AC 2010-1436: REMOTE USE OF A LINEAR AXIS RAPID DEVELOPMENT SYSTEM

Lie Tang, Missouri University of Science and Technology

Robert Landers, Missouri University of Science and Technology

Remote Use of a Linear Axis Rapid Development System

Abstract

A Linear Axis Rapid Development System (RDS) was developed and tested in a previous research study. The Linear Axis RDS, which is based on Matlab Simulink, provides the student with a tool to explore all phases of controller development (i.e., simulation, emulation, and implementation) after the theoretical work is complete. However, the Linear Axis RDS did not provide the students with the ability to explore the linear axis dynamic model. In this paper, the previously developed Linear Axis RDS is augmented such that the student can model the linear axis dynamics and analyze their model. The student encodes their linear axis dynamic model as a subsystem in Matlab Simulink. The subsystem inputs and outputs, along with their engineering units, are carefully specified. The student then utilizes the Linear Axis RDS to analyze the dynamic model. The Linear Axis RDS has two modes: simulation and implementation. In the simulation mode the student simulates a linear axis system dynamic response for a variety of command voltages signals. The student can specify the magnitude and frequency of square, triangle, and sinusoidal command voltage signals, or they can create their own command voltage signal. In this mode the student can check their linear dynamic model for obvious errors (e.g., it is unstable). In the implementation mode, command voltage signals are simultaneously sent to the Matlab Simulink model and the real linear axis system. Simulation and experimental data are gathered and compared. This paper describes the modifications made to the Linear Axis RDS such that students can perform dynamic modeling and analysis of the linear axis. The results of an initial usability study are presented and analyzed. The Linear Axis RDS is then implemented remotely in an actuators course being taken by students in a Mechatronics program in the university ESIGELEC in Rouen, France.

Introduction

A Rapid Development System (RDS) for a Linear Axis was developed in [1]. A RDS is a software environment that allows students to rapidly integrate their controller and analyze it via simulation, emulation, and implementation. In the simulation mode the student simulates a linear axis system that includes their controller and detailed models of the interface hardware and linear axis. In the emulation mode, the simulation is performed on the computer hardware that will implement the controller. In this mode the student can ensure their algorithm will run in real time (i.e., the algorithm's execution time is less than the sample period). In the implementation mode, the controller is deployed on the hardware system and experimental data is gathered. The Linear Axis RDS aided the students in the implementation of their controllers, even if the student had very little knowledge of control system hardware. The Linear Axis also allowed the students to focus on controller analysis.

This paper adds a functionality to the Linear Axis RDS, known as the Axis Modeling function. This function allows students to develop and analyze a model of the Linear Axis. The Axis Modeling function was implemented in a short course on actuators given to eight students at ESIGELEC in Rouen, France. The students utilized the Axis Modeling function in the Linear

Axis RDS remotely. An implementation of this functionality is provided and the results are discussed in detail.

Linear Axis Rapid Development System

A Linear Axis RDS has been developed in [1] for linear axis controller design. This paper describes the addition of linear axis modeling capabilities to the Linear Axis RDS and the application to to the x–axis of the mini–CNC machine (Figure 1). A host computer, which has a non real–time operating system, runs the Linear Axis RDS Graphical User Interface (GUI), as well as the controller and models in simulation mode. A target computer, which has a real–time operating system, runs the controller and models in emulation and implementation modes. The target computer has a National Instruments (NI) 6711 digital to analog (D/A) output board and a NI 6602 counter–time (C/T) board. The target computer, which utilizes the xPC real–time operating system, sends voltage commands through the output board and receives encoder measurements through the C/T board, both at a sampling rate of 1 kHz. The output voltage range is ±10 V and is amplified before reaching the DC motor by a factor of 2.4. The motor has a gear ratio of 20/1 from the motor shaft to the output shaft. The output shaft is connected to a lead screw with a pitch of 5.9 mm/rev, which translates the linear axis. The encoder, which has a resolution of 500 counts/rev, measures the motor shaft displacement. The C/T board is run in quadrature mode for an effective encoder resolution of 2000 counts/rev.

The Linear Axis RDS has two functions: Axis Controller and Axis Modeling. When the Axis Controller mode is utilized, the Linear Axis RDS allows the student to operate the controller in three different modes: simulation, emulation, and implementation. These modes are reviewed below. Then, the new Axis Modeling function is decribed.

The Axis Controller function has a Simulation mode that allows the student to run the Simulink model of the closed–loop system on a host computer. This is beneficial because the student does not need to be connected to the physical linear axis. The Simulink model used for simulation mode, shown in Figure 2, contains four subsystems: Reference Generator, Controller, Computer–System Interface, and Linear Axis Model. The Reference Generator subsystem contains code that generates the desired linear axis position, called the reference position, which is sent to the controller. The Controller subsystem contains the controller to be tested. The Controller subsystem receives the reference and measured positions from the Reference Generator and Linear Axis Model subsystems, respectively, and sends the command voltage to the Computer–System Interface subsystem. The Computer–System Interface subsystem, shown in Figure 3, contains the simulated quantization and saturation affects of the D/A output board and the simulated quantization affect of the encoder. The Linear Axis Model subsystem, shown in Figure 4, contains the linear axis dynamic model. The model calculates the linear axis position given the command voltage from the Computer–System Interface subsystem. The reference position, measured position, and commanded voltage are recorded for controller performance analysis.

The Axis Controller function has a Emulation mode that allows the student to run their controller on the target computer in real–time without moving the linear axis. The linear axis is simulated on the target computer. Emulation allows the student to determine if the target computer is able to perform all computations and perform send and receive tasks within the specified sample period. The Simulink model used for the Emulation mode contains the same four blocks as the Simulink model used for Simulation mode. The difference between the Simulink models lies in the Computer–System Interface subsystem. The output voltage sent to the physical linear axis is zero and the position measurements from the encoder are disregarded as shown in Figure 5. These send and receive blocks in the Computer–System Interface subsystem are used to determine the time required to perform the communication tasks on the target computer. Since the linear axis is only one component of the mini–CNC, zero voltage signals are also sent to the other components to ensure they do not move. The same controller used in Simulation mode is also used in Emulation mode.

