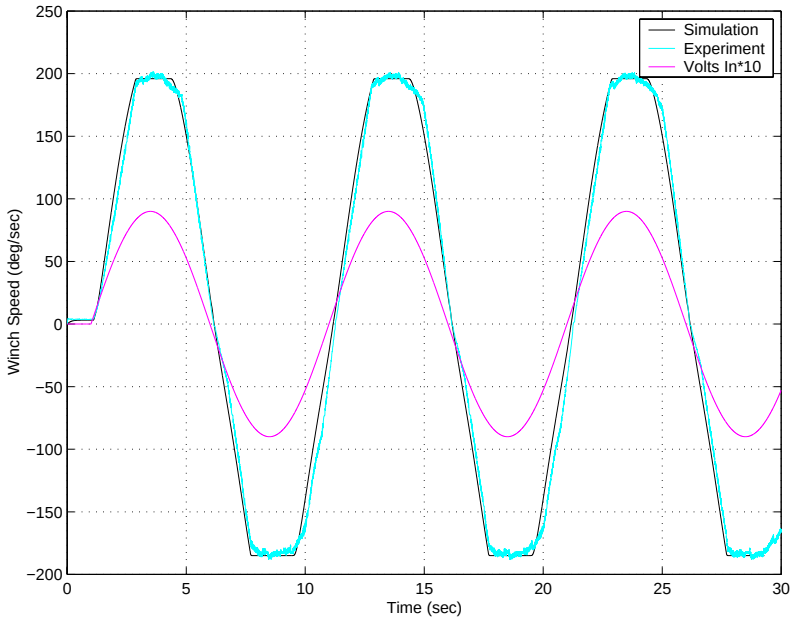

FIGURE 7.12 Hoist Winch Speed Data, 9V, 0.1Hz Sine (hoistr27.dat)

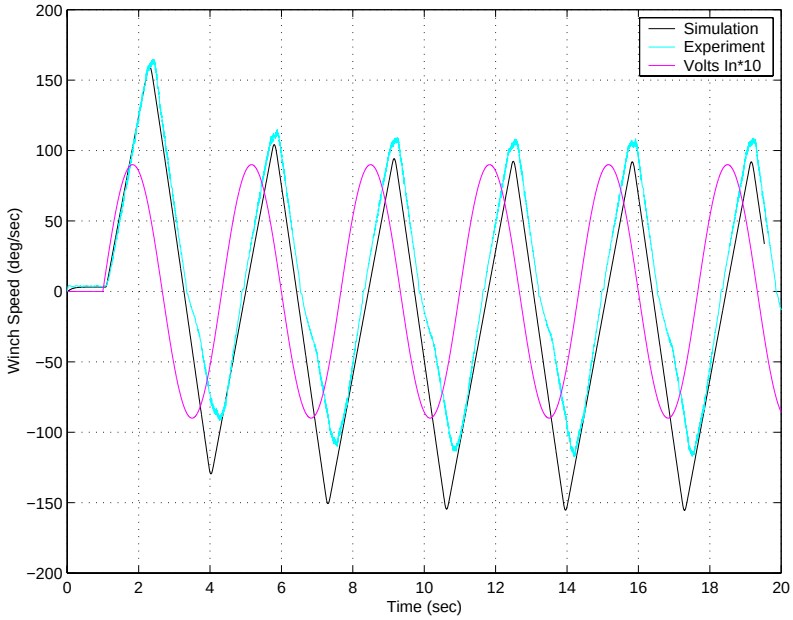

FIGURE 7.13 Hoist Winch Speed Data, 9V, 0.3Hz Sine (hoistr29.dat)

Figure 7.14 illustrates the characteristics of the internal states throughout the 9V, 0.1Hz sine wave of Figure 7.12.

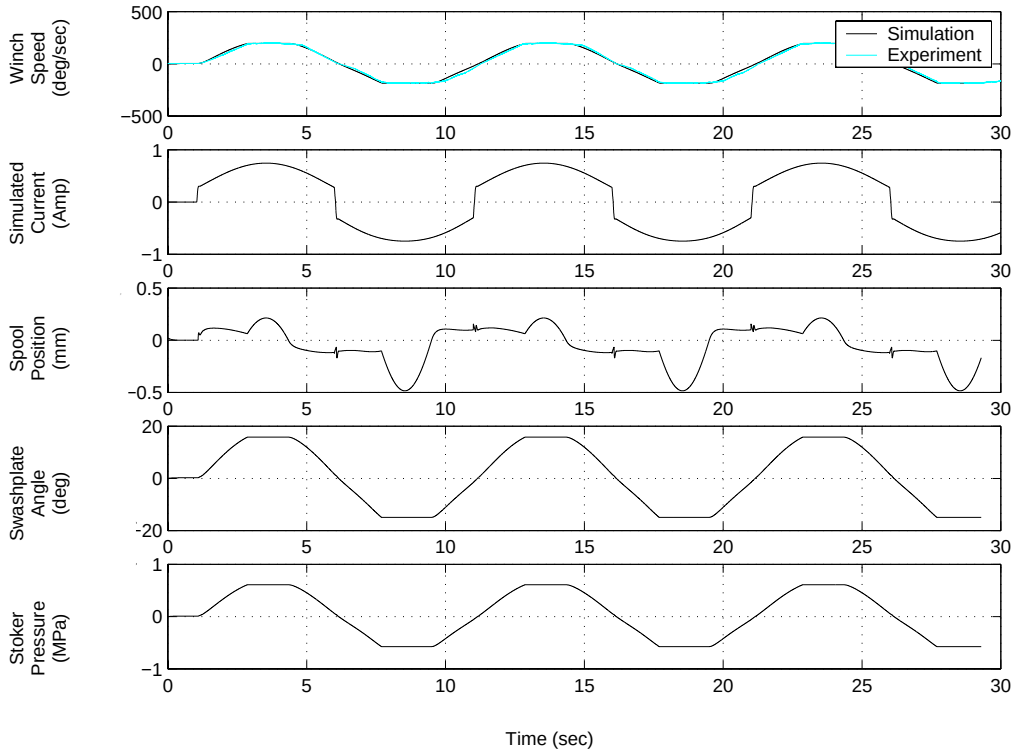

FIGURE 7.14 Internal States During 9V, 0.1Hz Sine (hoistr27.dat)

7.4 Slew Results

For all of data sets used in the optimization and shown in the figures below, the boom was raised to 53.5 degrees, however other tests run at 38 degrees show no apparent differences. The hook block was approximately 30 feet off the deck. All maneuvers end with a 4.5 second cubic spline to return the motor speed to zero if this is not already the case. A positive voltage indicates slewing to the right as seen from the cranes cab.

Table 7.3 Optimized Parameters, Slew Axis

Parameter	Optimized Value	Parameter	Optimized Value	Parameter	Optimized Value
Z_1	8.0275e-07	Z_8	6.3855e+05	Z_{15}	4.5768e+06
Z_2	0	Z_9	2.8887e+07	Z_{16} +/-	1.9799e-04/ -2.0014e-4
Z_3	0.23971	Z_{10}	0.29301	Z_{17}	0.16216
Z_4	0.019662	Z_{11}	0	Z_{18}	0
Z_5	0.011557	Z_{12}	0	Z_{19} +/-	3.5808/ 3.6609
Z_6	0.11475	Z_{13} +/-	0.25268/ -0.25564	Z_{20} +/-	0/0
Z_7	0	Z_{14}	0.0036461	Z_{21} +/-	1.6932/ 1.6940
				Z_{22} +/-	0/0

The following figures show simulated turret speeds compared with measured turret speeds. The voltage input signal is also included for reference.

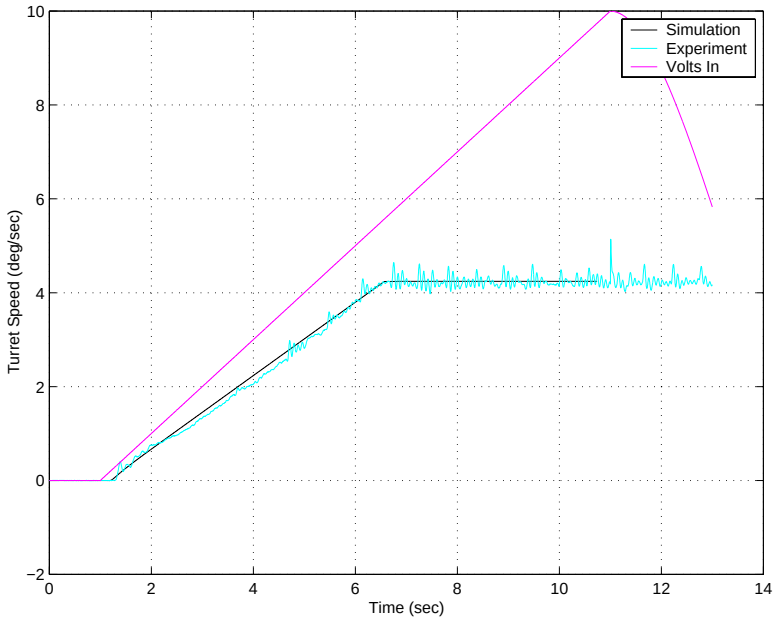

FIGURE 7.15 Slew Turret Speed Data, 1V/sec Ramp (slewr2.dat)

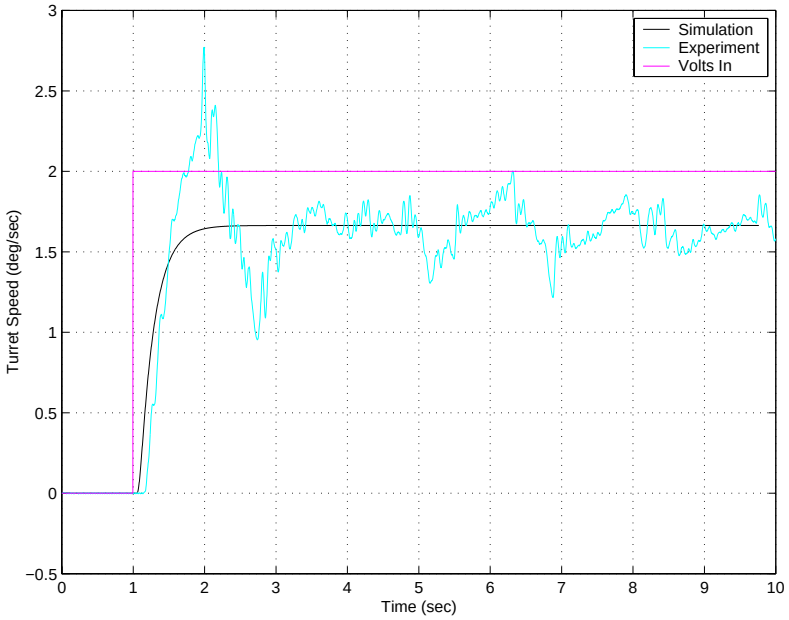

FIGURE 7.16 Slew Turret Speed Data, 2V Step (slewr10.dat)

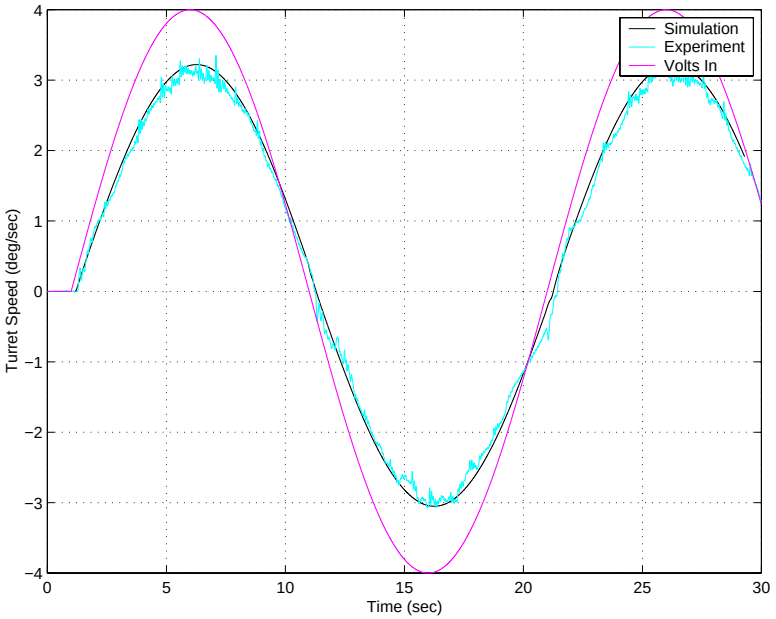

FIGURE 7.17 Slew Turret Speed Data, 4V, 0.05Hz Sine (slewr4.dat)

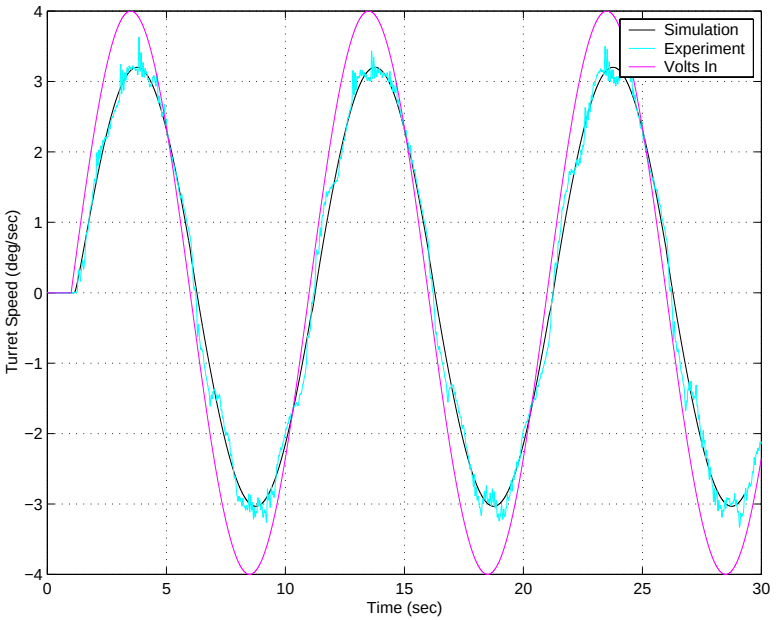

FIGURE 7.18 Slew Turret Speed Data, 4V, 0.1Hz Sine (slewr5.dat)

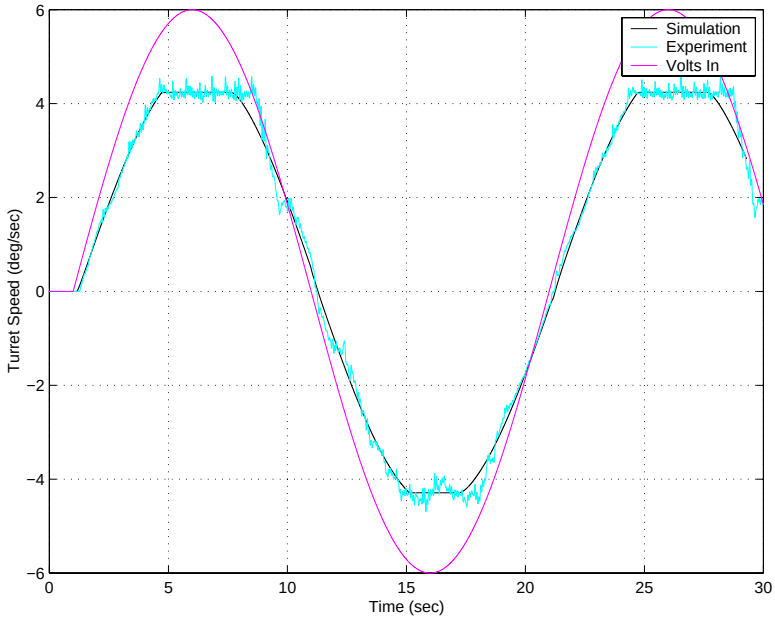

FIGURE 7.19 Slew Turret Speed Data, 6.0V, 0.05Hz Sine (slewr8.dat)

Figure 7.20 illustrates the characteristics of the internal states throughout the 6.0V, 0.05Hz sine wave of Figure 7.19.

