QUANSER
INNOVATE. EDUCATE.

STUDENT WORKBOOK

SRV02 Base Unit Experiment For Matlab®/Simulink® Users

Standardized for ABET Evaluation Criteria

Developed by:
Jacob Apkarian, Ph.D., Quanser
Michel Lévis, M.A.Sc., Quanser
Hakan Gurocak, Ph.D., Washington State University

SRV02 educational solutions are powered by:

MATLAB SIMULINK

Course material complies with:

ABET

CAPTIVATE. MOTIVATE. GRADUATE.

Quanser Inc.
119 Spy Court
Markham, Ontario
L3R 5H6
Canada
info@quanser.com
Phone: 1-905-940-3575
Fax: 1-905-940-3576

Printed in Markham, Ontario.

For more information on the solutions Quanser Inc. offers, please visit the web site at:
http://www.quanser.com

ACKNOWLEDGEMENTS

Quanser, Inc. would like to thank Dr. Hakan Gurocak, Washington State University Vancouver, USA, for rewriting this manual to include embedded outcomes assessment.

CONTENTS

Preface **v**

1 SRV02 Modeling **1**

1.1 Background 2

1.1.1 Modeling Using First-Principles 2

1.1.2 Modeling Using Experiments 5

1.2 Pre-Lab Questions 8

1.3 Lab Experiments 9

1.3.1 Frequency Response Experiment 9

1.3.2 Bump Test Experiment 12

1.3.3 Model Validation Experiment 13

1.3.4 Results 14

1.4 System Requirements 15

1.4.1 Overview of Files 15

1.4.2 Configuring the SRV02 and the Lab Files 15

1.5 Lab Report 17

1.5.1 Template for Content 17

1.5.2 Tips for Report Format 18

2 SRV02 Position Control **19**

2.1 Background 20

2.1.1 Desired Position Control Response 20

2.1.2 PV Controller Design 23

2.1.3 PIV Controller 24

2.2 Pre-Lab Questions 26

2.3 Lab Experiments 27

2.3.1 Step Response Using PV Controller 27

2.3.2 Ramp Response Using PV Controller 30

2.3.3 Ramp Response with No Steady-State Error 32

2.3.4 Results 33

2.4 System Requirements 34

2.4.1 Overview of Files 34

2.4.2 Setup for Position Control Simulations 34

2.4.3 Setup for Position Control Implementation 35

2.5 Lab Report 36

2.5.1 Template for Content (Step Response Experiment) 37

2.5.2 Template for Content (Ramp Response with PV) 38

2.5.3 Template for Content (Ramp Response with No Steady-State Error) 39

2.5.4 Tips for Report Format 40

3 SRV02 Speed Control **41**

3.1 Background 42

3.1.1 Desired Response 42

3.1.2 PI Control Design 43

3.1.3 Lead Control Design 45

3.1.4 Sensor Noise 50

3.2 Pre-Lab Questions 51

3.3 Lab Experiments 52

3.3.1 Step Response with PI Control 52

3.3.2 Step Response with LEAD Control 55

3.3.3 Results 56

3.4 System Requirements 58

3.4.1 Overview of Files 58

3.4.2 Setup for Speed Control Simulation 58

3.4.3 Setup for Speed Control Implementation 59

3.5 Lab Report 60

3.5.1 Template for Content (PI Control Experiments) 61

3.5.2 Template for Content (Lead Control Experiments) 62

3.5.3 Tips for Report Format 63

A SRV02 QUARC Integration 65

A.1 Applying Voltage to SRV02 Motor 66

A.1.1 Making the Simulink Model 66

A.1.2 Compiling the Model 68

A.1.3 Running QUARC Code 69

A.2 Reading Position using Potentiometer 71

A.2.1 Reading the Potentiometer Voltage 71

A.2.2 Measuring Position 72

A.3 Measuring Speed using Tachometer 75

A.4 Measuring Position using Encoder 77

A.5 Saving Data 79

PREFACE

Every laboratory chapter in this manual is organized into four sections.

Background section provides all the necessary theoretical background for the experiments. You should read this section first to prepare for the Pre-Lab questions and for the actual lab experiments.

Pre-Lab Questions section provides targetted questions for preliminary calculations that need to be done prior to the lab experiments. You should go through these questions and try to answer them using the background materials and the references given at the end of the manual.

Lab Experiments section provides you with step-by-step instructions to conduct the lab experiments and to record the collected data.

System Requirements section describes all the details of how to configure the hardware and software to conduct the experiments. It is assumed that the hardware and software configuration have been completed by the instructor or the teaching assistant *prior* to the lab sessions. If not, you can configure the systems by following the instructions given in this section.

When you write your lab report, you should use the specific template for content given at the end of each laboratory chapter. A section on *Tips for Report Format* is also provided at the end of each laboratory chapter.

LABORATORY 1

SRV02 MODELING

The objective of this experiment is to find a transfer function that describes the rotary motion of the SRV02 load shaft. The dynamic model is derived analytically from classical mechanics principles and using experimental methods.

Topics Covered

- Deriving the dynamics equation and transfer function for the SRV02 servo plant using the first-principles.
- Obtaining the SRV02 transfer function using a frequency response experiment.
- Obtaining the SRV02 transfer function using a bump test.
- Tuning the obtained transfer function and validating it with the actual system response.

Prerequisites
In order to successfully carry out this laboratory, the user should be familiar with the following:

- Data acquisition device (e.g. Q2-USB), the power amplifier (e.g. VoltPAQ-X1), and the main components of the SRV02 (e.g. actuator, sensors), as described in References [2], [4], and [6], respectively.
- Wiring and operating procedure of the SRV02 plant with the amplifier and data-aquisition (DAQ) device, as discussed in Reference [6].
- Transfer function fundamentals, e.g. obtaining a transfer function from a differential equation.
- Laboratory described in Appendix A to get familiar with using QUARC®with the SRV02.

1.1 Background

The angular speed of the SRV02 load shaft with respect to the input motor voltage can be described by the following first-order transfer function

$$\frac{\Omega_l(s)}{V_m(s)} = \frac{K}{(\tau s+1)} \tag{1.1.1}$$

where $\Omega_l(s)$ is the Laplace transform of the load shaft speed $\omega_l(t)$, $V_m(s)$ is the Laplace transform of motor input voltage $v_m(t)$, K is the steady-state gain, τ is the time constant, and s is the Laplace operator.

The SRV02 transfer function model is derived analytically in Section 1.1.1 and its K and τ parameters are evaluated. These are known as the nominal model parameter values. The model parameters can also be found experimentally. Sections 1.1.2.1 and 1.1.2.2 describe how to use the frequency response and bump-test methods to find K and τ. These methods are useful when the dynamics of a system are not known, for example in a more complex system. After the lab experiments, the experimental model parameters are compared with the nominal values.

1.1.1 Modeling Using First-Principles

1.1.1.1 Electrical Equations

The DC motor armature circuit schematic and gear train is illustrated in Figure 1.1. As specified in [6], recall that R_m is the motor resistance, L_m is the inductance, and k_m is the back-emf constant.