The Axis Controller function has an Implementation mode that allows the student to operate the linear axis system using their controller. The Simulink model used for Implementation mode, shown in Figure 6, contains three subsystems: Reference Generator, Controller, and Computer–System Interface. The difference between the implementation and emulation modes is 1) there are no simulated affects in the Computer–System Interface subsystem and 2) there is no model to simulate the linear axis. This is because these affects are present in the computer interface hardware and physical linear axis and, thus, do not need to be simulated. Within the Computer–System Interface subsystem the command voltage from the controller is sent to the D/A output board and the encoder measurements are received from the C/T board as shown in Figure 7. The encoder measurements are used to calculate the measured linear axis position, which is sent to the controller. The same controller used in Simulation and Emulation modes is also used in the Implementation mode.

The new function that has been added to the Linear Axis RDS is the Axis Modeling function. When used for axis modeling, the Linear Axis RDS can be used to collect system dynamic response data for modeling, simulate the linear axis dynamic response, and compare the model dynamic response to measured data. Three modes are provided for the student to perform axis modeling: Axis Test, Simulation, and Model Validation. The Axis Test mode allows the student to collect the linear axis dynamic response data for different command voltage signals. This data can be used to model the axis. The Axis Test mode Simulink model is shown in Figure 8. The command voltage is generated in the Voltage Signal subsystem and sent to the physical system via the IO Interface subsystem. The command voltage and measured position are recorded for further analysis. The Simulation mode allows the student to run the Simulink model on the host computer with a sine wave or a user defined command voltage signal. The Simulation mode Simulink model is shown in Figure 9. The command voltage generated in the Voltage Signal subsystem is sent to the Component model via the Computer–System Interface subsystem, which simulates the effects of physical input/output system such as quantization and saturation. The command voltage and measured position are recorded. The Model Validation mode allows the student to compare the simulated dynamic response with measured data for the same command voltage signal, which is beneficial because the student can observe the difference directly. The Model Validation Simulink model is shown in Figure 10. The command voltage is generated at the Voltage Signal subsystem and then sent to both the physical system and system model simultaneously. The command voltage, model response, and physical system response are recorded for further analysis.

Linear Axis RDS Modeling Functionalities

The Linear Axis RDS axis modeling GUI is used to build and test the Simulink model containing the axis model developed by the student. As shown in Figure 11, the GUI can be separated into the following sections: Operation, Simulink Model Builder, Voltage Signal, Animation, Parameters, Execution, and Results. The Operation section includes two pushbuttons: Main Menu and Jog, which direct the user to the main menu and jog GUIs. The Simulink Model Builder section is used to build the Simulink model (i.e., integrate the student's code and selections into one Simulink file). It consists of two radio buttons, Discrete and Continuous, which determines the simulation solver type, and three categories: Mode, Voltage Signal, and Model. The Mode category has three options: Simulation, Axis Test, and Model Validation, which are described above. The Voltage Signal category has two options: Sine wave and User Defined. The student can insert their own voltage signal by choosing the latter option. The Model category has two options: Default and User Defined. The Default model provides a benchmark with which the student can compare the performance of their own model. Upon finishing these settings, the Build pushbutton is pressed to build the Simulink model. Once the model is built, it can be viewed by pressing the View Model button. Voltage amplitude and frequency can be set in the Voltage Signal section if the Voltage Signal category option is sine wave. An animation of the linear axis is shown at the center of GUI. The animation provides quasi real time motion of the linear axis. The simulation final time and sample period can be designated in the Parameter section. Once these settings are made properly, the student can run the model using the pushbutton Simulate/Run, depending on the Mode option. If Axis Test or Model Validation is selected, the model is downloaded to the target processor by pressing Load Model. In the Results section, three options are provided for results processing: Plot, Save, and Email. The results can be plotted or saved using Plot, or Save. The results can be directly plotted on the screen using the Plot option or saved in a file using the Save option. The saved results can be emailed to the designated recipients using the Email option.

Results

The Axis Modeling function of the Linear Axis RDS was utilized by eight students in a short course on actuators at ESIGELEC in Rouen, France. The students are in the fifth year of an undergraduate/masters Mechatronics program in electrical engineering. The students were divided into four groups and each group conducted a project on modeling and analyzing a linear axis. A project is described using results from one of the groups. The project consisted of six tasks.

Task 1: Symbolically determine a set of first order differential equations describing the linear axis dynamics.

The leadscrew gear gain is

$$K_l \equiv \frac{T_i}{T_l} = \frac{\omega_l}{\omega_m} = \frac{\dot{\omega}_l}{\dot{\omega}_m} = 0.1659 \tag{1}$$

where T_i is the torque drained from the motor by the leadscrew (N·m), T_l is the torque supplied to the leadscrew by the motor (N·m), ω_l is the leadscrew angular velocity (rad/s), and ω_m is the motor angular velocity (rad/s). The leadscrew pitch is

$$p \equiv \frac{T_a}{f_a} = \frac{v}{\omega_l} = \frac{\dot{v}}{\dot{\omega}_l} = 2.022 \times 10^{-4}\,\frac{\text{m}}{\text{rad}} \tag{2}$$

where T_a is the torque drained from the leadscrew by the linear axis (N·m), f_a is the force supplied to linear axis by the leadscrew (N), and v is the linear axis velocity (m/s). Applying KVL to the electrical circuit

$$L_a \dot{I}_a(t) + R_a I_a(t) + K_v \omega_m(t) = K_a e_c(t) \tag{3}$$

where L_a is the motor inductance ($2.63 \cdot 10^{-3}$ H), I_a is the armature current (A), R_a is the motor electrical resistance (2.49 Ω), K_v is the motor voltage constant ($4.58 \cdot 10^{-2}$ V/(rad/s)), K_a is the amplifier gain (2.231), and e_c is the command voltage (V). Summing the torques applied to the motor

$$J_m \dot{\omega}_m(t) = -B_m \omega_m(t) - T_i(t) + T_m(t) \tag{4}$$

where J_m is the motor mass moment of inertia ($7.1 \cdot 10^{-6}$ kg·m^2), B_m is the motor viscous friction coefficient ($3.5 \cdot 10^{-6}$ N·m/(rad/s)), and $T_m = K_t I_a$ is the motor electrical torque (N·m), and K_t is the motor torque constant ($4.58 \cdot 10^{-2}$ N·m/A). Summing the torques applied to the leadscrew

$$J_l \dot{\omega}_l(t) = -B_l \omega_l(t) - T_a(t) + T_l(t) \tag{5}$$

where J_l is the leadscrew mass moment of inertia (kg·m^2) and B_l is the leadscrew viscous friction coefficient (N·m/(rad/s)). Summing the forces applied to the linear axis

$$m\dot{v}(t) = -b_a v(t) + f_a(t) \tag{6}$$

where m is the linear axis mass (kg) and b_a is the linear axis viscous friction coefficient (N/(m/s)). Multiplying equation (6) by p and combining the resulting equation with equation (5)

$$J_l \dot{\omega}_l(t) = -B_l \omega_l(t) - pm\dot{v}(t) - pb_a v(t) + T_l(t) \tag{7}$$