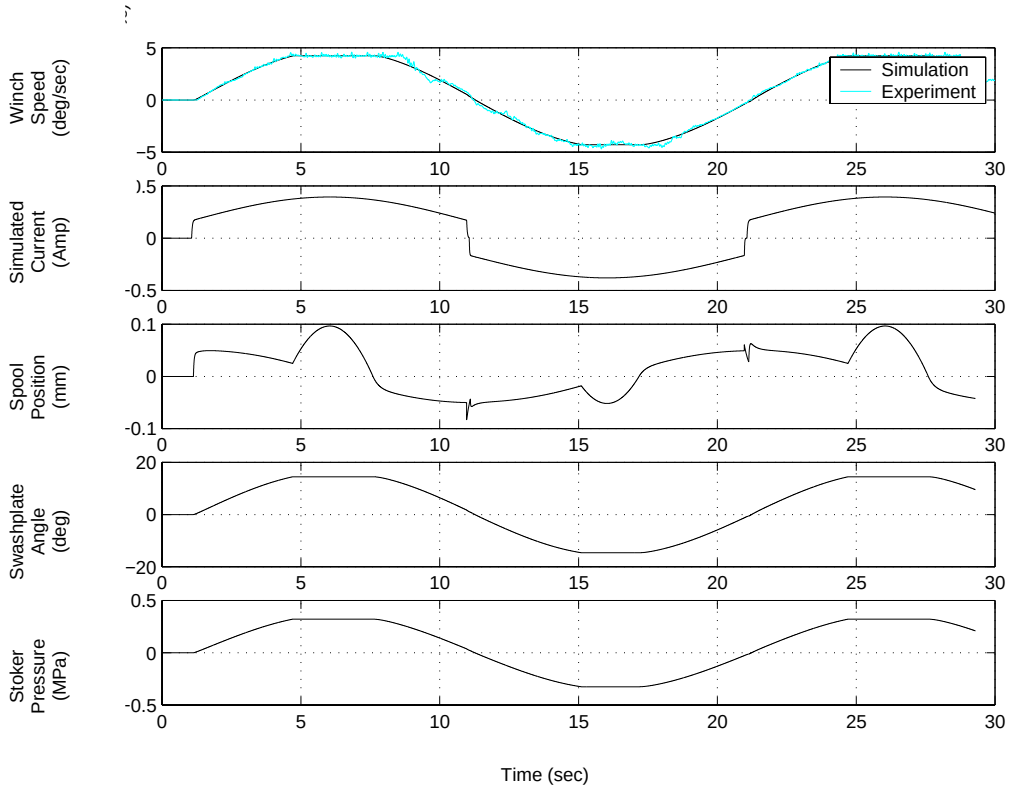

FIGURE 7.20 Internal States During 6.0V, 0.05Hz Sine (slewr8.dat)

7.5 Luff Results

The luff axis is parameterized differently from hoist and slew. Time constraints restricted the range of tests performed aboard the Flickertail State in June of 2000. Data was not taken using the Rexroth control card for this axis, only with the installed Hagglunds control card. It should also be noted that the current data taken using the Hagglunds control card during original tests of all three axes was corrupted by a malfunctioning A/D port. Tests were later run with the crane off to isolate the Hagglunds control card performance.

Parameterization of the luff axis while using the Rexroth control card required some extrapolation.

To create a complete set of data for optimization of the pump/motor model the Hagglunds card-only data was combined with the Hagglunds winch speed data. The Hagglunds current was used to drive the pump/motor simulation which was optimized against the Hagglunds winch speed data. This created all of the parameters for the pump/motor portion of the model as seen in Table 7.4 except for one. Because the Hagglunds winch speed data never reached a limit, it was not possible to ascertain the speed saturation level, parameter Z_{10}. The speed saturation limit was therefore placed, conservatively, just above the highest speed observed.

Assuming that the pump and motor work identically regardless of the type of control card creating the driving current, the modularity of the simulation can be exploited. By combining the model of the pump/motor dynamics with the Rexroth control card previously parameterized for the slew axis, an estimation of the luff axis performance with the Rexroth control card is obtained. Note that the slew and luff axes use the same control card.

It should be noted that during the time between when the winch rates were recorded and when the card-only data was recorded, the cards tuning potentiometers were adjusted. This means that the hybrid data sets used for parameter optimization are not necessarily consistent between current and winch speed. This was evident when hoist data was examined using a similar hybrid data set. Based on the hoist observation, the luff current was

increased by a factor of 1.5 for the optimization study below. The luff results, and the optimized parameters should be interpreted as approximate at best.

Figure 7.21 through 7.27 show the match between the pump/motor model and the winch speeds measured with the Hagglunds card. Figure 7.28 and 7.29 show the simulated winch speeds which resulted from the combination of the two individually optimized portions of the model. These plots show only the simulated winch speed as there is no winch speed measured with the Rexroth control card with which to compare them.

Note that in all figures below, a positive voltage indicates the boom moving upward. Maneuvers which do not naturally end at zero are brought to zero with a 2.5 second cubic spline.

Table 7.4 Optimized Parameters, Luff Axis

Parameter	Optimized Value	Parameter	Optimized Value	Parameter	Optimized Value
Z_1	2.9459e-07	Z_8	1.791e+06	Z_{15}	4.5768e+08
Z_2	0	Z_9	8.9295e+06	Z_{16} +/-	1.7482e-04/ -4.3292e-04
Z_3	4.6186	Z_{10}	45.742	Z_{17}	0.070183
Z_4	0.25419	Z_{11}	0	Z_{18}	0
Z_5	0.013711	Z_{12}	0	Z_{19} +/-	20.272/ 21.009
Z_6	13.228	Z_{13} +/-	0.059489/ -0.07282	Z_{20} +/-	0/0
Z_7	0	Z_{14}	0.0036461	Z_{21} +/-	0/0
				Z_{22} +/-	0/0

The following figures show simulated winch speeds, as driven by a Hagglunds control card, compared with winch speeds measured with the Hagglunds card in place. The voltage input signal is also included for reference.

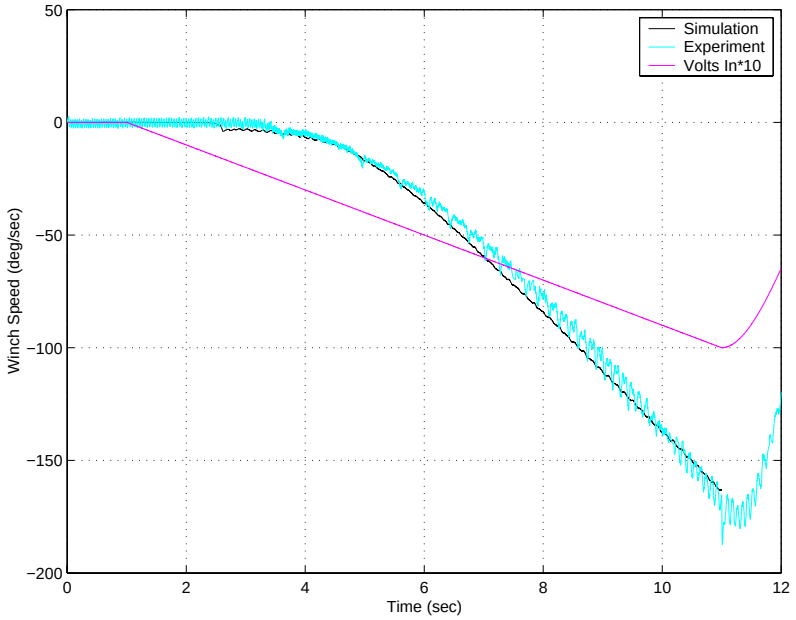

FIGURE 7.21 Luff Winch Speed Data, 1V/sec Ramp (luffh2.dat and luff9.dat)

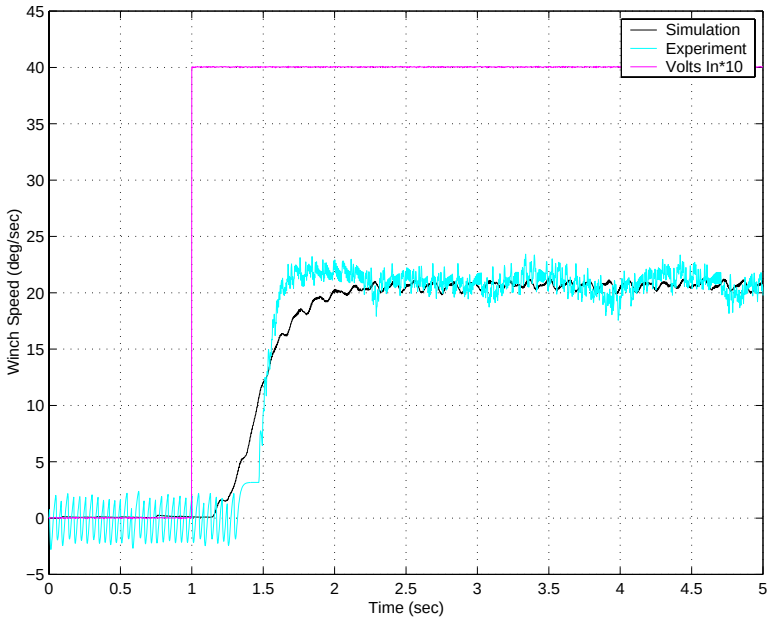

FIGURE 7.22 Luff Winch Speed Data, 4V Step (luffh3.dat and luff12.dat)

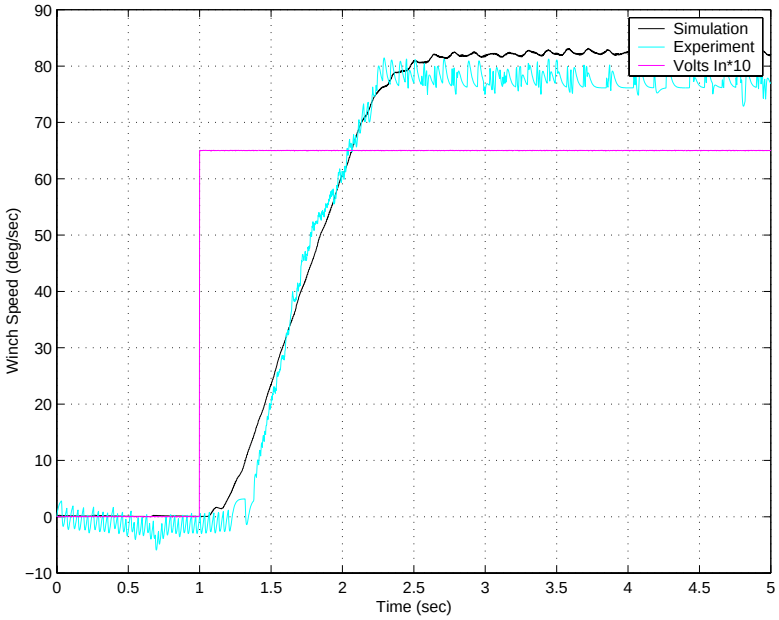

FIGURE 7.23 Luff Winch Speed Data, 6.5V Step (luffh5.dat and luff14.dat)

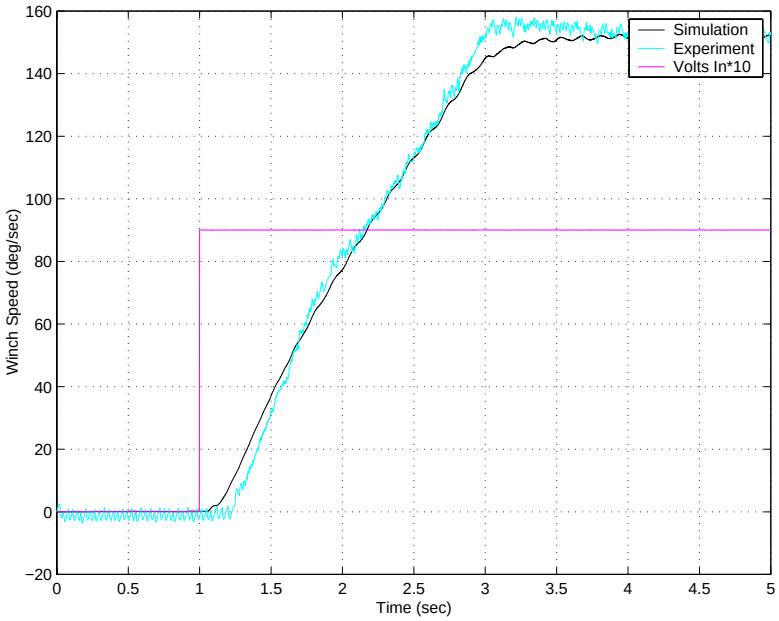

FIGURE 7.24 Luff Winch Speed Data, 9V Step (luffh7.dat and luff16.dat)

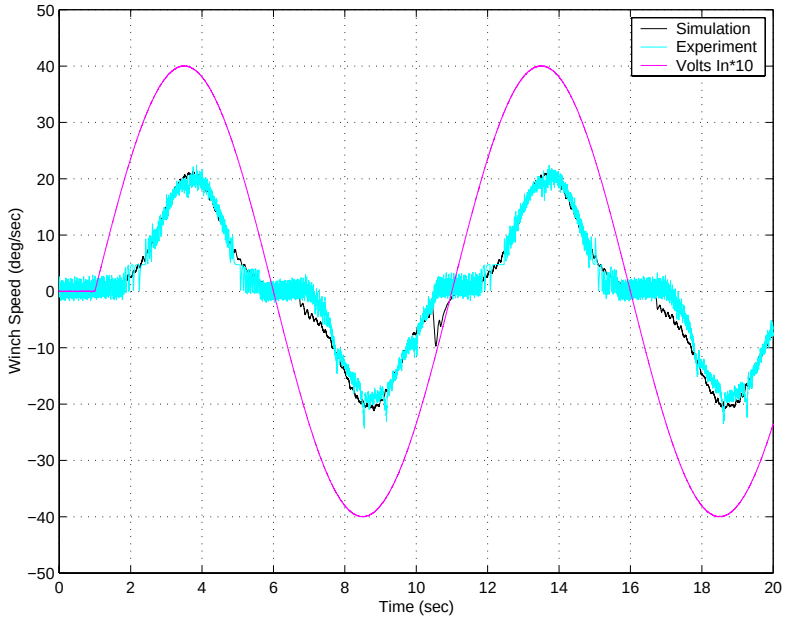

FIGURE 7.25 Luff Winch Speed Data, 4V, 0.1Hz Sine (luffh9.dat and luff19.dat)