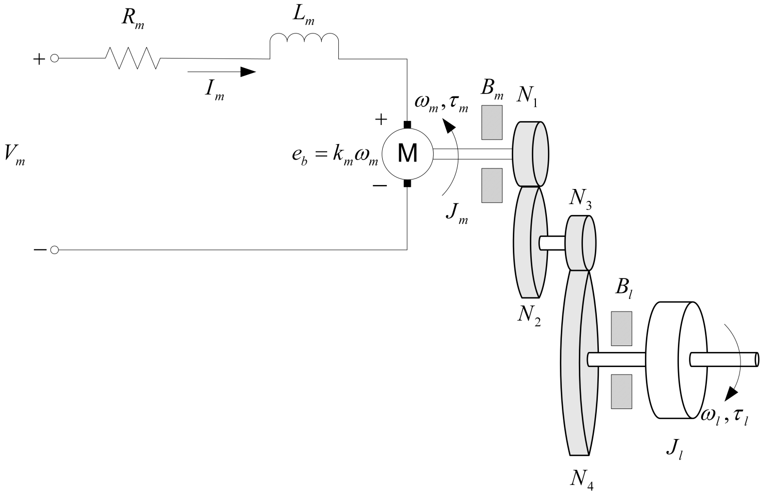


Figure 1.1: SRV02 DC motor armature circuit and gear train

The back-emf (electromotive) voltage $e_b(t)$ depends on the speed of the motor shaft, ω_m, and the back-emf constant of the motor, k_m. It opposes the current flow. The back emf voltage is given by:

$$e_b(t) = k_m\omega_m(t) \tag{1.1.2}$$

Using Kirchoff's Voltage Law, we can write the following equation:

$$V_m(t) - R_m I_m(t) - L_m\frac{dI_m(t)}{dt} - k_m\omega_m(t) = 0 \tag{1.1.3}$$

Since the motor inductance L_m is much less than its resistance, it can be ignored. Then, the equation becomes

$$V_m(t) - R_m I_m(t) - k_m\omega_m(t) = 0 \tag{1.1.4}$$

Solving for $I_m(t)$, the motor current can be found as:

$$I_m(t) = \frac{V_m(t) - k_m \omega_m(t)}{R_m} \tag{1.1.5}$$

1.1.1.2 Mechanical Equations

In this section the equation of motion describing the speed of the load shaft, ω_l, with respect to the applied motor torque, τ_m, is developed.

Since the SRV02 is a one degree-of-freedom rotary system, Newton's Second Law of Motion can be written as:

$$J \cdot \alpha = \tau \tag{1.1.6}$$

where J is the moment of inertia of the body (about its center of mass), α is the angular acceleration of the system, and τ is the sum of the torques being applied to the body. As illustrated in Figure 1.1, the SRV02 gear train along with the viscous friction acting on the motor shaft, B_m, and the load shaft B_l are considered. The load equation of motion is

$$J_l \frac{d\omega_l(t)}{dt} + B_l \omega_l(t) = \tau_l(t) \tag{1.1.7}$$

where J_l is the moment of inertia of the load and τ_l is the total torque applied on the load. The load inertia includes the inertia from the gear train and from any external loads attached, e.g. disc or bar. The motor shaft equation is expressed as:

$$J_m \frac{d\omega_m(t)}{dt} + B_m \omega_m(t) + \tau_{ml}(t) = \tau_m(t) \tag{1.1.8}$$

where J_m is the motor shaft moment of inertia and τ_{ml} is the resulting torque acting on the motor shaft from the load torque. The torque at the load shaft from an applied motor torque can be written as:

$$\tau_l(t) = \eta_g K_g \tau_{ml}(t) \tag{1.1.9}$$

where K_g is the gear ratio and η_g is the gearbox efficiency. The planetary gearbox that is directly mounted on the SRV02 motor (see [6] for more details) is represented by the N_1 and N_2 gears in Figure 1.1 and has a gear ratio of

$$K_{gi} = \frac{N_2}{N_1} \tag{1.1.10}$$

This is the *internal* gear box ratio. The motor gear N_3 and the load gear N_4 are directly meshed together and are visible from the outside. These gears comprise the *external* gear box which has an associated gear ratio of

$$K_{ge} = \frac{N_4}{N_3} \tag{1.1.11}$$

The gear ratio of the SRV02 gear train is then given by:

$$K_g = K_{ge} K_{gi} \tag{1.1.12}$$

Thus, the torque seen at the motor shaft through the gears can be expressed as:

$$\tau_{ml}(t) = \frac{\tau_l(t)}{\eta_g K_g} \tag{1.1.13}$$

Intuitively, the motor shaft must rotate K_g times for the output shaft to rotate one revolution.

$$\theta_m(t) = K_g \theta_l(t) \tag{1.1.14}$$

We can find the relationship between the angular speed of the motor shaft, ω_m, and the angular speed of the load shaft, ω_l by taking the time derivative:

$$\omega_m(t) = K_g \omega_l(t) \tag{1.1.15}$$

To find the differential equation that describes the motion of the load shaft with respect to an applied motor torque substitute (1.1.13), (1.1.15) and (1.1.7) into (1.1.8) to get the following:

$$J_m K_g \frac{d\omega_l(t)}{dt} + B_m K_g \omega_l(t) + \frac{J_l(\frac{d\omega_l(t)}{dt}) + B_l \omega_l(t)}{\eta_g K_g} = \tau_m(t) \tag{1.1.16}$$

Collecting the coefficients in terms of the load shaft velocity and acceleration gives

$$(\eta_g K_g^2 J_m + J_l)\frac{d\omega_l(t)}{dt} + (\eta_g K_g^2 B_m + B_l)\omega_l(t) = \eta_g K_g \tau_m(t) \tag{1.1.17}$$

Defining the following terms:

$$J_{eq} = \eta_g K_g^2 J_m + J_l \tag{1.1.18}$$

$$B_{eq} = \eta_g K_g^2 B_m + B_l \tag{1.1.19}$$

simplifies the equation as:

$$J_{eq} \frac{d\omega_l(t)}{dt} + B_{eq} \omega_l(t) = \eta_g K_g \tau_m(t) \tag{1.1.20}$$

1.1.1.3 Combining the Electrical and Mechanical Equations

In this section the electrical equation derived in Section 1.1.1.1 and the mechanical equation found in Section 1.1.1.2 are brought together to get an expression that represents the load shaft speed in terms of the applied motor voltage.

The motor torque is proportional to the voltage applied and is described as

$$\tau_m(t) = \eta_m k_t I_m(t) \tag{1.1.21}$$

where k_t is the current-torque constant $(N.m/A)$, η_m is the motor efficiency, and I_m is the armature current. See [6] for more details on the SRV02 motor specifications.