Multiplying equation (7) by K_l and combining the resulting equation with equation (4)

$$J_m \dot{\omega}_m(t) = -B_m \omega_m(t) - K_l J_l \dot{\omega}_l(t) - K_l B_l \omega_l(t) - pK_l m\dot{v}(t) - pK_l b_a v(t) + T_m(t) \tag{8}$$

Multiplying equation (8) by pK_l, substituting for T_m, and rearranging

$$\dot{v}(t) = -\frac{B_{eff}}{J_{eff}} v(t) + \frac{pK_l K_t}{J_{eff}} I_a(t) \tag{9}$$

where $J_{eff} = J_m + K_l^2 J_l + p^2 K_l^2 m$ and $B_{eff} = B_m + K_l^2 B_l + p^2 K_l^2 b_a$. Combining equations (1)–(3) and rearranging

$$\dot{I}_a(t) = -\frac{R_a}{L_a} I_a(t) - \frac{K_v}{L_a K_l p} v(t) + \frac{K_a}{L_a} e_c(t) \tag{10}$$

From kinematics

$$\dot{x}(t) = v(t) \tag{11}$$

where $x(t)$ is the linear axis position (m). Equations (9)–(11) are a set of first order differential equations describing the linear axis dynamics. The leadscrew has 20 threads/inch. The parameters J_l, K_l, m, and b_a are unknown and are assumed to be zero due to the substantial system gear gains.

Task 2: Determine a transfer function relating the axis position to the command voltage. Compute the poles. For each real pole, compute its time constant, rise time, and 2% settling time. For each complex conjugate pair of poles, compute their natural frequency, damping ratio, rise time, 2% settling time, peak time, and percent overshoot.

Taking the Laplace transform of equations (9)–(11) with zero initial conditions, combining the resulting equations, and rearranging, the transfer function relating linear axis position to command voltage is

$$\frac{X(s)}{E_c(s)} = -\frac{pK_l K_t K_a}{J_{eff} L_a s^3 + \left(J_{eff} R_a + L_a B_{eff}\right) s^2 + \left(R_a B_{eff} + K_t K_v\right) s} \tag{12}$$

Inserting numerical values, the poles are located at 0, –139.7, and –807.6. The pole located at 0 does not have a time constant or settling time associated with it. The pole located at –139.7 has a time constant of 7.159 ms and a 2% settling time of 28.64 ms. The pole located at –807.6 has a time constant of 1.238 ms and a 2% settling time of 4.957 ms.

Task 3: Create a Matlab Simulink simulation of the linear axis. Provide a screen shot of the Simulink block diagram in your report. Simulate the linear axis for the data file step.txt, which will be provided to you. The first column is time (s), the second column is command voltage (V), and third column is measured position (mm). Put the dynamic model in a subsystem called X Axis System.

A picture of the Simulink block diagram subsystem used to simulate the linear axis is shown in Figure 12 and the experimental results for a series of constant command voltages are shown in Figure 13. The dynamic model appears to fit the physical system very well.

Task 4: Copy the dynamic model subsystem and the corresponding Callback code to another Simulink file. Implement that file in the Linear Axis Rapid Development System (RDS) for three sinusoidal inputs. Run the Linear Axis RDS for sinusoidal input command voltages with magnitudes of 2 V and frequencies of 2 Hz, 5 Hz, and 10 Hz. Set the final time such that the command voltage completes five cycles.

Since the physical hardware is located at the Missouri University of Science and Technology (Missouri S&T) in Rolla, Missouri and the students are at ESIGELEC in Rouen, France, the equipment had to be use remotely. This was accomplished by using WebEx. Each group developed a complete Simulink model that included command voltage signals and plotting capabilities. The instructor started a session and the groups would email their linear axis dynamic model Simulink models to an assistant at Missouri S&T. Each group then utilized the Linear Axis RDS to simulate their model for the step and sinusoidal command voltage signals. Then, each group utilized the Linear Axis RDS to simulate their model and implement the physical linear axis for the same command voltage signals. The simulation and experimental data were simultaneously collected and subsequently emailed to the groups.

Task 5: For each of the three experiments in Task 4 make a figure with two subplots. The first subplot is the linear axis position versus time. The second subplot is the command voltage versus time. The first subplot should contain two lines: one for the simulation and one for the experimental data. For this subplot include a legend and clearly distinguish the lines.

The results for the sinusoidal command voltages with frequencies of 2, 5, and 10 Hz are shown in Figures 14, 15, and 16, respectively. It can be seen that the experimental position response tends to drift in the negative direction. This is probably due to Coulomb friction, a nonlinear constant friction that was not incorporated into the linear axis dynamic model. The Coulomb friction is probably larger in magnitude in direction in the positive direction, causing the linear axis is drift in the negative direction.

Task 6: Using the transfer function in Task 2, analytically determine the transfer function's frequency response. Using the data from the sinusoidal experiments, experimentally determine the frequency response for those specific frequencies. Make a figure with two subplots. The first subplot is the analytical and experimental magnitude frequency responses. The second subplot is the analytical and experimental phase frequency responses. Both subplots should contain a line for the analytical frequency response and distinct markers for the experimental frequency response. The subplots should include a legend.

Transforming equation (12) into the frequency domain

$$\frac{X(j\omega)}{E_c(j\omega)} = -\frac{b_0 a_2 \omega^2}{\left(a_2\omega^2\right)^2 + \left(a_1\omega - a_3\omega^3\right)^2} - \frac{b_0\left(a_1\omega - a_3\omega^3\right)}{\left(a_2\omega^2\right)^2 + \left(a_1\omega - a_3\omega^3\right)^2} j \tag{13}$$

where $b_0 = pK_l K_t K_a$, $a_1 = R_a B_{eff} + K_t K_v$, $a_2 = J_{eff} R_a + L_a B_{eff}$, and $a_3 = J_{eff} L_a$. The analytical and experimental frequency responses are compared in Figure 16. Although the experimental responses for the sinusoidal command voltages tended to drift, it can be seen that the magnitude ratio was the same for both the analytical and experimental data. Also, the analytical and experimental phases are very similar, with differences between 4 and 5 degrees.