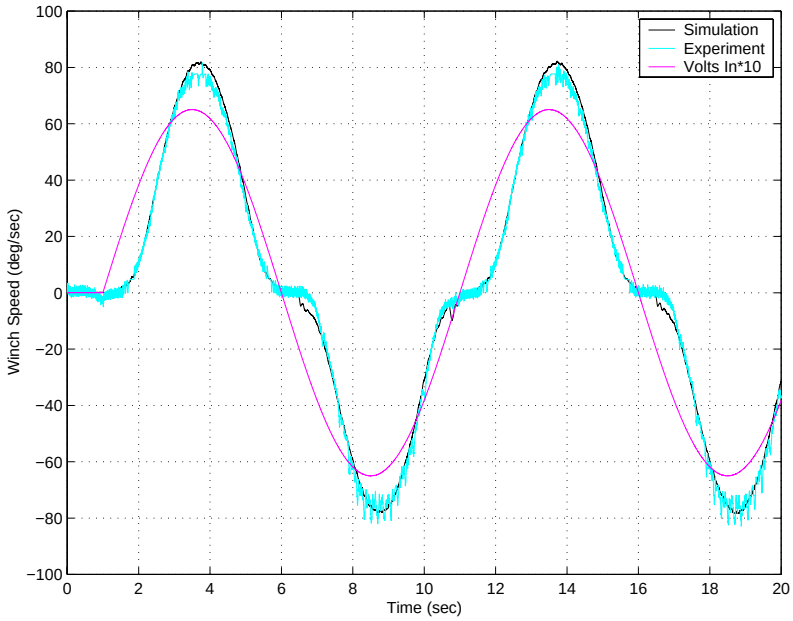

FIGURE 7.26 Luff Winch Speed Data, 6.5V, 0.1Hz Sine (luffh17.dat and luff26.dat)

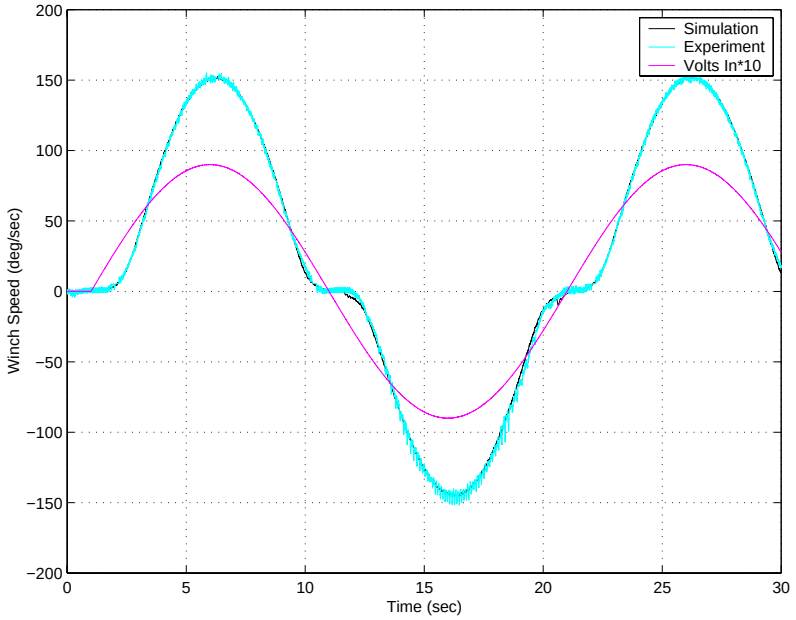

FIGURE 7.27 Luff Winch Speed Data, 9V, 0.05Hz Sine (luffh22.dat and luff31.dat)

Figure 7.28 and 7.29 show the extrapolated winch speed simulation for the luff axis using the Rexroth control card. Input voltage first leaves zero at 1 second.

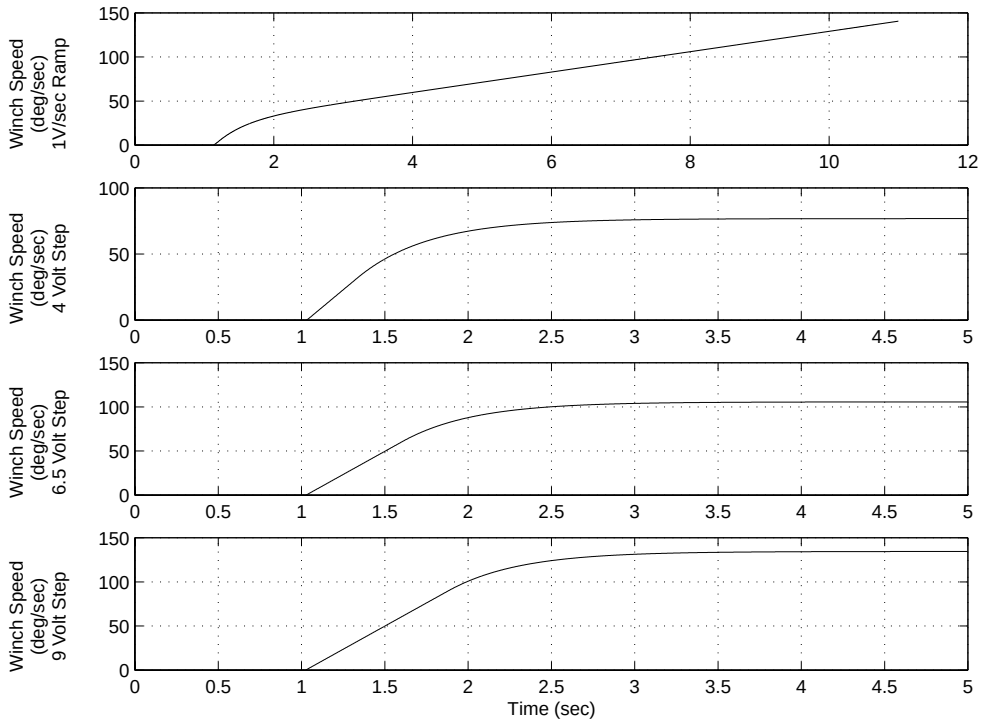

FIGURE 7.28 Rexroth Luff Step and Ramp Data

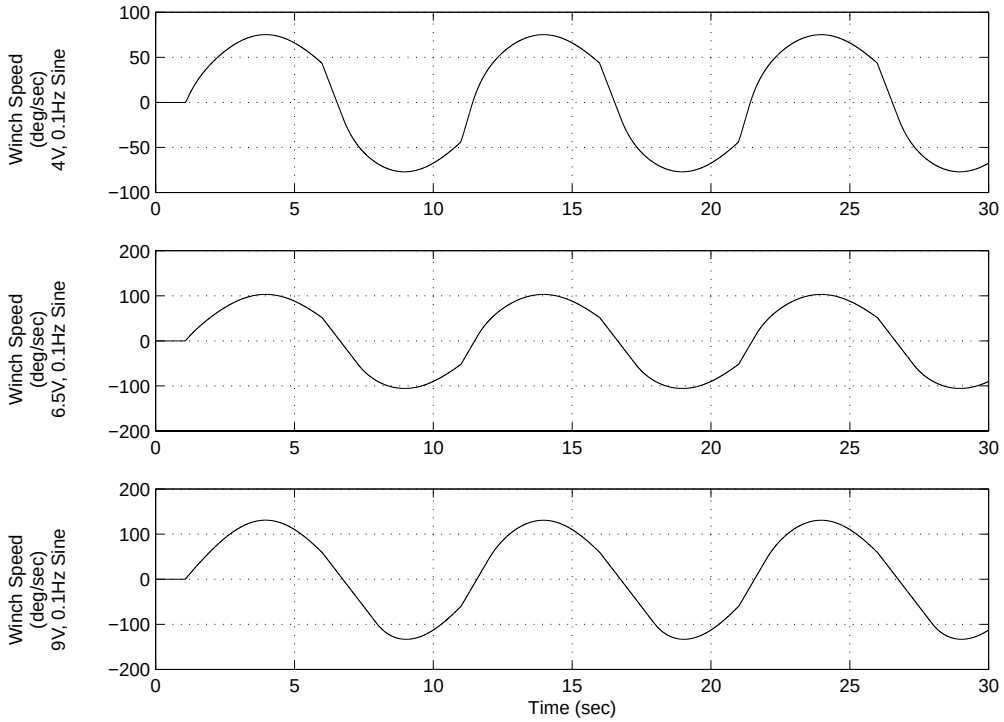

FIGURE 7.29 Rexroth Luff Sine Data

7.6 Error Quantification, Control Card Model

Tabulated in this section is the maximum percent error experienced for the complete model in each type of data set. The error is tabulated by axis, with the luff axis omitted due to the lack of a comparative signal.

Particular phenomenon were not included in the selection of the region with the maximum percent error due to either an acceptable explanation of the error and the inability to model it, or its irrelevance to the overall goals of the project. In general, sine data is considered to be the test maneuver most relevant to actual crane operation. Trade-offs in model performance, when they existed between test types, where chosen to the benefit of sine maneuvers. The overshoot seen in the hoist step data was not included in the modeling effort and thus is not considered in the error calculation. This overshoot could be due to nonlinearities in solenoid performance. Specifically, when the solenoid is stopped, a

higher force is required to break it free again. This observed behavior is similar to the differences between static and kinetic friction, however it is most likely due to the electromagnetic force nonlinearities instead of the contact forces between the plunger and the solenoid housing. Likewise, the small transient seen at the start of the signal is also dismissed from the error calculations. It is the effect of the plussing, which is implemented as a step immediately after the simulation is started. Work could be done with the initial conditions to remove this effect

Table 7.5 Maximum Percent Error in Winch Speed, Hoist Axis

Test Type	Maximum % Error	Speed/ Amp/ Freq	Associated Figure
Ramp	11.01%	1 V/sec	Figure 7.5
Step	2.42%	4 V	Figure 7.6
Sine	7.57%	4 V, 0.1 Hz	Figure 7.9

Table 7.6 Maximum Percent Error in Winch Speed, Slew Axis

Test Type	Maximum % Error	Speed/ Amp/ Freq	Associated Figure
Ramp	7.52%	1 V/sec	Figure 7.15
Step	-	-	-
Sine	18.2%	6 V, 0.05 Hz	Figure 7.19

The maximum percent area for step tests on the slew axis is not calculated due to the excitation caused by these maneuvers. The oscillatory phenomenon seen in Figure 7.16, pos-

sibly due to backlash in the slewing gears, make calculation of the steady state value difficult to predict with accuracy.

Also not included in the selection of the test and region with maximum percent error were the sine wave tests at 0.3 Hz. Although sine data is the most critical test type, as it is most similar to the types of inputs commanded by crane operators. The dominant roll frequency of the ship is approximately 0.1 Hz, and a pure 0.3 Hz frequency will probably never be demanded of the crane by the controller. The sea states of interest for controller design induce ship oscillations with a minimum period of 10 to 12 seconds. At 0.3 Hz the drive system model exhibits that there are some dynamic phenomenon which are not being captured, as illustrated by the phase shift in Figure 7.11, for instance. This is also seen in the hoist ramp data, Figure 7.5. The model is able to accurately capture the steady state gain in the step responses for hoist even when the ramp data would appear to predict an error at that voltage input. For instance, the model errs on the high side at 6.5 volts in the hoist ramp data (Figure 7.5), while the simulated 6.5 volt step signal (Figure 7.7) is low at steady state in the step test.

A possible explanation of this phenomenon is illustrated by examining the steady state error of a first order system. It should be noted that while a first order system seems overly simple, it was shown in Section 6.2.6 that when small motions are assumed, the voltage to motor speed model is first order. Writing the transfer function of the *actual* system in the example as

$$\frac{\Omega_a}{V} = \frac{k}{s + a} \tag{7.8}$$

and that of the *simulation* as

$$\frac{\Omega_s}{V} = \frac{k_s}{s + a_s} \tag{7.9}$$

the error between *simulated* and *actual* performance, $E = \Omega_a - \Omega_s$, is written as

$$E = \left[\frac{ks + ka_s - k_s s - ak_s}{(s + a)(s + a_s)}\right]V \tag{7.10}$$

If the input is a step, $V = \frac{1}{s}$, then the steady state error, defined as $e_{ss} = \lim_{s \to 0}(sE)$, can be written,

$$e_{ss} = \lim_{s \to 0} \frac{ks + ka_s - k_s s - k_s a}{(s + a)(s + a_s)} = \frac{ka_s - k_s a}{a - a_s} \tag{7.11}$$

This indicates that the steady state error is a constant for step inputs. Performing similar calculations for a ramp input, $V = \frac{1}{s^2}$, gives

$$e_{ss} = \lim_{s \to 0}\left(\frac{ks + ka_s - k_s s - k_s a}{(s + a)(s + a_s)}\right)\frac{1}{s} \tag{7.12}$$

If the steady state error is zero, however, for the step, then from Equation 7.11 $ka_s = ak_s$, and Equation 7.12 becomes

$$e_{ss} = \lim_{s \to 0}\left(\frac{ks - k_s s}{(s + a)(s + a_s)}\right)\frac{1}{s} \tag{7.13}$$

The cancellation of s gives a constant steady state error for ramp input, as described by

$$e_{ss} = \frac{k - k_s}{aa_s} \tag{7.14}$$

In summary, Equation 7.14 indicates a constant steady state error for a ramp input when that of the step input is zero, which is similar to what is experienced in the simulated data.

Also note that the relatively high maximum percent error in the slew axis sine tests is the result of the early end to the saturated region at the peaks. Considering the amplitude and the phase only, the maximum percent error across all of the sine tests would be 2.90%, and for that particular test would be 0.04%.

8 Model Summary

Presented in this section is a summary of the total drive system model. It is included as a review and consolidation of the developments contained in the bulk of the report. In this section the developmental equations are removed for clarity and those presented here are only those specifically needed for implementation. It is not the goal of this section to convey a complete understanding of the system. For this, the reader is directed to the referenced sections which fully detail the development of the model.

The block diagram of the control card is repeated here for ease of reference and is followed by the equations defining each block. The final three equations of motion describing the pump/motor system are also included. All equations in this section are exact replicas of equations found elsewhere in this document, and in all cases the original equation number is included to facilitate cross-referencing. The full tables of optimized parameters from voltage input to motor speed output for all three axes are also included.

8.1 Control Card Summary

The block diagram of Figure 8.1, illustrates both channels of the control card while the equations define each block along the positive channel only.

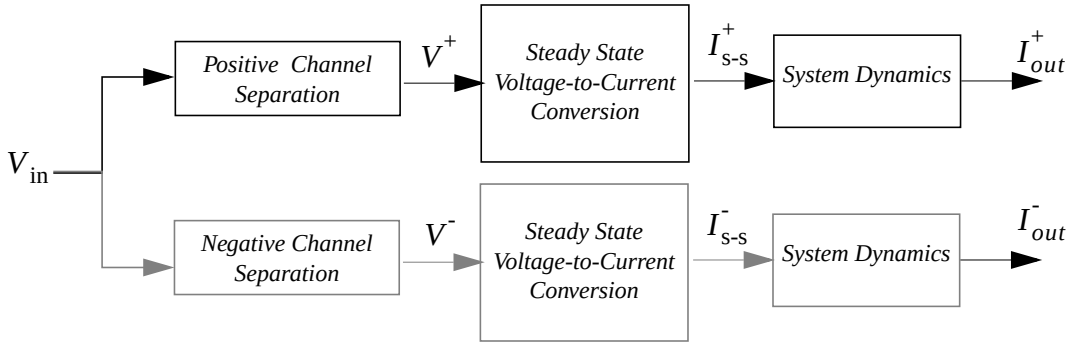

FIGURE 8.1 Overview of the Control Card

where the "Positive Channel Separation" block is handled by Equation 8.1 (also Section 3.1).

$$V = \begin{cases} V_{in} & V_{in} \geq 0 \\ 0 & V_{in} < 0 \end{cases} \tag{8.1}$$

the "Steady State Voltage-to-Current Conversion" block, is described by Equation 8.2 (also Equation 3.3),

$$I_{s\text{-}s} = \begin{cases} 0 & V < V_{dz} \\ J + G_1(V - V_{dz}) + G_2(V - V_{dz})^2 + G_3(V - V_{dz})^3 & V \geq V_{dz} \end{cases} \tag{8.2}$$

and the expanded "System Dynamics" block is represented by Equation 8.3 through 8.6.