We can express the motor torque with respect to the input voltage $V_m(t)$ and load shaft speed $\omega_l(t)$ by substituting the motor armature current given by equation 1.1.5 in Section 1.1.1.1, into the current-torque relationship given in equation 1.1.21:

$$\tau_m(t) = \frac{\eta_m k_t \left(V_m(t) - k_m \omega_m(t)\right)}{R_m} \tag{1.1.22}$$

To express this in terms of V_m and ω_l, insert the motor-load shaft speed equation 1.1.15, into 1.1.21 to get:

$$\tau_m(t) = \frac{\eta_m k_t \left(V_m(t) - k_m K_g \omega_l(t)\right)}{R_m} \tag{1.1.23}$$

If we substitute (1.1.23) into (1.1.20), we get:

$$J_{eq}\left(\frac{d}{dt} w_l(t)\right) + B_{eq} w_l(t) = \frac{\eta_g K_g \eta_m k_t \left(V_m(t) - k_m K_g \omega_l(t)\right)}{R_m} \tag{1.1.24}$$

After collecting the terms, the equation becomes

$$\left(\frac{d}{dt} w_l(t)\right) J_{eq} + \left(\frac{k_m \eta_g K_g^2 \eta_m k_t}{R_m} + B_{eq}\right) \omega_l(t) = \frac{\eta_g K_g \eta_m k_t V_m(t)}{R_m} \tag{1.1.25}$$

This equation can be re-written as:

$$\left(\frac{d}{dt}w_l(t)\right) J_{eq} + B_{eq,v}\omega_l(t) = A_m\, V_m(t) \tag{1.1.26}$$

where the equivalent damping term is given by:

$$B_{eq,v} = \frac{\eta_g K_g^2 \eta_m k_t k_m + B_{eq} R_m}{R_m} \tag{1.1.27}$$

and the actuator gain equals

$$A_m = \frac{\eta_g K_g \eta_m k_t}{R_m} \tag{1.1.28}$$

1.1.2 Modeling Using Experiments

In Section 1.1.1 you learned how the system model can be derived from the first-principles. A linear model of a system can also be determined purely experimentally. The main idea is to experimentally observe how a system reacts to different inputs and change structure and parameters of a model until a reasonable fit is obtained. The inputs can be chosen in many different ways and there are a large variety of methods. In Sections 1.1.2.1 and 1.1.2.2, two methods of modeling the SRV02 are outlined: (1) frequency response and, (2) bump test.

1.1.2.1 Frequency Response

In Figure 1.2, the response of a typical first-order time-invariant system to a sine wave input is shown. As it can be seen from the figure, the input signal (u) is a sine wave with a fixed amplitude and frequency. The resulting output (y) is also a sinusoid with the *same* frequency but with a different amplitude. By varying the frequency of the input sine wave and observing the resulting outputs, a Bode plot of the system can be obtained as shown in Figure 1.3.

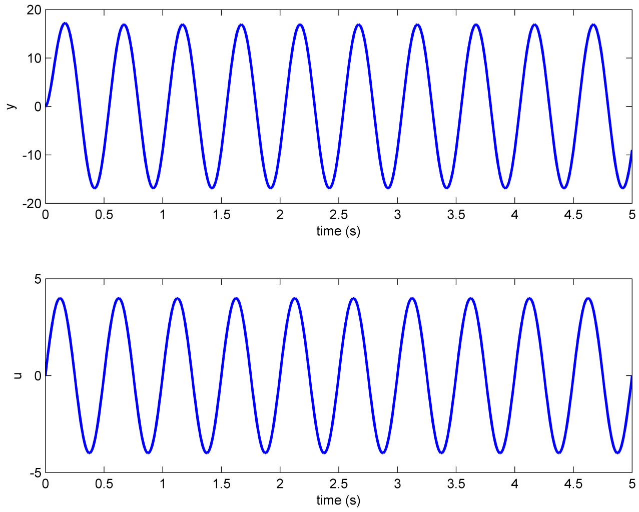


Figure 1.2: Typical frequency response

The Bode plot can then be used to find the steady-state gain, i.e. the DC gain, and the time constant of the system. The cuttoff frequency, ω_c, shown in Figure 1.3 is defined as the frequency where the gain is 3 dB less than the maximum gain (i.e. the DC gain). When working in the linear non-decibel range, the 3 dB frequency is defined as the frequency where the gain is $\frac{1}{\sqrt{2}}$, or about 0.707, of the maximum gain. The cutoff frequency is also known as the bandwidth of the system which represents how fast the system responds to a given input.

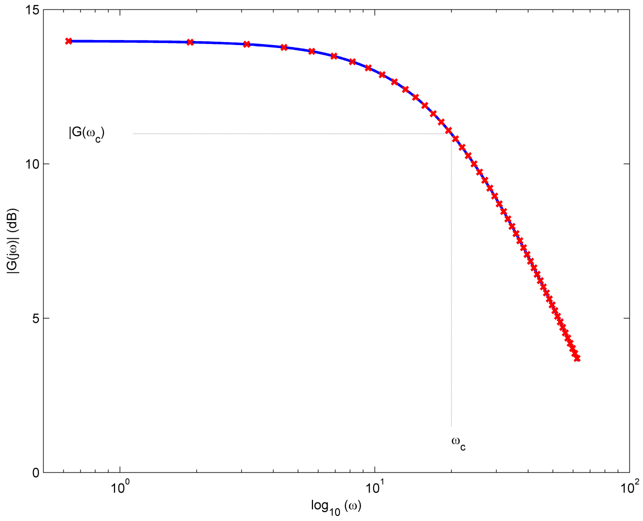


Figure 1.3: Magnitude Bode plot

The magnitude of the frequency response of the SRV02 plant transfer function given in equation 1.1.1 is defined as:

$$|G_{wl,v}(w)| = \left| \frac{\Omega_l(\omega\, j)}{V_m(\omega\, j)} \right| \tag{1.1.29}$$

where ω is the frequency of the motor input voltage signal V_m. We know that the transfer function of the system has the generic first-order system form given in Equation 1.1.1. By substituting $s = j\, w$ in this equation, we can find the frequency response of the system as:

$$\frac{\Omega_l(\omega\, j)}{V_m(\omega\, j)} = \frac{K}{\tau\omega\, j + 1} \tag{1.1.30}$$

Then, the magnitude of it equals

$$|G_{wl,v}(\omega)| = \frac{K}{\sqrt{1 + \tau^2\, \omega^2}} \tag{1.1.31}$$

Let's call the frequency response model parameters $K_{e,f}$ and $\tau_{e,f}$ to differentiate them from the nominal model parameters, K and τ, used previously. The steady-state gain or the DC gain (i.e. gain at zero frequency) of the model is:

$$K_{e,f} = |G_{wl,v}(0)| \tag{1.1.32}$$

1.1.2.2 Bump Test

The bump test is a simple test based on the step response of a stable system. A step input is given to the system and its response is recorded. As an example, consider a system given by the following transfer function:

$$\frac{Y(s)}{U(s)} = \frac{K}{\tau s + 1} \tag{1.1.33}$$

The step response shown in Figure 1.4 is generated using this transfer function with $K = 5$ rad/V.s and $\tau = 0.05$ s.