Assessment

The students were given a questionnaire to complete following the project assignment. The questionnaires have not yet been returned. However, anecdotally the students expressed satisfaction with the course project and said that it helped them learn the material on sensors better.

Summary and Conclusions

An Axis Modeling functionality was added to a Linear Axis Rapid Development System (RDS). The student encodes their linear axis dynamic model as a subsystem in Matlab Simulink and the new functionality allows the student to simulated their model and then simultaneously simulate their model while implementing the physical system. The Linear Axis RDS was implemented remotely by four project groups in an actuators course in a Mechatronics program in the university ESIGELEC in Rouen, France. The Axis Modeling functionality greatly enhances the Linear Axis RDS. Anecdotally, the students felt the project based on this new functionality helped them to learn the course material.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant Number 0736731. The authors would like to acknowledge the assistance of Dr. Hong Sheng at the Missouri University of Science and Technology in developing the assessment questions and Dulce Alford in the Information Technology department at the Missouri University of Science and Technology in the remote utilization of the Linear Axis Rapid Development System.

References

[1] Fleming, M., Jain, V., Landers, R.G., Sheng, H., and Hall, R., 2009, "Implementation and Evaluation of a Linear Axis Rapid Development System," *ASEE Annual Conference and Exhibition*, Austin, Texas, June 14–17.

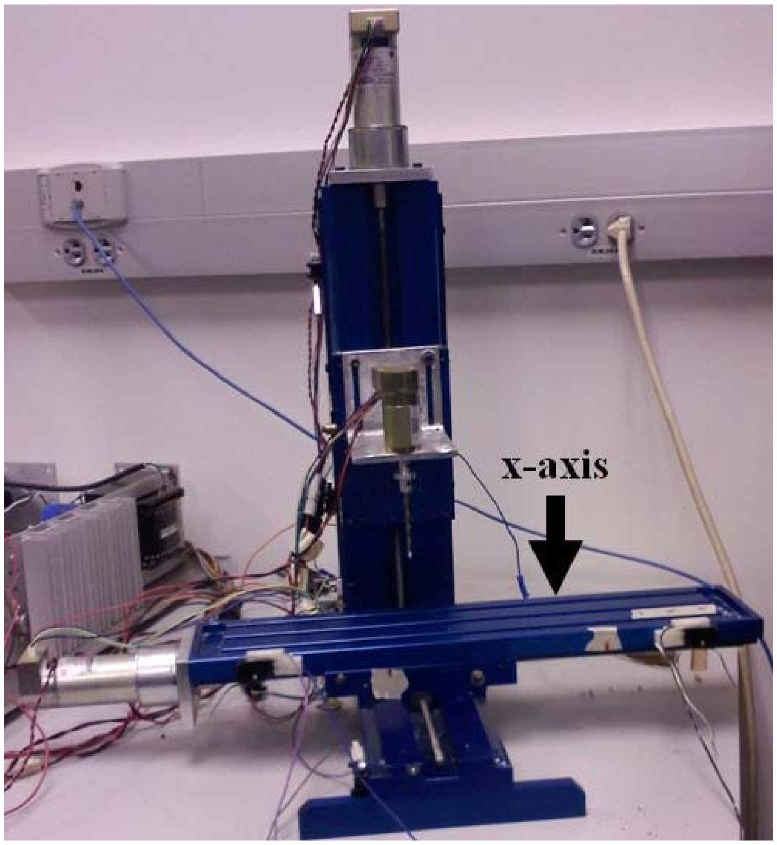


Figure 1: Mini–CNC System Containing Linear Axis Used in RDS.

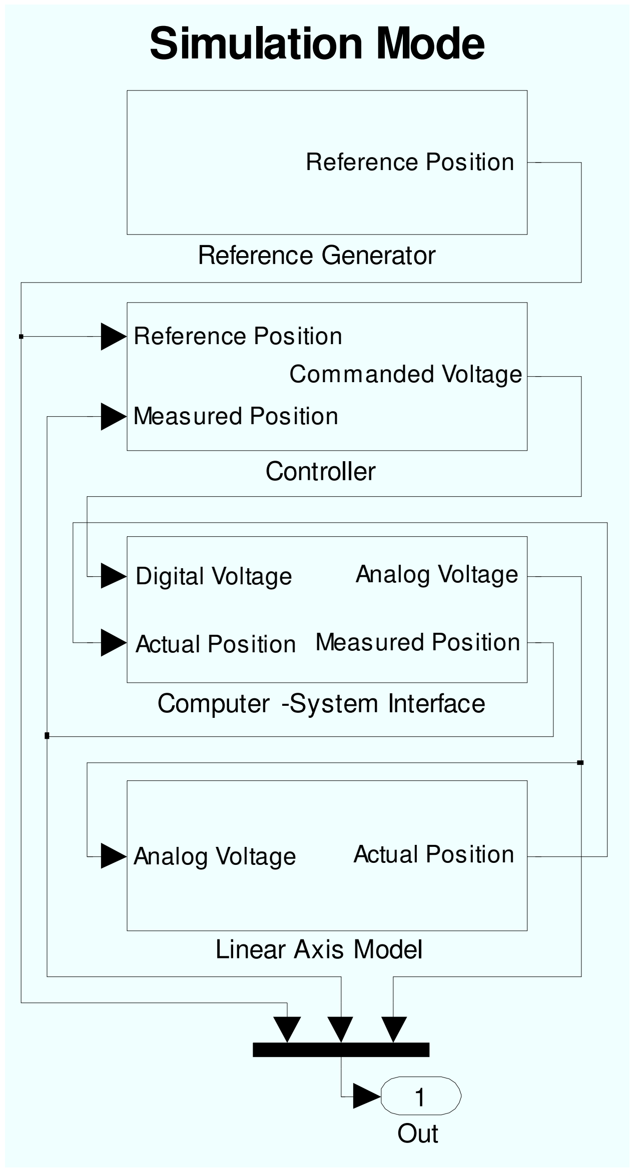


Figure 2: Simulink Model for Simulation Mode.