The time delay is described by Equation 8.3 (also Equation 3.4)

$$I_{\text{s-s,td}} = \begin{cases} 0 & t \leq \tau_{td} \\ I_{\text{s-s}}(t - \tau_{td}) & t > \tau_{td} \end{cases} \tag{8.3}$$

The second order filter and coloumb-like term are defined by Equation 8.4 (also Equation 3.6)

$$I^*_{s\text{-}s,\,n+1} = 2(1 - \zeta\omega_n h)I^*_{s\text{-}s,\,n} + (2\zeta\omega_n h - h^2\omega_n^2 - 1)I^*_{s\text{-}s,\,n-1} + \\ -Ch^2\,\text{sign}\left(\frac{1}{h}[I^*_{s\text{-}s,\,n} - I^*_{s\text{-}s,\,n-1}]\right) + (h^2\omega_n^2)I_{\text{s-s,td,n}} \tag{8.4}$$

Rate limits for increasing (positive) voltage inputs are written as: (also Equation 3.8)

$$\dot{I}_{out} = \begin{cases} \dot{I}_{lim1} & I^*_{s\text{-}s} > I_{t,\,1} \text{ and } \dot{I}^*_{s\text{-}s} \geq \dot{I}_{lim1} \\ \dot{I}^*_{s\text{-}s} & I^*_{s\text{-}s} > I_{t,\,1} \text{ and } \dot{I}^*_{s\text{-}s} \leq \dot{I}_{lim1} \\ \dot{I}^*_{s\text{-}s} & I^*_{s\text{-}s} \leq I_{t,\,1} \end{cases} \tag{8.5}$$

and for decreasing (positive) voltage inputs the rate limits and exponential response are described by: (also Equation 3.9)

$$\dot{I}_{out} = \begin{cases} \dot{I}_{lim2} & I^*_{s\text{-}s} > I_{t,\,2} \text{ and } -\dot{I}^*_{s\text{-}s} \geq \dot{I}_{lim2} \\ \dot{I}^*_{s\text{-}s} & I^*_{s\text{-}s} > I_{t,\,2} \text{ and } -\dot{I}^*_{s\text{-}s} \leq \dot{I}_{lim2} \\ \dfrac{d}{dt}(I^*_{s\text{-}s,\,i} e^{-a(t-\tau_i)}) & I^*_{s\text{-}s} \leq I_{t,\,2} \end{cases} \tag{8.6}$$

The parameters which define the card model for both channels are tabulated by axis in Section 8.3. Further discussion of the control card functions can be found in Section 3.

8.2 Pump and Motor Dynamic Equations

The final three dynamic equations of motions used to model the pump, control module and hydraulic motor take the current from the control card as input and output the motor speed. This model is general enough to capture all three of the crane's axes and reflects the force equilibrium simplifications described in Section 6.2.4.

The force is first computed from the input using (also Equation 6.61)

$$F_{sol} = \begin{cases} 0 & i \leq Z_{17} \\ Z_{22}(i - Z_{17})^4 + Z_{21}(i - Z_{17})^3 + Z_{20}(i - Z_{17})^2 + Z_{19}(i - Z_{17}) & i > Z_{17} \end{cases} \tag{8.7}$$

The equation of motion for the swashplate (also Equation 6.55) is

$$\alpha_{sw} = \tan^{-1}(Z_1 \delta P - Z_2) \tag{8.8}$$

which is then limited as described in Section 6.2.5.

$$\alpha_{sw} = \begin{cases} \alpha_{sw,\,max} & \alpha_{sw} \geq \alpha_{sw,\,max} \\ \alpha_{sw} & \alpha_{sw,\,min} < \alpha_{sw} < \alpha_{sw,\,max} \\ \alpha_{sw,\,min} & \alpha_{sw} \leq \alpha_{sw,\,min} \end{cases} \tag{8.9}$$

The spool valve equation (also Equation 6.57) is

$$x = \frac{-Z_5 Z_4 \sin\alpha\cos\alpha + Z_5^2 F_{sol}\sqrt{Z_4 - Z_5^2 F_{sol}^2}}{Z_4\cos^2\alpha - Z_5^2 F_{sol}^2} + Z_{18} \tag{8.10}$$

which is then limited as described in Section 6.2.5,

$$x_{eff} = \begin{cases} x_{max} & x \geq \alpha_{sw,\,max} \\ x & \alpha_{sw,\,min} < \alpha_{sw} < \alpha_{sw,\,max} \\ x_{min} & \alpha_{sw} \leq \alpha_{sw,\,min} \end{cases} \tag{8.11}$$

The equation describing pressure across the stroker (also Equation 6.67) is

$$\delta\dot{P} = \frac{Z_8}{1 + Z_6}\left(\sqrt{\frac{1}{2}(Z_9 - P)}\right)x_{eff} \tag{8.12}$$

which must be integrated using a suitable numerical integration method.

Finally, the motor speed equation is evaluated as (also Equation 6.69)

$$\Omega = Z_{10}\alpha_{sw} - Z_{11} - Z_{12}\cos(\alpha_{sw})\delta\dot{P} \tag{8.13}$$

The parameters which define the model for each individual axis are tabulated in Section 8.3. Further discussion the pump/motor model can be found in Section 6.

8.3 Optimized Parameterization

The final parameters, chosen through optimization, for the control card and pump/motor models are contained in Tables 8.1, 8.2, and 8.3 for the hoist, slew, and luff axes respectively.

Table 8.1 Optimized Parameters, Hoist Axis

Parameter	Optimized Value	Parameter	Optimized Value	Parameter	Optimized Value
Control Card Parameters					
V_{dz} +/- (V)	0.12663/ -0.097857	G_1 +/- (A/V)	0.049315/ 0.054706	$I_{t,1}$ +/- (A)	0.33628/ -0.34863
J +/- (A)	0.27473/ -0.2951	G_2 +/- (A/V)	0.00081975/ 0.00042879	$I_{t,2}$ +/- (A)	0.13724/ -0.10683
ζ +/- (n.d.)	0.71887/ 0.72563	G_3 +/- (A/V)	-4.5317e-05/ -9.7369e-05	$\dot{I}_{lim,1}$ +/- (A/sec)	2.2841/ -2.2733
ω_n +/- (rad/sec)	56.253/ 61.833	τ_{td} +/- (sec)	0.014	$\dot{I}_{lim,2}$ +/- (A/sec)	-2.2615/ 2.6483
C +/- (A)	0	a +/- ($\sec^{-1}$)	19.131/ 21.379		
Pump/Motor Parameters					
Z_1	4.6441e-07	Z_8	1.6514e+06	Z_{15}	4.1191e+06
Z_2	0	Z_9	1.0743e+07	Z_{16} +/-	1.1634e-04/ -1.5126e-04
Z_3	0.62542	Z_{10}	12.396	Z_{17}	0.27867
Z_4	0.053604	Z_{11}	0	Z_{18}	2.1667e-05
Z_5	0.0079568	Z_{12}	0	Z_{19} +/-	12.349/ 10.019
Z_6	0.073926	Z_{13} +/-	0.27596/ -0.26047	Z_{20} +/-	0.27028/ 3.7297
Z_7	0	Z_{14}	0.0036461	Z_{21} +/-	0.22421/ 1.6581
				Z_{22} +/-	0/ -54.153

Table 8.2 Optimized Parameters, Slew Axis

Parameter	Optimized Value	Parameter	Optimized Value	Parameter	Optimized Value
Control Card Parameters					
V_{dz} +/- (V)	0.090948/ -0.088321	G_1 +/- (A/V)	0.037892/ 0.036385	$I_{t,\,1}$ +/- (A)	0.33628/ -0.34863
J +/- (A)	0.17033/ -0.16474	G_2 +/- (A/V)	0	$I_{t,\,2}$ +/- (A)	0.12352/ -0.10683
ζ +/- (n.d.)	1.4157/ 1.1014	G_3 +/- (A/V)	0	$\dot{I}_{lim,\,1}$ +/- (A/sec)	2.2148/ -2.2733
ω_n +/- (rad/sec)	98.377/ 135.39	τ_{td} +/- (sec)	0.014	$\dot{I}_{lim,\,2}$ +/- (A/sec)	-2.2615 2.6483
C +/- (A)	4.8368/ 4.8368	a +/- ($\sec^{-1}$)	19.131/ 21.379		
Pump/Motor Parameters					
Z_1	8.0275e-07	Z_8	6.3855e+05	Z_{15}	4.5768e+06
Z_2	0	Z_9	2.8887e+07	Z_{16} +/-	1.9799e-04/ -2.0014e-4
Z_3	0.23971	Z_{10}	0.29301	Z_{17}	0.16216
Z_4	0.019662	Z_{11}	0	Z_{18}	0
Z_5	0.011557	Z_{12}	0	Z_{19} +/-	3.5808/ 3.6609
Z_6	0.11475	Z_{13} +/-	0.25268/ -0.25564	Z_{20} +/-	0/0
Z_7	0	Z_{14}	0.0036461	Z_{21} +/-	1.6932/ 1.6940
				Z_{22} +/-	0/0

Table 8.3 Optimized Parameters, Luff Axis

Parameter	Optimized Value	Parameter	Optimized Value	Parameter	Optimized Value
Control Card Parameters					
V_{dz} +/- (V)	0.090948/ -0.088321	G_1 +/- (A/V)	0.037892/ 0.036385	$I_{t,1}$ +/- (A)	0.33628/ -0.34863
J +/- (A)	0.17033/ -0.16474	G_2 +/- (A/V)	0	$I_{t,2}$ +/- (A)	0.12352/ -0.10683
ζ +/- (n.d.)	1.4157/ 1.1014	G_3 +/- (A/V)	0	$\dot{I}_{lim,1}$ +/- (A/sec)	-2.2148/ 2.2733
ω_n +/- (rad/sec)	98.377/ 135.39	τ_{td} +/- (sec)	0.014	$\dot{I}_{lim,2}$ +/- (A/sec)	2.2615 -2.6483
C +/- (A)	4.8368/ 4.8368	a +/- ($\sec^{-1}$)	19.131/ 21.379		
Pump/Motor Parameters					
Z_1	2.9459e-07	Z_8	1.791e+06	Z_{15}	4.5768e+08
Z_2	0	Z_9	8.9295e+06	Z_{16} +/-	1.7482e-04/ -4.3292e-04
Z_3	4.6186	Z_{10}	45.742	Z_{17}	0.070183
Z_4	0.25419	Z_{11}	0	Z_{18}	0
Z_5	0.013711	Z_{12}	0	Z_{19} +/-	20.272/ 21.009
Z_6	13.228	Z_{13} +/-	0.059489/ -0.07282	Z_{20} +/-	0/0
Z_7	0	Z_{14}	0.0036461	Z_{21} +/-	0/0
				Z_{22} +/-	0/0

9 Conclusions

The advantage of this model is that it is sufficiently general to allow the performance of all three axes to be captured with one set of equations. The model's other strength, however, is that it is also very modular. For instance, the ability exists within the model to simulate the effect on acceleration of an increased orifice size (parameter Z_{16}). The system identification method presented in this document can also be utilized to extrapolate drive system performance if the control module is replaced or the pump size increased. The effects of increasing both the orifice size and the maximum flow rate by 50% can be seen in Figure 9.1. Note that both the maximum speed and acceleration limit are increased.

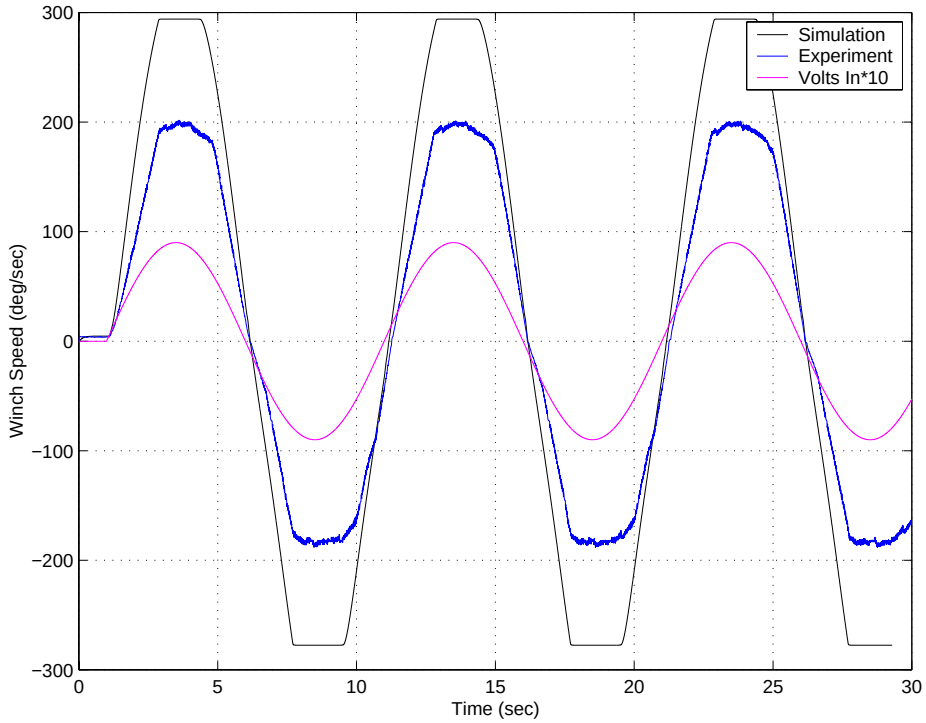

FIGURE 9.1 Extrapolated Hoist Winch Speed, 9V, 0.1Hz Sine (hoistr27.dat)

The ability to anticipate load dependent behavior could also be achieved by integrating this simulation with LoadSim, the dynamic load analysis tool.

If drive system tests are executed in the future, the following data would lead to a more accurate model:

- pressure drop across the hydraulic motor
- stroker pressure
- swash plate angle
- spool position
- independent solenoid current signals
- stair-step maneuvers

This information would allow the model's internal states to be better matched, resulting in better voltage to motor speed results. Specifically, the step response overshoot not predicted in the model could be determined. There also appear to be load dependencies on the rising side of hoist sine data that could be accurately captured with knowledge on the performance of the internal states, specifically, the pressure drop across the motor and its relationship to stroker pressure.

As mentioned in Section 1.2, there is a noticeable winch speed oscillation that is likely caused by the cam ring geometry of the motor. Figure 9.2 and 9.3 illustrate this effect. During these tests a hatch cover was being lifted, and the hoist winch speed recorded using the Hagglunds control card (the green traces). An empirical investigation of this phenomenon indicates that there is a 5.5 cycle per drum revolution effect. In any future servo design work the effects of this oscillation on high frequency crane modes, such as cable stretching boom bounce, should be considered.