The step input begins at time t_0. The input signal has a minimum value of u_{min} and a maximum value of u_{max}. The resulting output signal is initially at y_0. Once the step is applied, the output tries to follow it and eventually settles at its steady-state value y_{ss}. From the output and input signals, the steady-state gain is

$$K = \frac{\Delta y}{\Delta u} \tag{1.1.34}$$

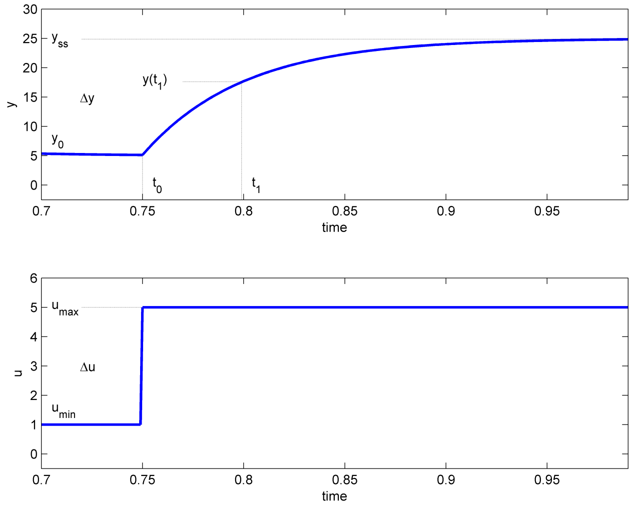


Figure 1.4: Input and output signal used in the bump test method

where $\Delta y = y_{ss} - y_0$ and $\Delta u = u_{max} - u_{min}$. In order to find the model time constant, τ, we can first calculate where the output is supposed to be at the time constant from:

$$y(t_1) = 0.632 y_{ss} + y_0 \tag{1.1.35}$$

Then, we can read the time t_1 that corresponds to $y(t_1)$ from the response data in Figure 1.4. From the figure we can see that the time t_1 is equal to:

$$t_1 = t_0 + \tau \tag{1.1.36}$$

From this, the model time constant can be found as:

$$\tau = t_1 - t_0 \tag{1.1.37}$$

Going back to the SRV02 system, a step input voltage with a time delay t_0 can be expressed as follows in the Laplace domain:

$$V_m(s) = \frac{A_v \, e^{(-s\,t_0)}}{s} \tag{1.1.38}$$

where A_v is the amplitude of the step and t_0 is the step time (i.e. the delay). If we substitute this input into the system transfer function given in Equation (1.1.1), we get:

$$\Omega_l(s) = \frac{K A_v \, e^{(-s\,t_0)}}{(\tau\,s + 1)\,s} \tag{1.1.39}$$

We can then find the SRV02 load speed step response, $w_l(t)$, by taking inverse Laplace of this equation. Here we need to be careful with the time delay t_0 and note that the initial condition is $\omega_l(0^-) = \omega_l(t_0)$.

$$\omega_l(t) = K\,A_v \left(1 - e^{\left(-\frac{t-t_0}{\tau}\right)}\right) + \omega_l\,(t_0) \tag{1.1.40}$$

1.2 Pre-Lab Questions

Before you start the lab experiments given in Section 1.3, you should study the background materials provided in Section 1.1 and work through the questions in this Section.

1. In Section 1.1.1.3 we obtained an equation (1.1.26) that described the dynamic behavior of the load shaft speed as a function of the motor input voltage. Starting from this equation, find the transfer function $\frac{\Omega_l(s)}{V_m(s)}$.
2. Express the steady-state gain (K) and the time constant (τ) of the process model (Equation (1.1.1)) in terms of the J_{eq}, $B_{eq,v}$, and A_m parameters.
3. Calculate the $B_{eq,v}$ and A_m model parameters using the system specifications given in [6]. The parameters are to be calculated based on an SRV02-ET in the high-gear configuration.
4. Calculate the moment of inertia about the motor shaft. Note that $J_m = J_{tach} + J_{m,rotor}$ where J_{tach} and $J_{m,rotor}$ are the moment of inertia of the tachometer and the rotor of the SRV02 DC motor, respectively. Use the specifications given in [6].
5. The load attached to the motor shaft includes a 24-tooth gear, two 72-tooth gears, and a single 120-tooth gear along with any other external load that is attached to the load shaft. Thus, for the gear moment of inertia J_g and the external load moment of inertia $J_{l,ext}$, the load inertia is $J_l = J_g + J_{l,ext}$. Using the specifications given in [6] find the total moment of inertia J_g from the gears . **Hint:** Use the definition of moment of inertia for a disc $J_{disc} = \frac{mr^2}{2}$.
6. Assuming the disc load is attached to the load shaft, calculate the inertia of the disc load, $J_{ext,l}$, and the total load moment of inertia, J_l.
7. Evaluate the equivalent moment of inertia J_{eq}.
8. Calculate the steady-state model gain K and time constant τ. These are the *nominal model parameters* and will be used to compare with parameters that are later found experimentally.
9. Referring to Section 1.1.2.1, find the expression representing the time constant τ of the frequency response model given in Equation 1.1.31. Begin by evaluating the magnitude of the transfer function at the cutoff frequency ω_c.
10. Referring to Section 1.1.2.2, find the steady-state gain of the step response and compare it with Equation 1.1.34. **Hint:** The the steady-state value of the load shaft speed can be defined as $\omega_{l,ss} = \lim_{t \to \infty} \omega_l(t)$.
11. Evaluate the step response given in equation 1.1.40 at $t = t_0 + \tau$ and compare it with Equation 1.1.34.

1.3 Lab Experiments

The main goal of this laboratory is to find a transfer function (model) that describes the rotary motion of the SRV02 load shaft as a function of the input voltage. We can obtain this transfer function experimentally using one of the following two methods:

- Frequency response, or
- Bump test

In this laboratory, first you will conduct two experiments exploring how these methods can be applied to a real system. Then, you will conduct a third experiment to fine tune the parameters of the transfer functions you obtained and to validate them.

Experimental Setup

The q_srv02_mdl Simulink diagram shown in Figure 1.5 will be used to conduct the experiments. The SRV02-ET subsystem contains QUARC®blocks that interface with the DC motor and sensors of the SRV02 system as discussed in Reference [4]. The SRV02 Model uses a *Transfer Fcn* block from the Simulink®library to simulate the SRV02 system. Thus, both the measured and simulated load shaft speed can be monitored simultaneously given an input voltage.

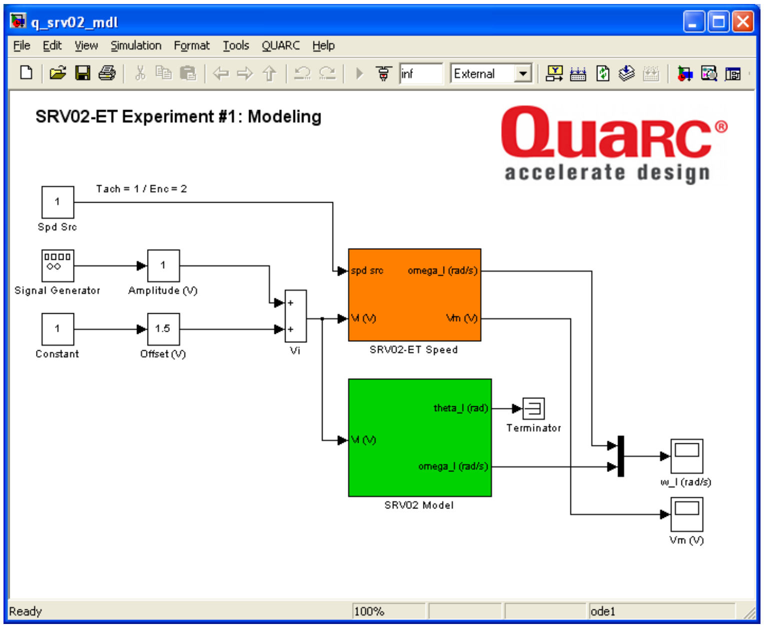


Figure 1.5: q_srv02_mdl Simulink diagram used to model SRV02.