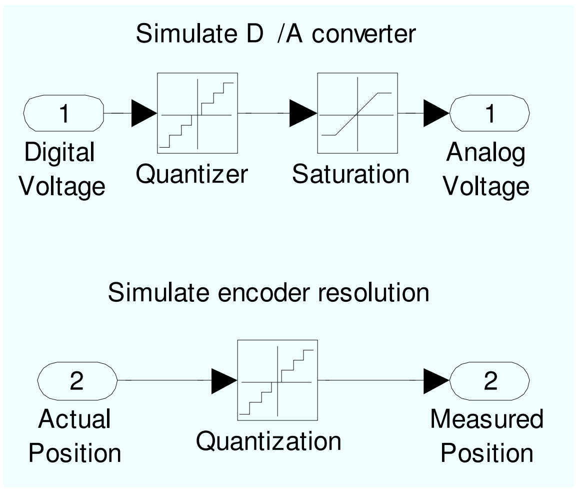


Figure 3: Contents of Computer–System Interface Subsystem for Simulation Mode.

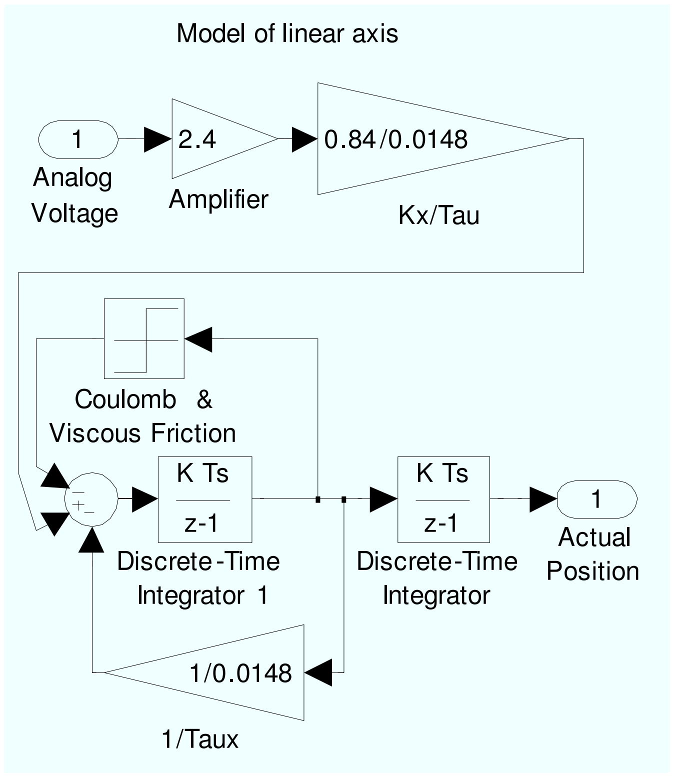


Figure 4: Linear Axis Model Contained in Linear Axis Model Subsystem.

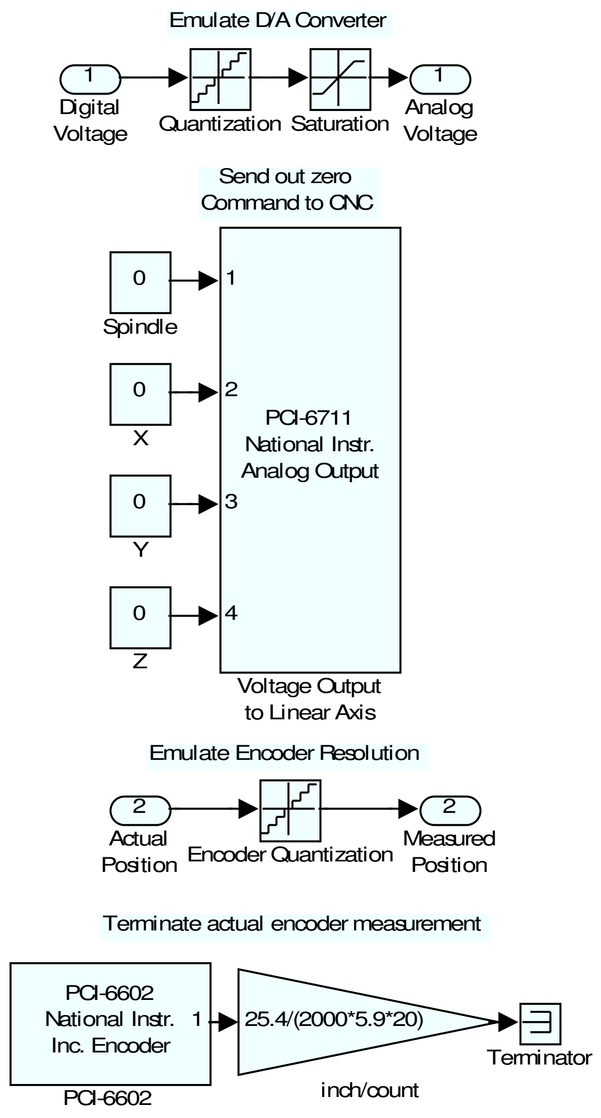

Emulate D/A Converter
1
Digital Voltage
Quantization
Saturation
1
Analog Voltage
Send out zero Command to CNC
0
Spindle
0
X
0
Y
0
Z
1
2
3
4
PCI-6711 National Instr. Analog Output
Voltage Output to Linear Axis
Emulate Encoder Resolution
2
Actual Position
Encoder Quantization
2
Measured Position
Terminate actual encoder measurement
PCI-6602 National Instr. Inc. Encoder
1
PCI-6602
25.4/(2000*5.9*20)
inch/count
Terminator

Figure 5: Contents of Computer–System Interface Subsystem for Emulation Mode.