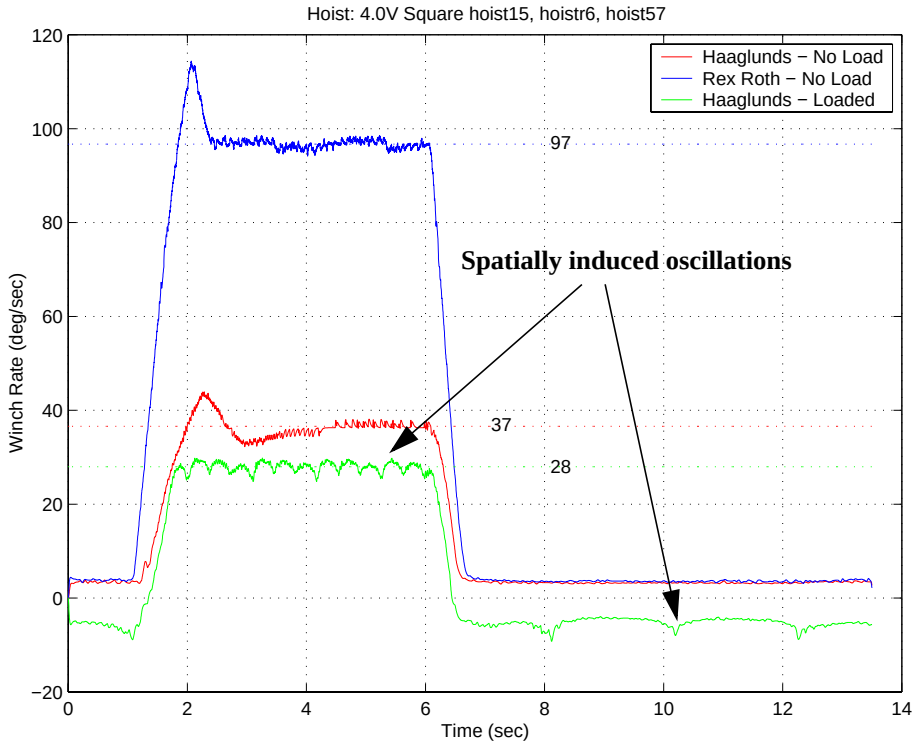

FIGURE 9.2 Illustration of Load Dependent Winch Speed Oscillation, Low Speed

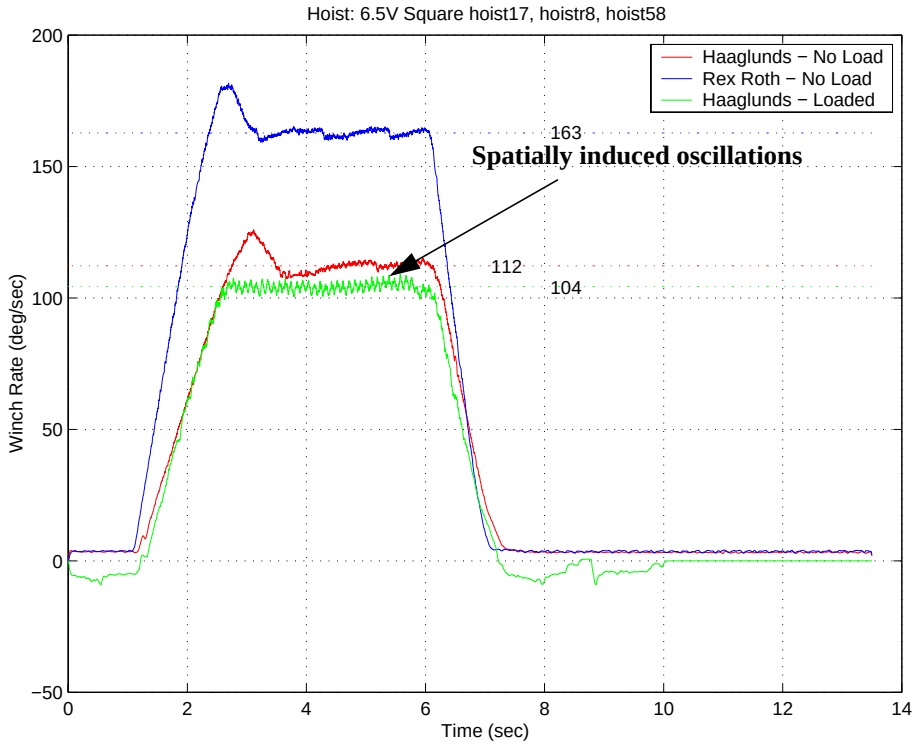

FIGURE 9.3 Illustration of Load Dependent Winch Speed Oscillation, Medium Speed

References

[1] H. Schaub, "Data Description of the Phase II Drive System Tests on TG3637 Shipboard Crane System," Sandia National Laboratories, Intelligent Systems and Robotics Center, November 15, 2001.

[2] Rexroth Worldwide Hydraulics, "Applications and Service Manual, AA4V Series 2 Hydrostatic Transmission Pump," Version RA 06715 / 08.88

[3] S. LeQuoc, Y. F. Xiong, and R. M. H. Cheng, "Identification and Control of Nonlinear Hydraulic System," SAE TEchnical Paper Series, No. 921622.

[4] Halme, Jarkko, "Utilization of Genetic Algorithm in Online Tuning of Fluid Power Servos," Ph.D. Dissertation, Lappeenranta University of Technology, 1997.

[5] J. L. Shearer, "Digital Simulation of a Coulomb-Damped Hydraulic Servosystem," Journal of Dynamic Systems, Measurement, and Control, Vol. 105, December, 1985.

[6] G. Vossoughi and M. Donath, "Dynamic Feedback Linearization for Electrohydraulically Actuated Control Systems," Transactions of the ASME, Vol. 117, December, 1995.

[7] J. E. Bobrow and K. Lum, "Adaptive, High Bandwidth Control of a Hydraulic Actuator," Transactions of the ASME, Vol. 118, December, 1996.

[8] H. E. Merritt, *Hydraulic Control Systems*, New York, Wiley, 1967.

[9] X. Zhang, J. Cho, S. S. Nair, and N. D. Manring, "Damping on the Swash Plate of an Axial-Piston Pump," Proceedings of the American Control Conference, Chicago, Illinois, June, 2000

[10] G. Zeiger and A. Akers, "Torque on the Swashplate of an Axial Piston Pump," Transactions of the ASME, Vol. 107, September, 1985.

[11] N. D. Manring, "The control and Containment Forces on the Swash Plate of an Axial-Piston Pump," Journal of Dynamic Systems, Measurement, and Control, Vol. 121, December, 1999.

[12] A. G. Alleyne, and R. Liu, "Systematic Control of a Class of Nonlinear Systems with Application to Electrohydraulic Cylinder Pressure Control," IEEE Transactions on Control Systems Technology, Vol. 8, No. 4, July 2000.

[13] B. Kwak, A. Yagle, and J. Levitt, "Nonlinear System Identification of Hydraulic Actuator Friction Dynamics using a Hammerstein Model," IEEE, No. 0-7803-4426-6/98, 1998.

[14] N. D. Vaughn, J. B. Gamble, "Modeling and Simulation of a Proportional Solenoid Valve," Transactions of the ASME, Vol. 118, March 1996.

[15] J. L. Shearer, "The Effects of Radial Clearance, Rounded Corners, and Underlap on Servovalve Characteristics," Proceedings of the Joint Automatic Control Conference, American Society of Mechanical Engineers, New Yourk, New York, 1980.

[16] B. Yao, F. Bu, and G. T. C. Chiu, "Nonlinear Adaptive Robust Control of Electro-Hydraulic Servo Systems with Discontinuous Projections," Proceedings of the 37th IEEE Conference on Decision & Control, Tampa, FL, December 1998.

[17] M. Jelali, and H. Schwarz, "Nonlinear Identification of Hydraulic Servo-Drive Systems," IEEE Control Systems, No. 072-1708/95, October, 1995.

[18] T. W. McLain, E. K. Iverson, C. C. Davis, and S.C. Jacobsen, "Development, Simulation, and Validation of a Highly Nonlinear Hydraulic Servosystem Model," Proceedings of the American Control Conference, Pittsburgh PA, July 21-23, 1989.

[19] Mannesmann Rexroth, "MDSD (Series 2X) Mobile Dual Solenoid Driver Specifications," Version RA 29 864 / 11.95.

[20] F. D. Norvelle, Electrohydraulic Control Systems, Prentice Hall, 2000.

[21] W. Stadler, *Analytical Robotics and Mechatronics*, McGraw-Hill, Inc., 1995.

Appendix

Appendix A-- smthenc.m

```
function outdata = smthenc(indata)

end_of_data = 0;
i0 = 1;
i1 = i0;
outdata = 0*indata;

while end_of_data==0,
  x0 = indata(i0);
  x1 = x0;
  while ( (x0 == x1) & (end_of_data==0) ),
    i1 = i1+1;
    end_of_data = ( i1 == length(indata) );
    x1 = indata(i1);
  end
  local_slope = (x1-x0)/(i1-i0);
  for j=i0:i1
    outdata(j) = x0 + local_slope*(j-i0);
  end
  i0 = i1;
end
```

Appendix B-- curset.m, Hoist Axis

```
global xnom
global leng4p tm4p volt4p ims4p exal4p spd4p w4p
global leng6p tm6p volt6p ims6p exal6p spd6p w6p
global leng9p tm9p volt9p ims9p exal9p spd9p w9p
global leng4n tm4n volt4n ims4n exal4n spd4n
global leng6n tm6n volt6n ims6n exal6n spd6n
global leng9n tm9n volt9n ims9n exal9n spd9n
global lengrp tmrp voltrp imsrp exalrp spdrp wrp
global lengrn tmrn voltrn imsrn exalrn spdrn wrn
global  leng4p1  tm4p1  volt4p1p  volt4p1n  ims4p1  exal4p1
exal4p1p exal4p1n spd4p1
global  leng6p1  tm6p1  volt6p1p  volt6p1n  ims6p1  exal6p1
exal6p1p exal6p1n spd6p1
global  leng9p1  tm9p1  volt9p1p  volt9p1n  ims9p1  exal9p1
exal9p1p exal9p1n spd9p1
global  leng4p3  tm4p3  volt4p3p  volt4p3n  ims4p3  exal4p3
exal4p3p exal4p3n spd4p3
global  leng6p3  tm6p3  volt6p3p  volt6p3n  ims6p3  exal6p3
exal6p3p exal6p3n spd6p3
global  leng9p3  tm9p3  volt9p3p  volt9p3n  ims9p3  exal9p3
exal9p3p exal9p3n spd9p3
global f1 f2 f3 f4 f5

% create the winch speed from the encoder data
fco_lo = 10.0*2*pi;  %10 Hz
fco_hi = 30.0*2*pi; %30 Hz
dt = 1/512;
% the b and a vector digital coefficients for a derivative
filter
bd_d_lo = [fco_lo -fco_lo];
ad_d_lo = [1 dt*fco_lo-1];

bd_d_hi = [fco_hi -fco_hi];
ad_d_hi = [1 dt*fco_hi-1];

% the b and a vector digital coefficients for a low pass
bd_lo = [0 fco_lo*dt];
ad_lo = [1 dt*fco_lo-1];

bd_hi = [0 fco_hi*dt];
ad_hi = [1 dt*fco_hi-1];

load datah4
leng4p = 512*10.0;
```

```
tm4p   = tout(1,1:leng4p);
volt4p = tout(2,1:leng4p);
ims4p  = tout(3,1:leng4p);
enc4p  = tout(4,1:leng4p)/400*2*pi;
enc4p  = enc4p - enc4p(1);
enc4p  = smthenc(enc4p);
spd4p  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc4p));

load datah6
leng6p = 512*10.0;
tm6p   = tout(1,1:leng6p);
volt6p = tout(2,1:leng6p);
ims6p  = tout(3,1:leng6p);
enc6p  = tout(4,1:leng6p)/400*2*pi;
enc6p  = enc6p - enc6p(1);
enc6p  = smthenc(enc6p);
spd6p  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc6p));

load datah9
leng9p = 512*10.0;
tm9p   = tout(1,1:leng9p);
volt9p = tout(2,1:leng9p);
ims9p  = tout(3,1:leng9p);
enc9p  = tout(4,1:leng9p)/400*2*pi;
enc9p  = enc9p - enc9p(1);
enc9p  = smthenc(enc9p);
spd9p  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc9p));

load datah4_nb
leng4n = 512*10.0;
tm4n   = tout(1,1:leng4n);
volt4n = -tout(2,1:leng4n);
ims4n  = tout(3,1:leng4n);
enc4n  = tout(4,1:leng4n)/400*2*pi;
enc4n  = enc4n - enc4n(1);
enc4n  = smthenc(enc4n);
spd4n  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc4n));

load datah6_nb
leng6n = 512*10.0;
tm6n   = tout(1,1:leng6n);
volt6n = -tout(2,1:leng6n);
ims6n  = tout(3,1:leng6n);
enc6n  = tout(4,1:leng6n)/400*2*pi;
enc6n  = enc6n - enc6n(1);
enc6n  = smthenc(enc6n);
```

```
spd6n  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc6n));

load datah9_nb
leng9n = 512*10.0;
tm9n   = tout(1,1:leng9n);
volt9n = -tout(2,1:leng9n);
ims9n  = tout(3,1:leng9n);
enc9n  = tout(4,1:leng9n)/400*2*pi;
enc9n  = enc9n - enc9n(1);
enc9n  = smthenc(enc9n);
spd9n  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc9n));

load datahr_p
lengrp = 512*13.0;
tmrp   = tout(1,1:lengrp);
voltrp = tout(2,1:lengrp);
imsrp  = tout(3,1:lengrp);
encrp  = tout(4,1:lengrp)/400*2*pi;
encrp  = encrp - encrp(1);
encrp  = smthenc(encrp);
spdrp  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,encrp));

load datahr_n
lengrn = 512*13.0;
tmrn   = tout(1,1:lengrn);
voltrn = -tout(2,1:lengrn);
imsrn  = tout(3,1:lengrn);
encrn  = tout(4,1:lengrn)/400*2*pi;
encrn  = encrn - encrn(1);
encrn  = smthenc(encrn);
spdrn  = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,encrn));

load datah4p1
leng4p1 = 512*30.0;
tm4p1   = tout(1,1:leng4p1);
volt4p1 = tout(2,1:leng4p1);
ims4p1  = tout(3,1:leng4p1);
enc4p1  = tout(4,1:leng4p1)/400*2*pi;
enc4p1  = enc4p1 - enc4p1(1);
enc4p1  = smthenc(enc4p1);
spd4p1 = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc4p1));
volt4p1p = zeros(size(volt4p1));
volt4p1n = zeros(size(volt4p1));
for i=1:leng4p1
  if volt4p1(i) >= -0.000001
    volt4p1p(i) = volt4p1(i);
```

```
  else
    volt4p1p(i) = 0.0;
  end;
  if volt4p1(i) <= 0.0000001
    volt4p1n(i) = -volt4p1(i);
  else
    volt4p1n(i) = 0.0;
  end;
end;

load datah6p1
leng6p1 = 512*30.0;
tm6p1   = tout(1,1:leng6p1);
volt6p1 = tout(2,1:leng6p1);
ims6p1  = tout(3,1:leng6p1);
enc6p1  = tout(4,1:leng6p1)/400*2*pi;
enc6p1  = enc6p1 - enc6p1(1);
enc6p1  = smthenc(enc6p1);
spd6p1 = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc6p1));
volt6p1p = zeros(size(volt6p1));
volt6p1n = zeros(size(volt6p1));
for i=1:leng6p1
  if volt6p1(i) >= -0.000001
    volt6p1p(i) = volt6p1(i);
  else
    volt6p1p(i) = 0.0;
  end;
  if volt6p1(i) <= 0.0000001
    volt6p1n(i) = -volt6p1(i);
  else
    volt6p1n(i) = 0.0;
  end;
end;

load datah9p1
leng9p1 = 512*30.0;
tm9p1   = tout(1,1:leng9p1);
volt9p1 = tout(2,1:leng9p1);
ims9p1  = tout(3,1:leng9p1);
enc9p1  = tout(4,1:leng9p1)/400*2*pi;
enc9p1  = enc9p1 - enc9p1(1);
enc9p1  = smthenc(enc9p1);
spd9p1 = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc9p1));
volt9p1p = zeros(size(volt9p1));
volt9p1n = zeros(size(volt9p1));
for i=1:leng9p1
```

```
  if volt9p1(i) >= -0.000001
    volt9p1p(i) = volt9p1(i);
  else
    volt9p1p(i) = 0.0;
  end;
  if volt9p1(i) <= 0.0000001
    volt9p1n(i) = -volt9p1(i);
  else
    volt9p1n(i) = 0.0;
  end;
end;

load datah6p3
leng6p3 = 512*20.0;
tm6p3   = tout(1,1:leng6p3);
volt6p3 = tout(2,1:leng6p3);
ims6p3  = tout(3,1:leng6p3);
enc6p3  = tout(4,1:leng6p3)/400*2*pi;
enc6p3  = enc6p3 - enc6p3(1);
enc6p3  = smthenc(enc6p3);
spd6p3 = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc6p3));
volt6p3p = zeros(size(volt6p3));
volt6p3n = zeros(size(volt6p3));
for i=1:leng6p3
  if volt6p3(i) >= -0.000001
    volt6p3p(i) = volt6p3(i);
  else
    volt6p3p(i) = 0.0;
  end;
  if volt6p3(i) <= 0.0000001
    volt6p3n(i) = -volt6p3(i);
  else
    volt6p3n(i) = 0.0;
  end;
end;

load datah9p3b
leng9p3 = 512*20.0;
tm9p3   = tout(1,1:leng9p3);
volt9p3 = tout(2,1:leng9p3);
ims9p3  = tout(3,1:leng9p3);
enc9p3  = tout(4,1:leng9p3)/400*2*pi;
enc9p3  = enc9p3 - enc9p3(1);
enc9p3  = smthenc(enc9p3);
spd9p3 = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc9p3));
volt9p3p = zeros(size(volt9p3));
```

```
volt9p3n = zeros(size(volt9p3));
for i=1:leng9p3
  if volt9p3(i) >= -0.000001
    volt9p3p(i) = volt9p3(i);
  else
    volt9p3p(i) = 0.0;
  end;
  if volt9p3(i) <= 0.0000001
    volt9p3n(i) = -volt9p3(i);
  else
    volt9p3n(i) = 0.0;
  end;
end;

load datah4p3
leng4p3 = 512*20.0;
tm4p3   = tout(1,1:leng4p3);
volt4p3 = tout(2,1:leng4p3);
ims4p3  = tout(3,1:leng4p3);
enc4p3  = tout(4,1:leng4p3)/400*2*pi;
enc4p3  = enc4p3 - enc4p3(1);
enc4p3  = smthenc(enc4p3);
spd4p3 = filter(bd_d_hi,ad_d_hi,filter(bd_lo,ad_lo,enc4p3));
volt4p3p = zeros(size(volt4p3));
volt4p3n = zeros(size(volt4p3));
for i=1:leng4p3
  if volt4p3(i) >= -0.000001
    volt4p3p(i) = volt4p3(i);
  else
    volt4p3p(i) = 0.0;
  end;
  if volt4p3(i) <= 0.0000001
    volt4p3n(i) = -volt4p3(i);
  else
    volt4p3n(i) = 0.0;
  end;
end;

% The noise increases with voltage amplitude. So, a gain
% is applied to the error to deweight higher voltage inputs.
% This is needed for ramp and sine data

w4p = ones(size(volt4p));
w4p(3072:3276) = w4p(3072:3276)*5;
w4p(512:614)   = w4p(512:614)*10;
w6p = ones(size(volt6p));
```

```
w6p(3072:3276) = w6p(3072:3276)*5;
w6p(512:614)   = w6p(512:614)*10;
w9p = ones(size(volt9p));
w9p(3072:3276) = w9p(3072:3276)*5;
w9p(512:614)   = w9p(512:614)*10;

V0 = 3;
for i=1:length(voltrn)
  if abs(voltrn(i)) > V0
    wrn(i) = 1-(.5/(10-V0))*(voltrn(i)-V0);
  else
    wrn(i) = 1;
  end;
end;
for i=1:length(voltrp)
  if abs(voltrp(i)) > V0
    wrp(i) = 1-(.5/(10-V0))*(voltrp(i)-V0);
  else
    wrp(i) = 1;
  end;
end;

xnom=[     0.12663
    0.097857
     0.27473
      0.2951
    0.049315
    0.054706
     0.71887
      56.253
           0
     0.13724
     0.33628
     0.10683
     0.34863
      2.2615
      2.2841
      2.6483
      2.2733
      19.131
      21.379
  0.00081975
 -4.5317e-05
  0.00042879
 -9.7369e-05
     0.72563
```

```
      61.833]';

myopts = foptions;
% number of equality constraints
myopts(13) = 0;
% max delta for gradients
myopts(16) = 0.001;
% max iterations
myopts(14) = 18*1000;

x0 = ones(25,1);

vlb = x0*0.5;
vub = x0*2;

vlb(20) = -5;
vlb(21) = -5;
vlb(22) = -5;
vlb(23) = -5;
vub(20) = 5;
vub(21) = 5;
vub(22) = 5;
vub(23) = 5;

f1=figure;
f2=figure;
f3=figure;
f4=figure;

%x = constr('curwrap',x0,myopts,vlb,vub);
[ef,ge]=curwrap(x0);
```