IMPORTANT: Before you can conduct these experiments, you need to make sure that the lab files are configured according to your SRV02 setup. If they have not been configured already, then you need to go to Section 1.4.2 to configure the lab files first.

1.3.1 Frequency Response Experiment

As explained in 1.1.2.1 earlier, the frequency response of a linear system can be obtained by providing a sine wave input signal to it and recording the resulting output sine wave from it. In this experiment, the input signal is the motor voltage and the output is the motor speed.

In this method, we keep the amplitude of the input sine wave constant but vary its frequency. At each frequency setting, we record the amplitude of the output sine wave. The ratio of the output and input amplitudes at a given frequency can then be used to create a Bode magnitude plot. Then, the transfer function for the system can be extracted from this Bode plot.

1.3.1.1 Steady-state gain

First, we need to find the steady-state gain of the system. This requires running the system with a constant input voltage. To create a $2V$ constant input voltage follow these steps:

1. In the Simulink®diagram, double-click on the *Signal Generator* block and ensure the following parameters are set:
    - Wave form: sine
    - Amplitude: 1.0
    - Frequency: 0.0
    - Units: Hertz
2. Set the *Amplitude (V)* slider gain to 0.
3. Set the *Offset (V)* block to 2.0 V.
4. Open the load shaft speed scope, *w_l (rad/s)*, and the motor input voltage scope, *V_m (V)*.
5. Click on QUARC | Build to compile the Simulink diagram.
6. Select QUARC | Start to run the controller. The SRV02 unit should begin rotating in one direction. The scopes should be reading something similar to Figures 1.6 and 1.7. Note that in the *w_l (rad/s)* scope, the yellow trace is the measured speed while the purple trace is the simulated speed (generated by the SRV02 Model block).

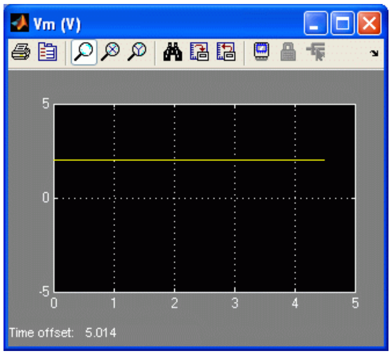


Figure 1.6: Constant input motor voltage.

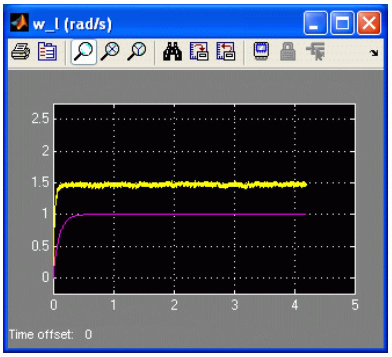


Figure 1.7: Load shaft speed response to a constant input.

7. Measure the speed of the load shaft and enter the measurement in Table 1.1 below under the *f = 0 Hz* row.

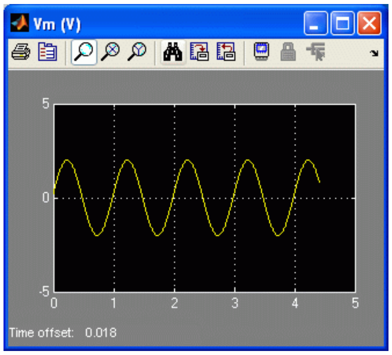
    The measurement can be done directly from the scope. Alternatively, you can use Matlab®commands to find the maximum load speed using the saved *wl* variable. When the controller is stopped, the *w_l (rad/s)* scope saves the last 5 seconds of response data to the Matlab®workspace in the *wl* parameter. It has the following structure: *wl(:,1)* is the time vector, *wl(:,2)* is the measured speed, and *wl(:,3)* is the simulated speed.
8. Calculate the steady-state gain both in linear and decibel (dB) units as explained in 1.1.2.1. Enter the resulting numerical value in the *f = 0 Hz* row of Table 1.1. Also, enter its non-decibel value in Table 1.2 in Section 1.3.4.

1.3.1.2 Gain at varying frequencies

In this part of the experiment, we will send an input sine wave at a certain frequency to the system and record the amplitude of the output signal. We will then increment the frequency and repeat the same observation.

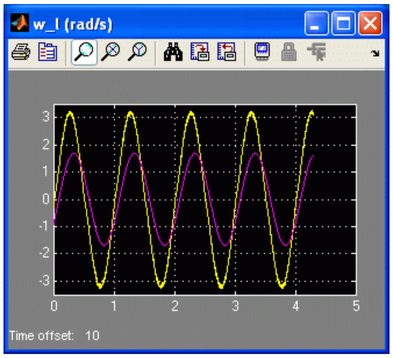
To create the input sine wave:

1. Set the *Offset (V)* block to 0 V.
2. Set the *Amplitude (V)* slider gain to 2.0 V.
3. The SRV02 unit should begin rotating smoothly back and forth and the scopes should be reading a response similar to Figures 1.8 and 1.9.



Time offset: 0.018

Figure 1.8: Input motor voltage scope.



Time offset: 10

Figure 1.9: Load shaft speed sine wave response.

4. Measure the maximum positive speed of the load shaft at f = 1.0 Hz input and enter it in Table 1.1 below.

   As before, this measurement can be done directly from the scope or, preferably, you can use Matlab®commands to find the maximum load speed using the saved *wl* variable.
5. Calculate the gain of the system (in both linear and dB units) and enter the results in Table 1.1.
6. Now increase the frequency to f = 2.0 Hz by adjusting the frequency parameter in the *Signal Generator* block. Measure the maximum load speed and calculate the gain. Repeat this step for each of the frequency settings in Table 1.1.
7. Using the Matlab®*plot* command and the data collected in Table 1.1, generate a Bode magnitude plot. Make sure the amplitude and frequency scales are in decibels. When making the Bode plot, ignore the f = 0 Hz entry as the logarithm of 0 is not defined.
8. Calculate the time constant $\tau_{e,f}$ using the obtained Bode plot by finding the cutoff frequency. Label the Bode plot with the -3 dB gain and the cutoff frequency. Enter the resulting time constant in Table 1.2. **Hint:** Use the *ginput* command to obtain values from the Matlab®figure.
9. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.
10. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

f (Hz)	Amplitude (V)	Maximum Load Speed (rad/s)	Gain:|G(ω)| (rad/s/V)	Gain:|G(ω)| (rad/s/V, dB)
0.0	2.0			
1.0	2.0			
2.0	2.0			
3.0	2.0			
4.0	2.0			
5.0	2.0			
6.0	2.0			
7.0	2.0			
8.0	2.0			

Table 1.1: Collected frequency response data.