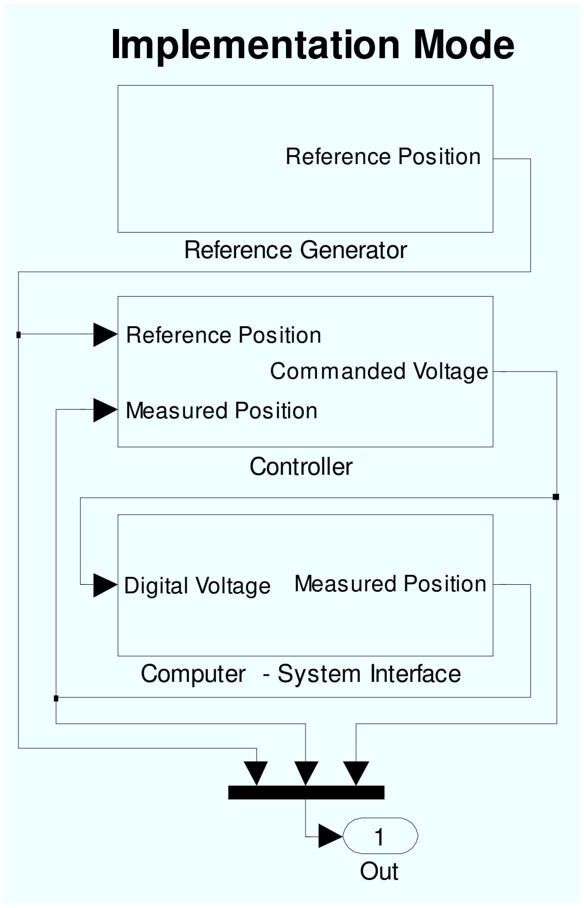


Figure 6: Simulink Model for Implementation Mode.

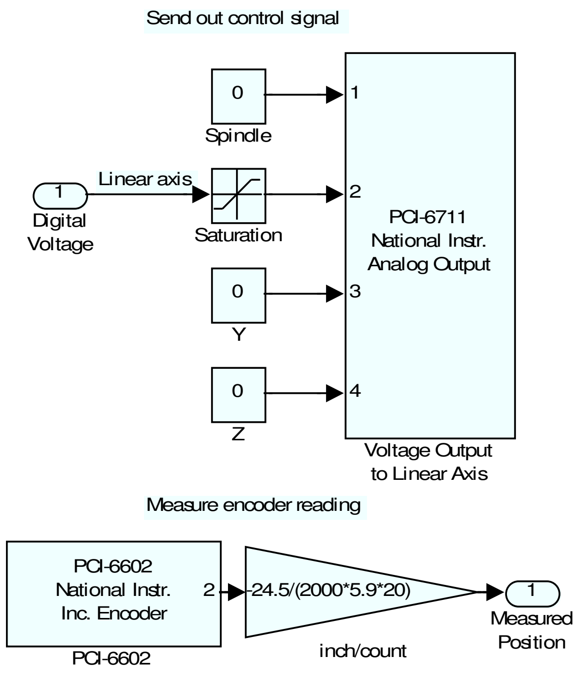


Figure 7: Contents of Computer–System Interface for Implementation Mode.

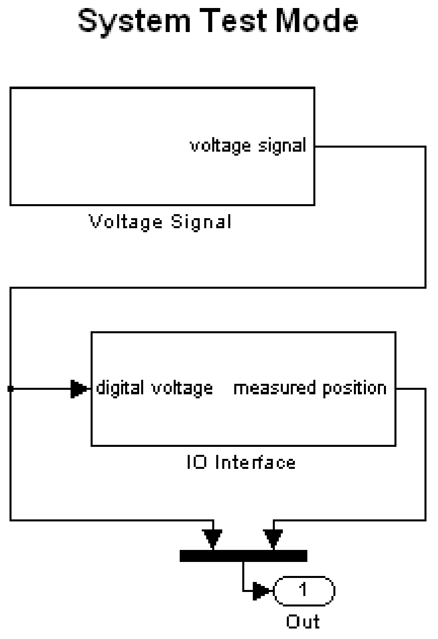


Figure 8: Axis modeling axis test mode Simulink model.

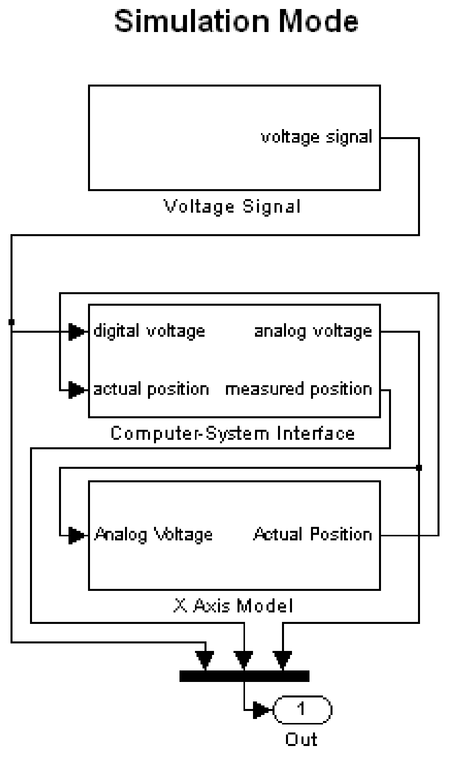


Figure 9: Axis modeling simulation mode Simulink model.