Appendix C-- curwrap.m, Hoist Axis

```
function [err,gc] = curwrap(x)

global xnom
global leng4p tm4p volt4p ims4p exal4p spd4p w4p
global leng6p tm6p volt6p ims6p exal6p spd6p w6p
global leng9p tm9p volt9p ims9p exal9p spd9p w9p
global leng4n tm4n volt4n ims4n exal4n spd4n
global leng6n tm6n volt6n ims6n exal6n spd6n
global leng9n tm9n volt9n ims9n exal9n spd9n
global lengrp tmrp voltrp imsrp exalrp spdrp wrp
global lengrn tmrn voltrn imsrn exalrn spdrn wrn
global  leng4p1  tm4p1  volt4p1p  volt4p1n  ims4p1  exal4p1
exal4p1p exal4p1n spd4p1
global  leng6p1  tm6p1  volt6p1p  volt6p1n  ims6p1  exal6p1
exal6p1p exal6p1n spd6p1
global  leng9p1  tm9p1  volt9p1p  volt9p1n  ims9p1  exal9p1
exal9p1p exal9p1n spd9p1
global  leng4p3  tm4p3  volt4p3p  volt4p3n  ims4p3  exal4p3
exal4p3p exal4p3n spd4p3
global  leng6p3  tm6p3  volt6p3p  volt6p3n  ims6p3  exal6p3
exal6p3p exal6p3n spd6p3
global  leng9p3  tm9p3  volt9p3p  volt9p3n  ims9p3  exal9p3
exal9p3p exal9p3n spd9p3
global f1 f2 f3 f4 f5

optrun = 1;

xnew = (xnom').*x;

nzlev = 0.015;

% evaluate 1-sided inputs
exal4p = currcost(1/512,leng4p/512,1,nzlev,xnew,volt4p);
exal6p = currcost(1/512,leng6p/512,1,nzlev,xnew,volt6p);
exal9p = currcost(1/512,leng9p/512,1,nzlev,xnew,volt9p);
exal4n = currcost(1/512,leng4n/512,-1,nzlev,xnew,volt4n);
exal6n = currcost(1/512,leng6n/512,-1,nzlev,xnew,volt6n);
exal9n = currcost(1/512,leng9n/512,-1,nzlev,xnew,volt9n);
exalrp = currcost(1/512,lengrp/512,1,nzlev,xnew,voltrp);
exalrn = currcost(1/512,lengrn/512,-1,nzlev,xnew,voltrn);

%evaluate 2-sided inputs
exal4p1p= currcost(1/512,leng4p1/512,1,nzlev,xnew,volt4p1p);
exal4p1n=currcost(1/512,leng4p1/512,-1,nzlev,xnew,volt4p1n);
```

```
exal4p1 = exal4p1p;
exal4p1(2,:) = exal4p1p(2,:)+exal4p1n(2,:);
exal6p1p= currcost(1/512,leng6p1/512,1,nzlev,xnew,volt6p1p);
exal6p1n=currcost(1/512,leng6p1/512,-1,nzlev,xnew,volt6p1n);
exal6p1 = exal6p1p;
exal6p1(2,:) = exal6p1p(2,:)+exal6p1n(2,:);
exal9p1p= currcost(1/512,leng9p1/512,1,nzlev,xnew,volt9p1p);
exal9p1n=currcost(1/512,leng9p1/512,-1,nzlev,xnew,volt9p1n);
exal9p1 = exal9p1p;
exal9p1(2,:) = exal9p1p(2,:)+exal9p1n(2,:);
exal4p3p= currcost(1/512,leng4p3/512,1,nzlev,xnew,volt4p3p);
exal4p3n=currcost(1/512,leng4p3/512,-1,nzlev,xnew,volt4p3n);
exal4p3 = exal4p3p;
exal4p3(2,:) = exal4p3p(2,:)+exal4p3n(2,:);
exal6p3p= currcost(1/512,leng6p3/512,1,nzlev,xnew,volt6p3p);
exal6p3n currcost(1/512,leng6p3/512,-1,nzlev,xnew,volt6p3n);
exal6p3 = exal6p3p;
exal6p3(2,:) = exal6p3p(2,:)+exal6p3n(2,:);
exal9p3p= currcost(1/512,leng9p3/512,1,nzlev,xnew,volt9p3p);
exal9p3n=currcost(1/512,leng9p3/512,-1,nzlev,xnew,volt9p3n);
exal9p3 = exal9p3p;
exal9p3(2,:) = exal9p3p(2,:)+exal9p3n(2,:);

if optrun == 1
figure(f1);
subplot(3,1,1);
plot(tm4p,ims4p,exal4p(1,1:leng4p),exal4p(2,1:leng4p),'m');g
rid;
subplot(3,1,2);
plot(tm6p,ims6p,exal6p(1,1:leng6p),exal6p(2,1:leng6p),'m');g
rid;
subplot(3,1,3);
plot(tm9p,ims9p,exal9p(1,1:leng9p),exal9p(2,1:leng9p),'m');g
rid;
figure(f2);
subplot(3,1,1);
plot(tm4n,ims4n,exal4n(1,1:leng4n),exal4n(2,1:leng4n),'m');g
rid;
subplot(3,1,2);
plot(tm6n,ims6n,exal6n(1,1:leng6n),exal6n(2,1:leng6n),'m');g
rid;
subplot(3,1,3);
plot(tm9n,ims9n,exal9n(1,1:leng9n),exal9n(2,1:leng9n),'m');g
rid;
figure(f3);
subplot(4,1,1);
```

```
plot(tm4p1,ims4p1,exal4p1(1,1:leng4p1),exal4p1(2,1:leng4p1),
'm');grid;
subplot(4,1,2);
plot(tm6p1,ims6p1,exal6p1(1,1:leng6p1),exal6p1(2,1:leng6p1),
'm');grid;
subplot(4,1,3);
plot(tm9p1,ims9p1,exal9p1(1,1:leng9p1),exal9p1(2,1:leng9p1),
'm');grid;
subplot(4,1,4);
plot(tm6p3,ims6p3,exal6p3(1,1:leng6p3),exal6p3(2,1:leng6p3),
'm');grid;
figure(f4);
subplot(2,1,1);
plot(tmrn,imsrn,exalrn(1,1:lengrn),exalrn(2,1:len-
grn),'m');grid;
ylabel('neg ramp');
subplot(2,1,2);
plot(tmrp,imsrp,exalrp(1,1:lengrp),exalrp(2,1:len-
grp),'m');grid;
ylabel('pos ramp');
pause(0.1);

else


% save data files
tout = [exal4p(1:2,1:leng4p)' spd4p' volt4p'];
save sc4p tout
tout = [exal6p(1:2,1:leng6p)' spd6p' volt6p'];
save sc6p tout
tout = [exal9p(1:2,1:leng9p)' spd9p' volt9p'];
save sc9p tout
tout  =  [exal4n(1,1:leng4n)'  -exal4n(2,1:leng4n)'  spd4n'
volt4n'];
save sc4n tout
tout  =  [exal6n(1:1,1:leng6n)'  -exal6n(2,1:leng6n)'  spd6n'
volt6n'];
save sc6n tout
tout  =  [exal9n(1,1:leng9n)'  -exal9n(2,1:leng9n)'  spd9n'
volt9n'];
save sc9n tout
tout = [exalrp(1:2,1:lengrp)' spdrp' voltrp'];
save scrp tout
tout = [exalrn(1,1:lengrn)' -exalrn(2,1:lengrn)' spdrn' vol-
trn'];
save scrn tout
```

```
tout   =   [exal4p1p(1,1:leng4p1)'   exal4p1p(2,1:leng4p1)'-
exal4p1n(2,1:leng4p1)' spd4p1' ...
   (volt4p1p-volt4p1n)'];
save sc4p1 tout
tout   =   [exal6p1p(1,1:leng6p1)'   exal6p1p(2,1:leng6p1)'-
exal6p1n(2,1:leng6p1)' spd6p1' ...
   (volt6p1p-volt6p1n)'];
save sc6p1 tout
tout   =   [exal9p1p(1,1:leng9p1)'   exal9p1p(2,1:leng9p1)'-
exal9p1n(2,1:leng9p1)' spd9p1' ...
   (volt9p1p-volt9p1n)'];
save sc9p1 tout
tout   =   [exal4p3p(1,1:leng4p3)'   exal4p3p(2,1:leng4p3)'-
exal4p3n(2,1:leng4p3)' spd4p3' ...
   (volt4p3p-volt4p3n)'];
save sc4p3 tout
tout   =   [exal6p3p(1,1:leng6p3)'   exal6p3p(2,1:leng6p3)'-
exal6p3n(2,1:leng6p3)' spd6p3' ...
   (volt6p3p-volt6p3n)'];
save sc6p3 tout
tout   =   [exal9p3p(1,1:leng9p3)'   exal9p3p(2,1:leng9p3)'-
exal9p3n(2,1:leng9p3)' spd9p3' ...
   (volt9p3p-volt9p3n)'];
save sc9p3 tout

%make plots and save them
%figure;
%plot(tm4p,ims4p,'b',exal4p(1,1:leng4p),exal4p(2,1:leng4p),'
k',...
%     tm4p,volt4p/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig4p

%figure;
%plot(tm6p,ims6p,'b',exal6p(1,1:leng6p),exal6p(2,1:leng6p),'
k',...
%     tm6p,volt6p/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig6p
```

```
%figure;
%plot(tm9p,ims9p,'b',exal9p(1,1:leng9p),exal9p(2,1:leng9p),'
k',...
%     tm9p,volt9p/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig9p

%figure;
%plot(tm4n,ims4n,'b',exal4n(1,1:leng4n),exal4n(2,1:leng4n),'
k',...
%     tm4n,volt4n/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig4n

%figure;
%plot(tm6n,ims6n,'b',exal6n(1,1:leng6n),exal6n(2,1:leng6n),'
k',...
%     tm6n,volt6n/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig6n

%figure;
%plot(tm9n,ims9n,'b',exal9n(1,1:leng9n),exal9n(2,1:leng9n),'
k',...
%     tm9n,volt9n/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig9n

%figure;
%plot(tmrp,imsrp,'b',exalrp(1,1:lengrp),exalrp(2,1:len-
grp),'k',...
%     tmrp,voltrp/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
```

```
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 figrp

%figure;
%plot(tm4p1,ims4p1,'b',exal4p1(1,1:leng4p1),exal4p1(2,1:leng
4p1),'k',...
%     tm4p1,(volt4p1p-volt4p1n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig4p1

%figure;
%plot(tm6p1,ims6p1,'b',exal6p1(1,1:leng6p1),exal6p1(2,1:leng
6p1),'k',...
%     tm6p1,(volt6p1p-volt6p1n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig6p1

%figure;
%plot(tm9p1,ims9p1,'b',exal9p1(1,1:leng9p1),exal9p1(2,1:leng
9p1),'k',...
%     tm9p1,(volt9p1p-volt9p1n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig9p1

%figure;
%%plot(tm4p3,ims4p3,'b',exal4p3(1,1:leng4p3),exal4p3(2,1:len
g4p3),'k',...
%     tm4p3,(volt4p3p-volt4p3n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig4p3

%figure;
```

```
%plot(tm6p3,ims6p3,'b',exal6p3(1,1:leng6p3),exal6p3(2,1:leng
6p3),'k',...
%     tm6p3,(volt6p3p-volt6p3n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig6p3

%figure;
%plot(tm9p3,ims9p3,'b',exal9p3(1,1:leng9p3),exal9p3(2,1:leng
9p3),'k',...
%     tm6p3,(volt6p3p-volt6p3n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig6p3

%figure;
%plot(tm9p3,ims9p3,'b',exal9p3(1,1:leng9p3),exal9p3(2,1:leng
9p3),'k',...
%     tm9p3,(volt9p3p-volt9p3n)/10,'m');grid;
%legend('Measured   Current','Simulated   Current','Voltage
Input/10');
%xlabel('Time (sec)');
%ylabel('Current (A), Voltage (.1V)');
%print -depsc2 fig9p3

end

err4p = (exal4p(2,1:leng4p)-ims4p).*w4p;
err6p = (exal6p(2,1:leng6p)-ims6p).*w6p;
err9p = (exal9p(2,1:leng9p)-ims9p).*w9p;
err4n = (exal4n(2,1:leng4n)-ims4n).*w4p;
err6n = (exal6n(2,1:leng6n)-ims6n);
err9n = (exal9n(2,1:leng9n)-ims9n).*w9p;
err4p1 = (exal4p1(2,1:leng4p1)-ims4p1);
err6p1 = (exal6p1(2,1:leng6p1)-ims6p1);
err9p1 = (exal9p1(2,1:leng9p1)-ims9p1);
err6p3 = (exal6p3(2,1:leng6p3)-ims6p3);
errrn = (exalrn(2,1:lengrn)-imsrn).*wrn;
errrp = (exalrp(2,1:lengrp)-imsrp).*wrp;
err = 0 + ...
      sum(err4p.*err4p) + ...
      sum(err6p.*err6p) + ...
```