1.3.2 Bump Test Experiment

In this method, a step input is given to the SRV02 and the corresponding load shaft response is recorded. Using the saved response, the model parameters can then be found as discussed in Section 1.1.2.2.

To create the step input:

1. Double-click on the *Signal Generator* block and ensure the following parameters are set:
   - Wave form: square
   - Amplitude: 1.0
   - Frequency: 0.4
   - Units: Hertz
2. Set the Amplitude *(V)* slider gain to *1.5 V*.
3. Set the Offset *(V)* block to *2.0 V*.
4. Open the load shaft speed scope, *w_l (rad/s)*, and the motor input voltage cope, *V_m (V)*.
5. Click on QUARC | Build to compile the Simulink diagram.
6. Select QUARC | Start to run the controller. The gears on the SRV02 should be rotating in the same direction and alternating between low and high speeds. The response in the scopes should be similar to Figures 1.10 and 1.11.

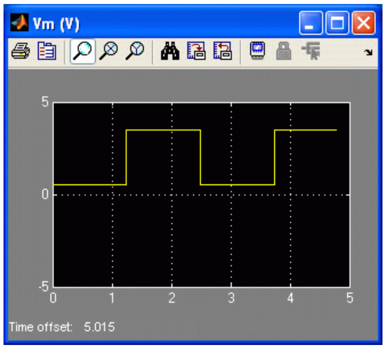


Figure 1.10: Square input motor voltage.

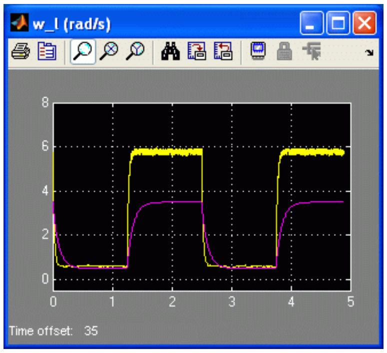


Figure 1.11: Load shaft speed step response.

7. Plot the response in Matlab®. Recall that the maximum load speed is saved in the Matlab®workspace under the *wl* variable.

8. Find the steady-state gain using the measured step response and enter it in Table 1.2. *Hint:* Use the Matlab®*ginput* command to measure points off the plot.

9. Find the time constant from the obtained response and enter the result in Table 1.2.

10. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.

11. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

1.3.3 Model Validation Experiment

In this experiment, you will adjust the model parameters you found in the previous experiments to tune the transfer function. Our goal is to match the simulated system response with the parameters you found as closely as possible to the response of the actual system.

To create a step input:

1. Double-click on the *Signal Generator* block and ensure the following parameters are set:
   - Wave form: square
   - Amplitude: 1.0
   - Frequency: 0.4
   - Units: Hertz

2. Set the *Amplitude (V)* slider gain to 1.0 V.

3. Set the *Offset (V)* block to 1.5 V.

4. Open the load shaft speed scope, *w_l (rad/s)*, and the motor input voltage scope, *V_m (V)*.

5. Click on QUARC | Build to compile the Simulink®diagram.

6. Select QUARC | Start to run the controller. The gears on the SRV02 should be rotating in the same direction and alternating between low and high speeds and the scopes should be as shown in figures 1.12 and 1.13. Recall that the yellow trace is the measured load shaft rate and the purple trace is the simulated trace. By default, the steady-state gain and the time constant of the transfer function used in simulation are set to: $K = 1\,rad/s/V$ and $\tau = 0.1s$. These model parameters do not accurately represent the system.

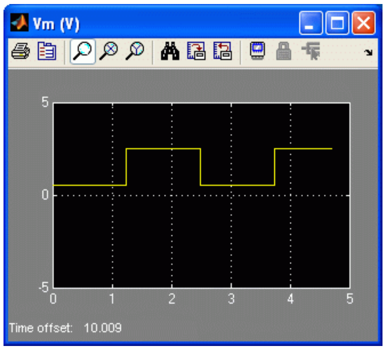


Figure 1.12: Input square voltage.

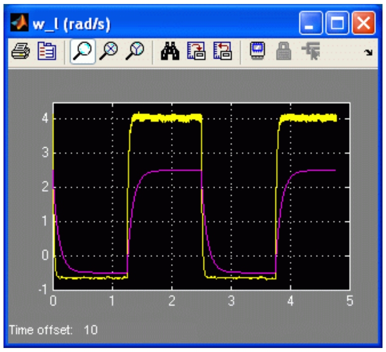


Figure 1.13: Speed step response. Simulation done with default model parameters: $K = 1$ and $\tau = 0.1$.

7. Enter the command $K = 1.25$ in the Matlab®Command Window.

8. Update the parameters used by the Transfer Function block in the simulation by selecting the Edit | Update Diagram item in the q_srv02_mdl Simulink®diagram and observe how the simulation changes.

9. Enter the command $tau = 0.2$ in the Matlab®Command Window.

10. Update the simulation again by selecting the Edit | Update Diagram and observe how the simulation changes.

11. Vary the gain and time constant model parameters. How do the gain and the time constant affect the system response?

12. Enter the nominal values, K and τ, that were found in Section 1.2 in the Matlab®Command Window. Update the parameters and examine how well the simulated response matches the measured one.

13. If the calculations were done properly, then the model should represent the actual system quite well. However, there are always some differences between each servo unit and, as a result, the model can always be tuned to match the system better. Try varying the model parameters until the simulated trace matches the measured response better. Enter these tuned values under the Model Validation section of Table 1.2.

14. Provide two reasons why the nominal model does not represent the SRV02 with better accuracy.

15. Create a Matlab®figure that shows the measured and simulated response of each method (the nominal model, the frequency response model, and the bumptest model). Enter the nominal values, K and τ, in the Matlab®Command Window, update the parameters, and examine the response. Repeat for the frequency response parameters $K_{e,f}$ and $\tau_{e,f}$ along with the bump test variables $K_{e,b}$ and $\tau_{e,b}$.

16. Explain how well the nominal model, the frequency response model, and the bumptest model represent the SRV02 system.

17. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.

18. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

1.3.4 Results

Fill out Table 1.2 below, with your results.

Section	Description	Symbol	Value	Unit
1.2	**Nominal Values** Open-Loop Steady-State Gain Open-Loop Time Constant	 K τ		
1.3.1	**Frequency Response Exp.** Open-Loop Steady-State Gain Open-Loop Time Constant	 $K_{e,f}$ $\tau_{e,f}$		
1.3.2	**Bump Test Exp.** Open-Loop Steady-State Gain Open-Loop Time Constant	 $K_{e,b}$ $\tau_{e,b}$		
1.3.3	**Model Validation** Open-Loop Steady-State Gain Open-Loop Time Constant	 $K_{e,v}$ $\tau_{e,v}$		

Table 1.2: Summary of results for the SRV02 Modeling laboratory.