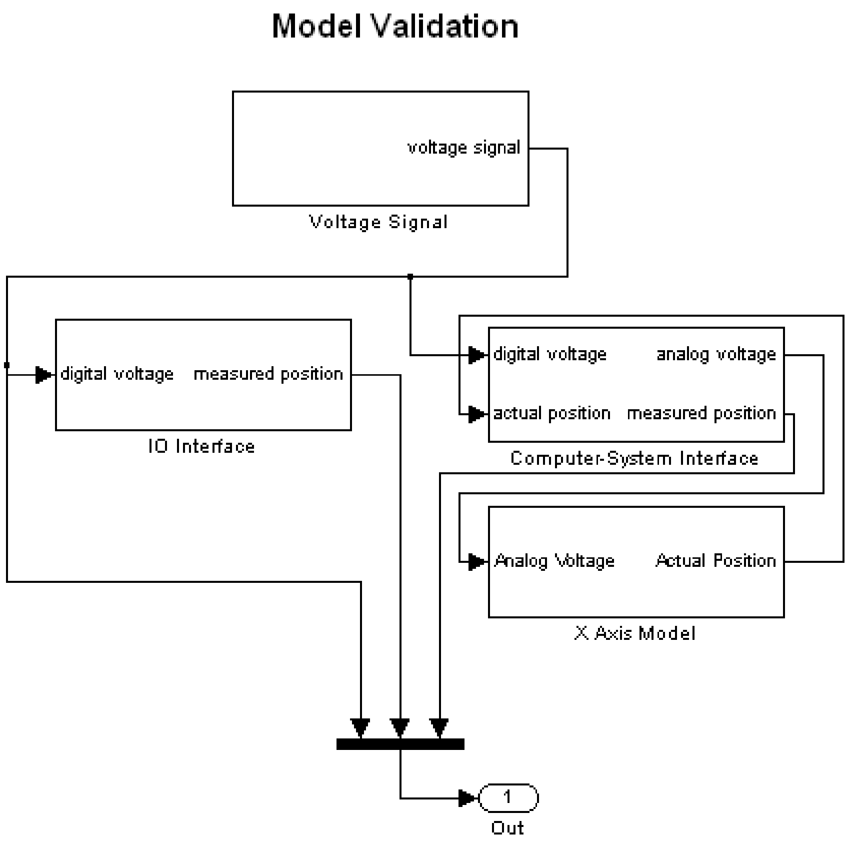
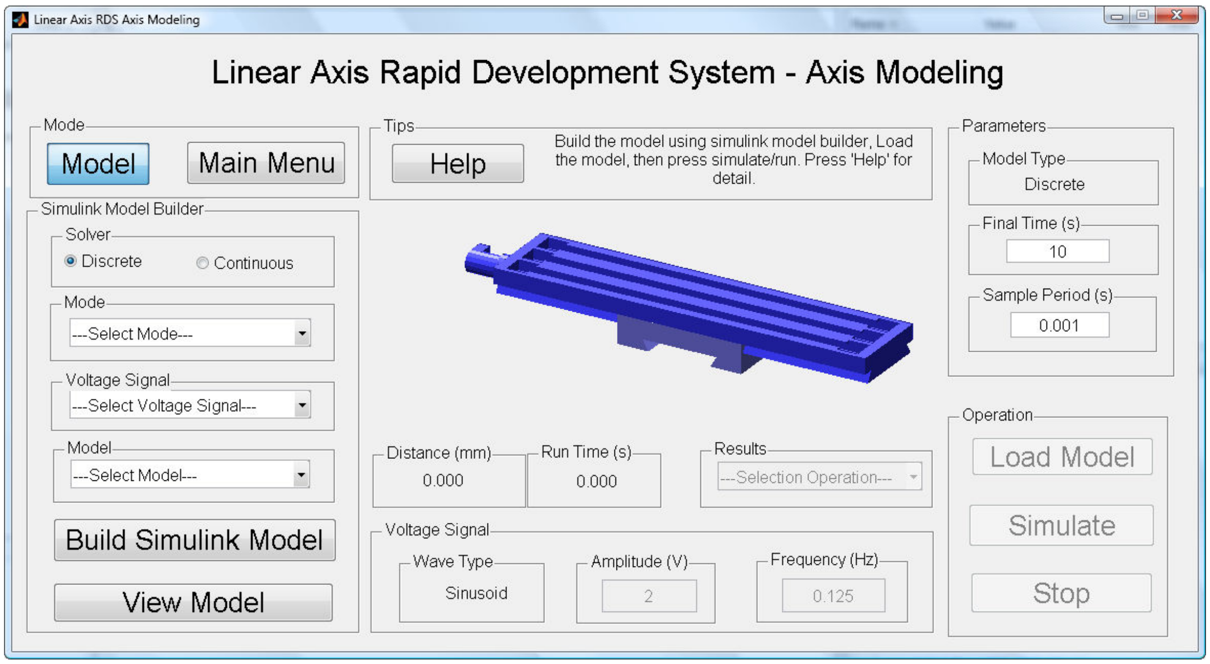


Figure 10: Axis modeling model validation mode Simulink model.



Build the model using simulink model builder, Load the model, then press simulate/run. Press 'Help' for detail.

Model Type: Discrete
Final Time (s): 10
Sample Period (s): 0.001
Solver: [✓ Discrete] Continuous
Mode: ---Select Mode---
Voltage Signal: ---Select Voltage Signal---
Model: ---Select Model---
Distance (mm): 0.000
Run Time (s): 0.000
Results: ---Selection Operation---
Voltage Signal
Wave Type: Sinusoid
Amplitude (V): 2
Frequency (Hz): 0.125

Figure 11: Linear Axis RDS axis modeling GUI.

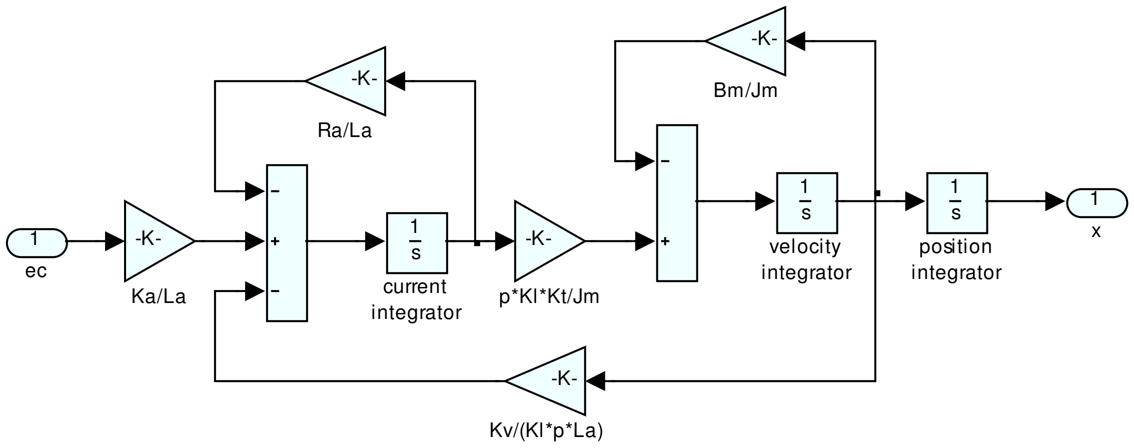


Figure 12: Linear Axis Simulink subsystem.