Appendix D-- elset.m, Hoist Axis

```
clear all
global deltime tfin ic ovs xnom exal
global currh4p1 mtmh4p1 mwspdh4p1 lengh4p1 mvh4p1
global currh6p1 mtmh6p1 mwspdh6p1 lengh6p1 mvh6p1
global currh9p1 mtmh9p1 mwspdh9p1 lengh9p1 mvh9p1
global currh6p3 mtmh6p3 mwspdh6p3 lengh6p3 mvh6p3
global currh9p3 mtmh9p3 mwspdh9p3 lengh9p3 mvh9p3
global currh4 mtmh4 mwspdh4 lengh4 mvh4
global currh6 mtmh6 mwspdh6 lengh6 mvh6
global currh9 mtmh9 mwspdh9 lengh9 mvh9
global currhr mtmhr mwspdhr lenghr exalhr mvhr
global currhrn mtmhrn mwspdhrn lenghrn mvhrn

global f1 f2 f3

global exalh4 exalh6 exalh9 exalhr exalhrn exalh4p1
global exalh6p1 exalh9p1 exalh6p3 exalh9p3

deltime = 1/512;
tfin = 10.0;

load sc4p
lengh4 = length(tout);
currh4 = tout(:,2);
mtmh4  = tout(:,1);
mwspdh4 = tout(:,3)*400/2526.3;
mvh4  = tout(:,4);

load sc6p
lengh6 = length(tout);
currh6 = tout(:,2);
mtmh6  = tout(:,1);
mwspdh6 = tout(:,3)*400/2526.3;
mvh6  = tout(:,4);

load sc9p
lengh9 = length(tout);
currh9 = tout(:,2);
mtmh9  = tout(:,1);
mwspdh9 = tout(:,3)*400/2526.3;
mvh9  = tout(:,4);

load scrp
lenghr = length(tout);
currhr = tout(:,2);
```

```
mtmhr  = tout(:,1);
mwspdhr = tout(:,3)*400/2526.3;
mvhr  = tout(:,4);

load scrn
lenghrn = length(tout);
currhrn = tout(:,2);
mtmhrn  = tout(:,1);
mwspdhrn = tout(:,3)*400/2526.3;
mvhrn  = tout(:,4);

load sc4p1
lengh4p1 = length(tout);
currh4p1 = tout(:,2);
mtmh4p1  = tout(:,1);
mwspdh4p1 = tout(:,3)*400/2526.3;
mvh4p1  = tout(:,4);

load sc6p1
lengh6p1 = length(tout);
currh6p1 = tout(:,2);
mtmh6p1  = tout(:,1);
mwspdh6p1 = tout(:,3)*400/2526.3;
mvh6p1  = tout(:,4);

load sc9p1
lengh9p1 = length(tout);
currh9p1 = tout(:,2);
mtmh9p1  = tout(:,1);
mwspdh9p1 = tout(:,3)*400/2526.3;
mvh9p1  = tout(:,4);

load sc6p3
lengh6p3 = length(tout);
currh6p3 = tout(:,2);
mtmh6p3  = tout(:,1);
mwspdh6p3 = tout(:,3)*400/2526.3;
mvh6p3  = tout(:,4);

load sc9p3
lengh9p3 = length(tout);
currh9p3 = tout(:,2);
mtmh9p3  = tout(:,1);
mwspdh9p3 = tout(:,3)*400/2526.3;
mvh9p3  = tout(:,4);
```

```
xnom=[  4.6441e-07
             0
      0.053604
     0.0079568
      0.073926
             0
    1.6514e+06
        12.396
             0
             0
       0.27596
     0.0036461
    4.1191e+06
    0.00011634
    1.0743e+07
    2.1667e-05
        12.349
       0.27028
       0.22421
       0.27867
       0.62542
       0.26047
    0.00015126
        10.019
        3.7297
        1.6581
       -54.153]';

xnom = xnom';

myopts = foptions;
% number of equality constraints
myopts(13) = 0;
% max delta for gradients
myopts(16) = 0.01;
% max iterations
myopts(14) = 18*1000;

x0 = ones(27,1);

vlb = x0*0.8;
vub = x0*1.4;

vlb(1) = .8;
vub(1) = 2;
```

```
vlb(3) = 0.8;
vub(3) = 2.;

vlb(5) = 0.8;
vub(5) = 2.;

vlb(6) = 0.8;
vub(6) = 2;

vlb(7) = 0.8;
vub(7) = 2;

vlb(8) = 0.95;
vub(8) = 1.05;

vlb(10) = -1.5;
vub(10) = 1.5;

vlb(11) = 0.8;
vub(11) = 2.;

vlb(13) = 0.8;
vub(13) = 2.;

vlb(14) = 0.8;
vub(14) = 2;

vlb(15) = 0.8;
vub(15) = 5;

vlb(18) = -5;
vub(18) = 5;

vlb(19) = -1.8;
vub(19) = 1.8;

vlb(20) = 0.85;
vub(20) = 1.15;

vlb(22) = 0.3;
vub(22) = 3.;

vlb(23) = 0.3;
vub(23) = 3;

vlb(23) = 0.5;
```

```
vub(23) = 10;

vlb(25) = -1.5;
vub(25) = 1.5;

vlb(26) = -1.5;
vub(26) = 1.5;

vlb(27) = -1.5;
vub(27) = 1.5;

f1 = figure;
f2 = figure;
f3 = figure;

%x = constr('elwrap_hoist2',x0,myopts,vlb,vub);
[ef,ge]=elwrap_hoist2(x0);
```