1.4 System Requirements

Before you begin this laboratory make sure:

- QUARC®is installed on your PC, as described in Reference [1].
- You have a QUARC compatible data-aquisition (DAQ) card installed in your PC. For a listing of compliant DAQ cards, see Reference [5].
- SRV02 and amplifier are connected to your DAQ board as described Reference [6].

1.4.1 Overview of Files

Table 1.3: Files supplied with the SRV02 Modeling laboratory.

File Name	**Description**
01 - SRV02 Modeling - Student Manual.pdf	This laboratory guide contains pre-lab and in-lab exercises demonstrating how to model the Quanser SRV02 rotary plant. The in-lab exercises are explained using the QUARC software.
setup_srv02_exp01_mdl.m	The main Matlab®script that sets the SRV02 motor and sensor parameters. **Run this file only to setup the laboratory.**
config_srv02.m	Returns the configuration-based SRV02 model specifications *Rm, kt, km, Kg, eta_g, Beq, Jeq,* and *eta_m,* the sensor calibration constants *K_POT, K_ENC,* and *K_TACH,* and the amplifier limits *VMAX_AMP* and *IMAX_AMP*.
calc_conversion_constants.m	Returns various conversions factors.
q_srv02_mdl.mdl	Simulink®file that implements the open-loop controller for the SRV02 system using QUARC.

1.4.2 Configuring the SRV02 and the Lab Files

Before beginning the lab exercises the SRV02 device, the q_srv02_mdl Simulink®diagram, and the setup_srv02_exp02.m script must be configured.

Follow these steps to get the system ready for this lab:

1. Set up the SRV02 in the high-gear configuration and with the disc load as described in Reference [5].
2. Load the Matlab®software.
3. Browse through the Current Directory window in Matlab®and find the folder that contains the SRV02 modeling files, e.g. q_srv02_mdl.mdl.
4. Double-click on the q_srv02_mdl.mdl file to open the Simulink®diagram shown in Figure 1.5.
5. **Configure DAQ:** Double-click on the HIL Initialize block in the Simulink®diagram and ensure it is configured for the DAQ device that is installed in your system. For instance, the block shown in Figure 1.5 is setup for the Quanser Q8 hardware-in-the-loop board. See [1] for more information on configuring the HIL Initialize block.

6. **Configure Sensor:** The speed of the load shaft can be measured using various sensors. Set the *Spd Src* Source block in q_srv02_mdl, as shown in Figure 1.5, as follows:

    - 1 to use tachometer
    - 2 to use the encoder

7. It is recommended that the tachometer sensor be used to perform this laboratory. However, for users who do not have a tachometer with their servo, e.g. SRV02 or SRV02-E options, they may choose to use the encoder with a high-pass filter to get a velocity measurement.

8. Go to the *Current Directory* window and double-click on the setup_srv02_exp01_mdl.m file to open the setup script for the q_srv02_mdl Simulink®model.

9. **Configure setup script:** The beginning of the setup script is shown below. Ensure the script is setup to match the configuration of your actual SRV02 device. For example, the script given below is setup for an SRV02-ET plant in the high-gear configuration mounted with a disc load and it is actuated using the Quanser VoltPAQ device with a motor cable gain of 1. See [6] for more information on SRV02 plant options and corresponding accessories.
    Finally, make sure MODELING_TYPE is set to 'MANUAL'.

```
%% SRV02 Configuration
% External Gear Configuration: set to 'HIGH' or 'LOW'
EXT_GEAR_CONFIG = 'HIGH';
% Encoder Type: set to 'E' or 'EHR'
ENCODER_TYPE = 'E';
% Is SRV02 equipped with Tachometer? (i.e. option T): set to 'YES' or 'NO'
TACH_OPTION = 'YES';
% Type of Load: set to 'NONE', 'DISC', or 'BAR'
LOAD_TYPE = 'DISC';
% Amplifier Gain: set VoltPAQ amplifier gain to 1
K_AMP = 1;
% Power Amplifier Type: set to 'VoltPAQ', 'UPM_1503', 'UPM_2405', or 'Q3'
AMP_TYPE = 'VoltPAQ';
% Digital-to-Analog Maximum Voltage (V)
VMAX_DAC = 10;
%
%% Lab Configuration
% Type of Controller: set it to 'AUTO', 'MANUAL'
MODELING_TYPE = 'AUTO';
% MODELING_TYPE = 'MANUAL';
```

10. Run the script by selecting the Debug | Run item from the menu bar or clicking on the *Run* button in the tool bar. The messages shown below should be generated in the Matlab®Command Window. *These are default model parameters and do not accurately represent the SRV02 system.*

```
Calculated SRV02 model parameter:
   K = 1 rad/s/V
   tau = 0.1 s
```

1.5 Lab Report

When you prepare your lab report, you can follow the outline given in Section 1.5.1 to build the *content* of your report. Also, in Section 1.5.2 you can find some basic tips for the *format* of your report.

1.5.1 Template for Content

I. PROCEDURE

I.1. Frequency Response Experiment

1. Briefly describe the main goal of this experiment and the procedure.
    - Briefly describe the experimental procedure (Section 1.3.1.1), *Steady-state gain*
    - Briefly describe the experimental procedure (Section 1.3.1.2), *Gain at varying frequencies*

I.2. Bump Test Experiment

1. Briefly describe the main goal of this experiment and the experimental procedure (Section 1.3.2).

I.3. Model Validation Experiment

1. Briefly describe the main goal of this experiment and the experimental procedure (Section 1.3.3).

II. RESULTS
Do not interpret or analyze the data in this section. Just provide the results.

1. Bode plot from step 7 in Section 1.3.1.2, *Gain at varying frequencies*.
2. Response plot from step 7 in Section 1.3.2, *Bump Test Experiment*.
3. Response plot from step 15 in Section 1.3.3, *Model Validation Experiment*.
4. Provide data collected in this laboratory (from Table 1.1).

III. ANALYSIS
Provide details of your calculations (methods used) for analysis for each of the following:

III.1. Frequency Response Experiment

1. Step 8 in Section 1.3.1.1, *Steady-state gain*.
2. Step 8 in Section 1.3.1.2, *Gain at varying frequencies*.

III.2. Bump Test Experiment

1. Steps 8 and 9 in Section 1.3.2.

IV. CONCLUSIONS
Interpret your results to arrive at logical conclusions.

1. Steps 11, 14, and 16 in Section 1.3.3.

1.5.2 Tips for Report Format

PROFESSIONAL APPEARANCE

- Has cover page with all necessary details (title, course, student name(s), etc.)
- Each of the required sections is completed (Procedure, Results, Analysis and Conclusions).
- Typed.
- All grammar/spelling correct.
- Report layout is neat.
- Does not exceed specified maximum page limit, if any.
- Pages are numbered.
- Equations are consecutively numbered.
- Figures are numbered, axes have labels, each figure has a descriptive caption.
- Tables are numbered, they include labels, each table has a descriptive caption.
- Data are presented in a useful format (graphs, numerical, table, charts, diagrams).
- No hand drawn sketches/diagrams.
- References are cited using correct format.