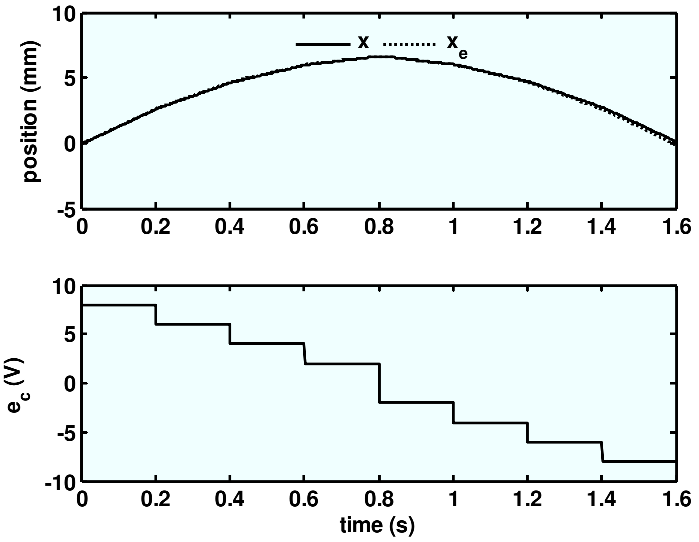


Figure 13: Experimental and simulation results for step command voltage.

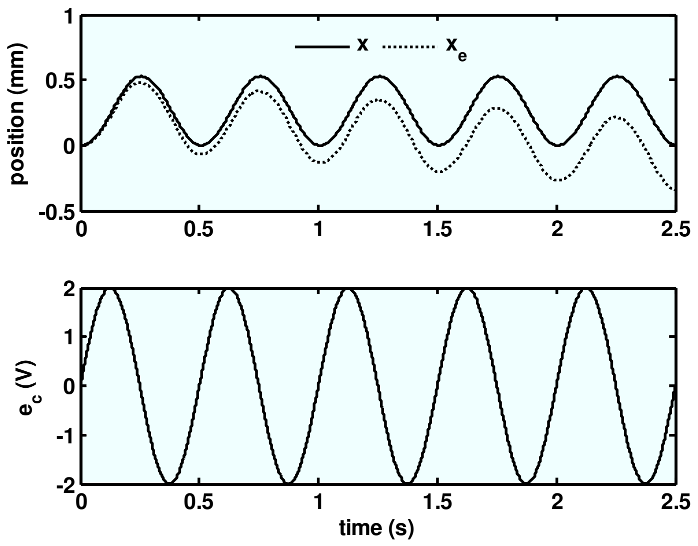


Figure 13: Experimental and simulation results for sinusoidal command voltage with frequency 2 Hz.

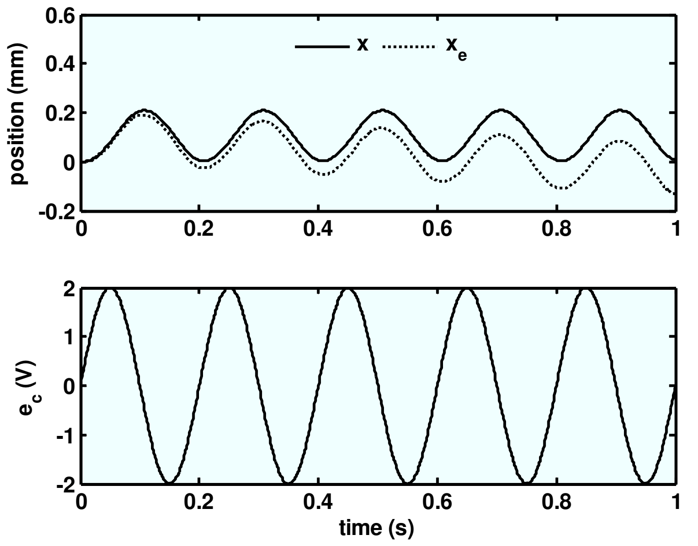


Figure 14: Experimental and simulation results for sinusoidal command voltage with frequency 5 Hz.

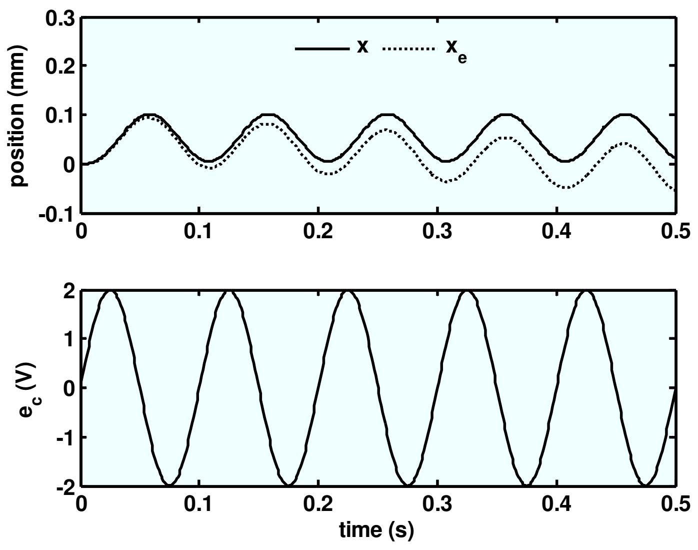


Figure 15: Experimental and simulation results for sinusoidal command voltage with frequency 10 Hz.

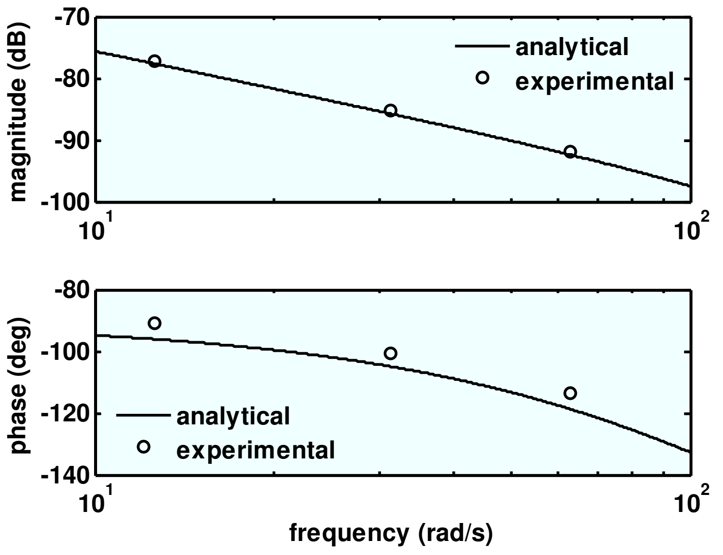


Figure 16: Analytical frequency response compared to experimental sinusoidal results.