Appendix E-- elwrap_hoist2.m, Hoist Axis

```
function [err,gc] = elwrap(x)

global deltime tfin ic ovs xnom exal
global currh4p1 mtmh4p1 mwspdh4p1 lengh4p1 mvh4p1
global currh6p1 mtmh6p1 mwspdh6p1 lengh6p1 mvh6p1
global currh9p1 mtmh9p1 mwspdh9p1 lengh9p1 mvh9p1
global currh6p3 mtmh6p3 mwspdh6p3 lengh6p3 mvh6p3
global currh9p3 mtmh9p3 mwspdh9p3 lengh9p3 mvh9p3
global currh4 mtmh4 mwspdh4 lengh4 mvh4
global currh6 mtmh6 mwspdh6 lengh6 mvh6
global currh9 mtmh9 mwspdh9 lengh9 mvh9
global currhr mtmhr mwspdhr lenghr exalhr mvhr
global currhrn mtmhrn mwspdhrn lenghrn mvhrn

global f1 f2 f3

global exalh4 exalh6 exalh9 exalhr exalhrn exalh4p1
global exalh6p1 exalh9p1 exalh6p3 exalh9p3

printplots = 1;

% create the winch speed from the encoder data
fco_lo = 10.0*2*pi;  %10 Hz
fco_hi = 30.0*2*pi; %30 Hz
dt = 1/512;
% the b and a vector digital coefficients for a derivative
filter
bd_d_lo = [fco_lo -fco_lo];
ad_d_lo = [1 dt*fco_lo-1];

bd_d_hi = [fco_hi -fco_hi];
ad_d_hi = [1 dt*fco_hi-1];

% the b and a vector digital coefficients for a low pass
bd_lo = [0 fco_lo*dt];
ad_lo = [1 dt*fco_lo-1];

bd_hi = [0 fco_hi*dt];
ad_hi = [1 dt*fco_hi-1];

% don’t change the plussing
x(16) = 1;
% don’t change Z11
x(9) = 1;
% for poly fit only
```

```
%x(1:23) = ones(23,1);


ovs = x.*xnom;
% set up the initial condition based on the swash plate equa-
tion
ic = ovs(2)/ovs(1);

%set the low alpha limit based on the motor gain
%ovs(22) = 185/180*pi/ovs(8);
%set the high alpha limit based on the motor gain
%ovs(11) = 196/180*pi/ovs(8);

exalh4 = elcost(deltime,10.0,ic,ovs,currh4);
exalh6 = elcost(deltime,10.0,ic,ovs,currh6);
exalh9 = elcost(deltime,10.0,ic,ovs,currh9);
exalhr = elcost(deltime,13.0,ic,ovs,currhr);
exalhrn = elcost(deltime,13.0,ic,ovs,currhrn);
exalh4p1 = elcost(deltime,30.0,ic,ovs,currh4p1);
exalh6p1 = elcost(deltime,30.0,ic,ovs,currh6p1);
exalh9p1 = elcost(deltime,30.0,ic,ovs,currh9p1);
exalh6p3 = elcost(deltime,20.0,ic,ovs,currh6p3);
exalh9p3 = elcost(deltime,20.0,ic,ovs,currh9p3);

exalh4(10,1:lengh4) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh4(10,1:lengh4)));
exalh6(10,1:lengh6) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh6(10,1:lengh6)));
exalh9(10,1:lengh9) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh9(10,1:lengh9)));
exalhr(10,1:lenghr) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalhr(10,1:lenghr)));
exalhrn(10,1:lenghrn) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalhrn(10,1:lenghrn)));
exalh4p1(10,1:lengh4p1) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh4p1(10,1:lengh4p1)));
exalh6p1(10,1:lengh6p1) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh6p1(10,1:lengh6p1)));
exalh9p1(10,1:lengh9p1) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh9p1(10,1:lengh9p1)));
exalh6p3(10,1:lengh6p3) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh6p3(10,1:lengh6p3)));
exalh9p3(10,1:lengh9p3) = filter(bd_hi,ad_hi,...
  filter(bd_lo,ad_lo,exalh9p3(10,1:lengh9p3)));

figure(f1);
```

```
subplot(5,1,1);
plot(exalh4p1(1,1:lengh4p1),exalh4p1(10,1:lengh4p1),...
  'k',mtmh4p1,mwspdh4p1*180/pi);grid;
legend('Simulation','Experiment');
subplot(5,1,2);
plot(exalh6p1(1,1:lengh6p1),exalh6p1(10,1:lengh6p1),...
  'k',mtmh6p1,mwspdh6p1*180/pi);grid;
ylabel('Hoist Winch Speed (deg/s)');
subplot(5,1,3);
plot(exalh9p1(1,1:lengh9p1),exalh9p1(10,1:lengh9p1),...
  'k',mtmh9p1,mwspdh9p1*180/pi);grid;
xlabel('Time (sec)');
subplot(5,1,4);
plot(exalh6p3(1,1:lengh6p3),exalh6p3(10,1:lengh6p3),...
  'k',mtmh6p3,mwspdh6p3*180/pi);grid;
xlabel('Time (sec)');
subplot(5,1,5);
plot(exalh9p3(1,1:lengh9p3),exalh9p3(10,1:lengh9p3),...
  'k',mtmh9p3,mwspdh9p3*180/pi);grid;
xlabel('Time (sec)');
figure(f2);
subplot(5,1,1);
plot(exalhr(1,1:lenghr),exalhr(10,1:lenghr),...
  'k',mtmhr,mwspdhr*180/pi);grid;
subplot(5,1,2);
plot(exalhrn(1,1:lenghrn),exalhrn(10,1:lenghrn),...
  'k',mtmhrn,mwspdhrn*180/pi);grid;
subplot(5,1,3);
plot(exalh4(1,1:lengh4),exalh4(10,1:lengh4),...
  'k',mtmh4,mwspdh4*180/pi);grid;
ylabel('Hoist Winch Speed (deg/s)');
subplot(5,1,4);
plot(exalh6(1,1:lengh4),exalh6(10,1:lengh4),...
  'k',mtmh6,mwspdh6*180/pi);grid;
xlabel('Time (sec)');
subplot(5,1,5);
plot(exalh9(1,1:lengh9),exalh9(10,1:lengh9),...
  'k',mtmh9,mwspdh9*180/pi);grid;
xlabel('Time (sec)');

figure(3);
subplot(5,1,1);
plot(exalh9p1(1,1:lengh9p1),exalh9p1(10,1:lengh9p1),...
  'k',mtmh9p1,mwspdh9p1*180/pi);grid;
set(gca,'FontSize',8);
subplot(5,1,2);
```

```
plot(exalh9p1(1,1:lengh9p1),exalh9p1(3,1:lengh9p1));...
  grid;ylabel('xsp');
set(gca,'FontSize',8);
subplot(5,1,3);
plot(exalh9p1(1,1:lengh9p1),exalh9p1(2,1:lengh9p1));...
  grid;ylabel('alf');
set(gca,'FontSize',8);
subplot(5,1,4);
plot(exalh9p1(1,1:lengh9p1),exalh9p1(4,1:lengh9p1));...
  grid;ylabel('P');
set(gca,'FontSize',8);
subplot(5,1,5);
plot(exalh9p1(1,1:lengh9p1),exalh9p1(8,1:lengh9p1));...
  grid;ylabel('F');
set(gca,'FontSize',8);


if printplots == 1
% start print plots here
figure;
plot(exalh4p1(1,1:15000),exalh4p1(10,1:15000),'k',...
  mtmh4p1,mwspdh4p1*180/pi,'c',...
  mtmh4p1,10*mvh4p1,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h4p1

figure;
plot(exalh6p1(1,1:15000),exalh6p1(10,1:15000),'k',...
  mtmh6p1,mwspdh6p1*180/pi,'c',...
  mtmh6p1,10*mvh6p1,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h6p1

figure;
plot(exalh9p1(1,1:15000),exalh9p1(10,1:15000),'k',...
  mtmh9p1,mwspdh9p1*180/pi,'c',...
  mtmh9p1,10*mvh9p1,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h9p1
```

```
figure;
subplot(5,1,1);
plot(exalh9p1(1,1:15000),exalh9p1(10,1:15000),'k',...
  mtmh9p1,mwspdh9p1*180/pi,'c');
ylabel('Winch Speed (deg/sec)');
grid;
legend('Simulation','Experiment');

subplot(5,1,2);
plot(mtmh9p1,currh9p1,'k');
ylabel('Current (amp)');
grid;
set(get(gca,'XLabel'),'Visible','off');

subplot(5,1,3);
plot(exalh9p1(1,1:15000),1000*exalh9p1(3,1:15000),'k');
grid;
ylabel('Spool Position (mm)');

subplot(5,1,4);
plot(exalh9p1(1,1:15000),exalh9p1(2,1:15000),'k');
grid;
ylabel('Swash Plate Angle (deg)');

subplot(5,1,5);
plot(exalh9p1(1,1:15000),exalh9p1(4,1:15000)/1e6,'k');
grid;
ylabel('Stroker Pressure (MPa)');
print -depsc2 hcombo


figure;
plot(exalh6p3(1,1:10000),exalh6p3(10,1:10000),'k',...
  mtmh6p3,mwspdh6p3*180/pi,'c',...
  mtmh6p3,10*mvh6p3,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h6p3

figure
plot(exalh9p3(1,1:10000),exalh9p3(10,1:10000),'k',...
  mtmh9p3,mwspdh9p3*180/pi,'c',...
  mtmh9p3,10*mvh9p3,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
```

```
ylabel('Winch Speed (deg/sec)');
print -depsc2 h9p3

figure
plot(exalh9(1,1:5000),exalh9(10,1:5000),'k',mtmh9,mwspdh9*18
0/pi,'c',...
  mtmh9,10*mvh9,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h9

figure
plot(exalh6(1,1:5000),exalh6(10,1:5000),'k',mtmh6,mwspdh6*18
0/pi,'c',...
  mtmh6,10*mvh6,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h6

figure
plot(exalh4(1,1:5000),exalh4(10,1:5000),'k',mtmh4,mwspdh4*18
0/pi,'c',...
  mtmh4,10*mvh4,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 h4

figure
plot(exalhr(1,1:5500),exalhr(10,1:5500),'k',mtmhr,mwsp-
dhr*180/pi,'c',...
  mtmhr,10*mvhr,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
print -depsc2 hr

figure
plot(exalhrn(1,1:5500),exalhrn(10,1:5500),'k',mtmhrn,mwsp-
dhrn*180/pi,'c',...
  mtmhrn,-10*mvhrn,'m');grid;
legend('Simulation','Experiment','Volts In*10');
xlabel('Time (sec)');
ylabel('Winch Speed (deg/sec)');
```

```
print -depsc2 hrn
end
% end print plots


temperrh4           =           ((exalh4(10,400:lengh4)')-
(mwspdh4(400:lengh4)*180/pi))/500;
temperrh6           =           ((exalh6(10,400:lengh6)')-
(mwspdh6(400:lengh6)*180/pi))/500;
temperrh9           =           ((exalh9(10,400:lengh9)')-
(mwspdh9(400:lengh9)*180/pi))/500;
temperrhr         =         ((exalhr(10,400:lenghr)')-(mwsp-
dhr(400:lenghr)*180/pi))/500;
temperrhrn = ((exalhrn(10,400:lenghrn)')-...
  (mwspdhrn(400:lenghrn)*180/pi))/500;
temperrh4p1 = ((exalh4p1(10,400:lengh4p1)')-...
  (mwspdh4p1(400:lengh4p1)*180/pi))/500;
temperrh6p1 = ((exalh6p1(10,400:lengh6p1)')-...
  (mwspdh6p1(400:lengh6p1)*180/pi))/500;
temperrh9p1 = ((exalh9p1(10,400:lengh9p1)')-...
  (mwspdh9p1(400:lengh9p1)*180/pi))/500;
temperrh6p3 = ((exalh6p3(10,400:lengh6p3)')-...
  (mwspdh6p3(400:lengh6p3)*180/pi))/500;
temperrh9p3 = ((exalh9p3(10,400:lengh9p3)')-...
  (mwspdh9p3(400:lengh9p3)*180/pi))/500;
wtsh42 = ones(size(temperrh4));
wtsh62 = ones(size(temperrh6));
wtsh92 = ones(size(temperrh9));
wtsh42(6*512-400:ceil(6.7*512)-400)=...
  wtsh42(6*512-400:ceil(6.7*512)-400)*10;
wtsh42(1*512-400:ceil(1.8*512)-400)=...
  wtsh42(1*512-400:ceil(1.8*512)-400)*10;
wtsh62(6*512-400:ceil(7.1*512)-400)=...
  wtsh62(6*512-400:ceil(7.1*512)-400)*10;
wtsh62(1*512-400:ceil(2.3*512)-400)=...
  wtsh62(1*512-400:ceil(2.3*512)-400)*10;
wtsh92(6*512-400:ceil(7.4*512)-400)=...
  wtsh92(6*512-400:ceil(7.4*512)-400)*10;
wtsh92(1*512-400:ceil(2.5*512)-400)=...
  wtsh92(1*512-400:ceil(2.5*512)-400)*10;

temperrh42 = temperrh4.*wtsh42;
temperrh62 = temperrh6.*wtsh62;
temperrh92 = temperrh9.*wtsh92;
err = 2*sum(temperrh4p1.*temperrh4p1)+...
      2*sum(temperrh6p1.*temperrh6p1)+...
```

```
      2*sum(temperrh9p1.*temperrh9p1)+...
      .1*sum(temperrh9p3.*temperrh9p3)+...
      4*16.1*sum(temperrh42.*temperrh42)+...
      9.52*sum(temperrh62.*temperrh62)+...
      8*sum(temperrh92.*temperrh92)+...
      2*sum(temperrhr.*temperrhr)+...
      2*sum(temperrhrn.*temperrhrn)+...
      .1*sum(temperrh6p3.*temperrh6p3);

[x ovs]
err;
tau=1/(ovs(1)*ovs(21)*ovs(4)*ovs(7)*sqrt(.5*(ovs(15)-ovs(2)/
ovs(1)))+ovs(6) );
gc =[];
[err gc ovs(3) tau]
pause(0.1);
```

Appendix F-- elcost.c

```
#include "stdio.h"
#include "math.h"
#include "mex.h"

void eval_deriv(double *xn,
                double *ovs,
                double *xef,
                double *flag_alf,
                double *fn)
{
  if ( (*flag_alf < -0.5) || (*flag_alf > 0.5) )
/*     the     swash     plate     is     at     a     limit
*/
       *fn  =  -ovs[5]*(*xn)  +  ovs[6]*sqrt((ovs[14]-(*xn))/
2.)*(*xef);
  else
/*    the    swash    plate    is    not    at    a    limit
*/
    *fn = -ovs[5]*(*xn)/(1.+ovs[4]) +
      ovs[6]*sqrt((ovs[14]-(*xn))/2.)*(*xef)/(1.+ovs[4]);
}

void rk4(double *xn,
         double *ovs,
         double *xef,
         double *flag_alf,
         double *dt,
         double *newxn)
{
  double k1,k2,k3,k4;
  double y2,y3,y4;
  double dt_div_2;

  dt_div_2 = (*dt)/2.;
  eval_deriv(xn,ovs,xef,flag_alf,&k1);
  y2 = k1*dt_div_2 + *xn;

  eval_deriv(&y2,ovs,xef,flag_alf,&k2);
  y3 = k2*dt_div_2 + *xn;

  eval_deriv(&y3,ovs,xef,flag_alf,&k3);
  y4 = k3*(*dt) + *xn;

  eval_deriv(&y4,ovs,xef,flag_alf,&k4);
  *newxn = *xn + (*dt/6.)*(k1+2.*(k2+k3)+k4);
```

```
}

void checklim(double *xin,
              double *xlimlo,
              double *xlimhi,
              double *xflag,
              double *xout)
{
    if (*xin >= *xlimhi)
    {
      *xout = *xlimhi;
      *xflag = 1;
    }
    else if (*xin <= -(*xlimlo))
    {
      *xout = -(*xlimlo);
      *xflag = -1;
    }
    else
    {
      *xout = *xin;
      *xflag = 0;
    }
}

void cost(double *delt,
          double *tfinal,
          double *ic,
          double *ovs,
          double *curr,
          double *xall)
{
  #define NSTATES 1

  double xn, newxn;
  double fn;
  double time;
  double alf;
  double calf,salf;
  double alfarm;
  double calfarm,salfarm;
  double xsp, xsptemp, sqrtxsptemp;
  double fsol;
  double flag_alf, flag_xsp, flag_xn, flag_xf;
  double xef;
  double thtd, presd;
```

```
  double curdz;

  double xsp_lim_lo, xsp_lim_hi;
  double alf_lim_lo, alf_lim_hi;
  double prs_lim_lo, prs_lim_hi;
  double xef_lim_lo, xef_lim_hi;

  double C1, C2, C3, C4;

  double rt1, rt2, rlim;


  int i,j;

/*   initialize   the   state   vector   and   other   stuff
*/
  xn = *ic;
  time = 0.0;
  j = 0;
  flag_alf = 0.0;
  flag_xsp = 0.0;
  flag_xn  = 0.0;
  flag_xf  = 0.0;
  xef = 0.0;

/* assign the limits, used in this subroutine, to their val-
ues
          obtained   from   the   optimization   parame-
ters                        */

xsp_lim_lo =  ovs[11];
xsp_lim_hi =  ovs[11];
alf_lim_lo =  ovs[21];
alf_lim_hi =  ovs[10];
prs_lim_lo =  ovs[12];
prs_lim_hi =  ovs[12];
xef_lim_lo =  ovs[22];
xef_lim_hi =  ovs[13];

/*              start              the              simula-
tion                                              */
  while (time < *tfinal)
  {
    if (curr[j+1] >= 0.)
    {
      C1 = ovs[16];
```

```
      C2 = ovs[17];
      C3 = ovs[18];
      C4 = 0.0;
    }
    else
    {
      C1 = ovs[23];
      C2 = ovs[24];
      C3 = ovs[25];
      C4 = ovs[26];
    }

/*      implement      deadzone      on      the      cur-
rent                                  */
    if (curr[j+1] < -ovs[19])
      curdz = curr[j+1] + ovs[19];
    else if (curr[j+1] < ovs[19])
      curdz = 0.0;
    else
      curdz = curr[j+1] - ovs[19];

/* The constant form works ok, cubic is allowed to match
small
                                solenoid          nonlineari-
ties                                          */
    fsol = C1*curdz + C2*pow(curdz,2) + C3*pow(curdz,3) +
      C4*pow(curdz,4);

/*      calculate      the      swash      plate      angle
*/
    alf = atan(ovs[0]*xn-ovs[1]);
/*    limit    the    swash    plate    angle    if    needed
*/
    checklim(&alf,&alf_lim_lo,&alf_lim_hi,&flag_alf,&alf);
    if (flag_alf > 0.5 & flag_xn < 0.5)
      prs_lim_hi = 0.999*xn;
    if (flag_alf < -0.5 & flag_xn > -0.5)
      prs_lim_lo = 0.999*fabs(xn);


/*     calculate     some     commonly     needed     stuff
*/
    calf = cos(alf);
    salf = sin(alf);

    alfarm = asin(ovs[20]*tan(alf));
```

```
    calfarm = cos(alfarm);
    salfarm = sin(alfarm);
/*         calculate         the         spool         posi-
tion                                    */
    xsptemp = ovs[2] - pow(ovs[3]*fsol,2);
    sqrtxsptemp = sqrt(xsptemp);

    xsp = (-ovs[2]*ovs[3]*salfarm*calfarm +
           pow(ovs[3],2)*fsol*sqrtxsptemp)/
               (ovs[2]*pow(calfarm,2)-pow(ovs[3]*fsol,2)) +
ovs[15];

    rt1 = (-ovs[2]*ovs[3]*salfarm*calfarm +
           pow(ovs[3],2)*fsol*sqrtxsptemp)/
          (ovs[2]*pow(calfarm,2)-pow(ovs[3]*fsol,2));
    rt2 = (-ovs[2]*ovs[3]*salfarm*calfarm -
           pow(ovs[3],2)*fsol*sqrtxsptemp)/
          (ovs[2]*pow(calfarm,2)-pow(ovs[3]*fsol,2));
     rlim = -ovs[2]*ovs[3]*salfarm*calfarm/(ovs[2]*pow(cal-
farm,2));
/*   limit   the   spool   valve   position   if   needed
*/
    checklim(&xsp,&xsp_lim_lo,&xsp_lim_hi,&flag_xsp,&xsp);
/*       limit       the       pressure       if       needed
*/
    checklim(&xn,&prs_lim_lo,&prs_lim_hi,&flag_xn,&xn);
/* calculate the effective spool position based on the ori-
fice      */
    checklim(&xsp,&xef_lim_lo,&xef_lim_hi,&flag_xf,&xef);
/*             calculate             winch             speed
*/
   eval_deriv(&xn,ovs,&xef,&flag_alf,&presd);
   thtd = ovs[7]*alf - ovs[8] - ovs[9]*calf*presd;
/*     save     off     the     current     step     informa-
tion                             */
    *(xall+j*10) = time;
    *(xall+1+j*10) = alf*57.296;
    *(xall+2+j*10) = xsp;
    *(xall+3+j*10) = xn;
    *(xall+4+j*10) = xef;
    *(xall+5+j*10) = rt1;
    *(xall+6+j*10) = sqrtxsptemp;
    *(xall+7+j*10) = fsol;
    *(xall+8+j*10) = rt2;
    *(xall+9+j*10) = thtd*57.296;
```

```
/* integrate pressure using RK4 to get the n+1 informa-
tion          */
    rk4(&xn,ovs,&xef,&flag_alf,delt,&newxn);
    xn = newxn;
    j = j+1;
    time = time + *delt;
  }
}

void mexFunction( int nlhs, mxArray *plhs[],
                  int nrhs, const mxArray *prhs[])
{

  double *deltime;
  double *tfin;
  double *ic;
  double *ovs;
  double *curr;
  double *exal;

/*         create         matrices         for         return
*/
  plhs[0] = mxCreateDoubleMatrix(10,15361,mxREAL);

  deltime = mxGetPr(prhs[0]);
  tfin    = mxGetPr(prhs[1]);
  ic      = mxGetPr(prhs[2]);
  ovs     = mxGetPr(prhs[3]);
  curr    = mxGetPr(prhs[4]);


  exal    = mxGetPr(plhs[0]);

  cost(deltime,tfin,ic,ovs,curr,exal);
}
```