LABORATORY 2

SRV02 POSITION CONTROL

The objective of this laboratory is to develop feedback systems that control the position of the rotary servo load shaft. Using the proportional-integral-derivative (PID) family, controllers are designed to meet a set of specifications.

Topics Covered

- Design of a proportional-velocity (PV) controller for position control of the servo load shaft to meet certain time-domain requirements.
- Actuator saturation.
- Design of a proportional-velocity-integral (PIV) controller to track a ramp reference signal.
- Simulation of the PV and PIV controllers using the developed model of the plant to ensure the specifications are met without any actuator saturation.
- Implementation of the controllers on the Quanser SRV02 device to evaluate their performance.

Prerequisites
In order to successfully carry out this laboratory, the user should be familiar with the following:

- Data acquisition device (e.g. Q2-USB), the power amplifier (e.g. VoltPAQ-X1), and the main components of the SRV02 (e.g. actuator, sensors), as described in References [2], [4], and [6], respectively.
- Wiring and operating procedure of the SRV02 plant with the amplifier and data-aquisition (DAQ) device, as discussed in Reference [6].
- Transfer function fundamentals, e.g. obtaining a transfer function from a differential equation.
- Laboratory described in Appendix A to get familiar with using QUARC®with the SRV02.

2.1 Background

2.1.1 Desired Position Control Response

The block diagram shown in Figure 2.1 is a general unity feedback system with compensator (controller) C(s) and a transfer function representing the plant, P(s). The measured output, Y(s), is supposed to track the reference signal R(s) and the tracking has to match to certain desired specifications.

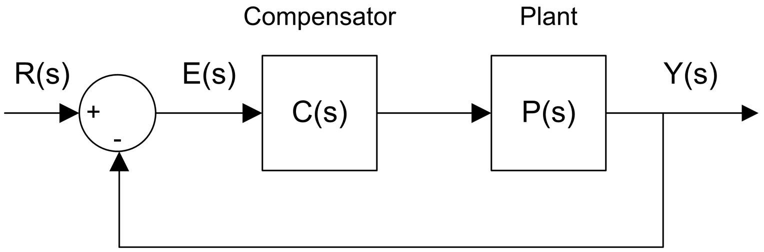


Figure 2.1: Unity feedback system.

The output of this system can be written as:

$$Y(s) = C(s)\,P(s)\,\left(R(s) - Y(s)\right) \tag{2.1.1}$$

By solving for $Y(s)$, we can find the closed-loop transfer function:

$$\frac{Y(s)}{R(s)} = \frac{C(s)\,P(s)}{1 + C(s)\,P(s)} \tag{2.1.2}$$

Recall in Laboratory: SRV02 Modelling Section 1, the SRV02 voltage-to-speed transfer function was derived. To find the voltage-to-position transfer function, we can put an integrator $(1/s)$ in series with the speed transfer function (effectively integrating the speed output to get position). Then, the resulting open-loop voltage-to-load gear position transfer function becomes:

$$P(s) = \frac{K}{s\,(\tau s + 1)} \tag{2.1.3}$$

As you can see from this equation, the plant is a second order system. In fact, when a second order system is placed in series with a proportional compensator in the feedback loop as in Figure 2.1, the resulting closed-loop transfer function can be expressed as:

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\,\omega_n\,s + \omega_n^2} \tag{2.1.4}$$

where ω_n is the natural frequency and ζ is the damping ratio. This is called the *standard second-order* transfer function. Its response proporeties depend on the values of ω_n and ζ.

2.1.1.1 Peak Time and Overshoot

Consider a second-order system as shown in Equation 2.1.4 subjected to a step input given by

$$R(s) = \frac{R_0}{s} \tag{2.1.5}$$

with a step amplitude of $R_0 = 1.5$. The system response to this input is shown in Figure 2.2, where the red trace is the response (output), $y(t)$, and the blue trace is the step input $r(t)$. The maximum value of the response is denoted by the variable y_{max} and it occurs at a time t_{max}. For a response similar to Figure 2.2, the percent overshoot is found using

$$PO = \frac{100\,(y_{max} - R_0)}{R_0} \tag{2.1.6}$$

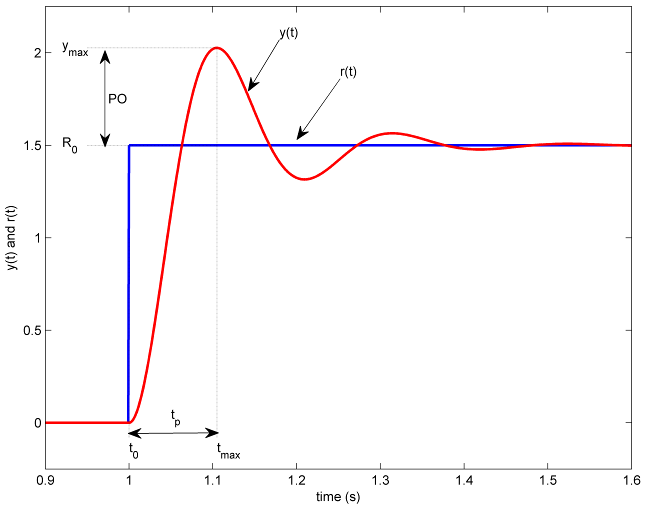


Figure 2.2: Standard second-order step response.

From the initial step time, t_0, the time it takes for the response to reach its maximum value is

$$t_p = t_{max} - t_0 \tag{2.1.7}$$

This is called the *peak time* of the system.

In a second-order system, the amount of overshoot depends solely on the damping ratio parameter and it can be calculated using the equation

$$PO = 100\, e^{\left(-\frac{\pi\,\zeta}{\sqrt{1-\zeta^2}}\right)} \tag{2.1.8}$$

The peak time depends on both the damping ratio and natural frequency of the system and it can be derived as:

$$t_p = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} \tag{2.1.9}$$

Generally speaking, the damping ratio affects the shape of the response while the natural frequency affects the speed of the response.

2.1.1.2 Steady State Error

Steady-state error is illustrated in the ramp response given in Figure 2.3 and is denoted by the variable e_{ss}. It is the difference between the reference input and output signals after the system response has settled. Thus, for a time t when the system is in steady-state, the steady-state error equals

$$e_{ss} = r_{ss}(t) - y_{ss}(t) \tag{2.1.10}$$

where $r_{ss}(t)$ is the value of the steady-state input and $y_{ss}(t)$ is the steady-state value of the output.

We can find the error transfer function $E(s)$ in Figure 2.1 in terms of the reference $R(s)$, the plant $P(s)$, and the compensator $C(s)$. The Laplace transform of the error is

$$E(s) = R(s) - Y(s) \tag{2.1.11}$$

Solving for $Y(s)$ from equation 2.1.3 and substituting it in equation 2.1.11 yields

$$E(s) = \frac{R(s)}{1 + C(s)\,P(s)} \tag{2.1.12}$$

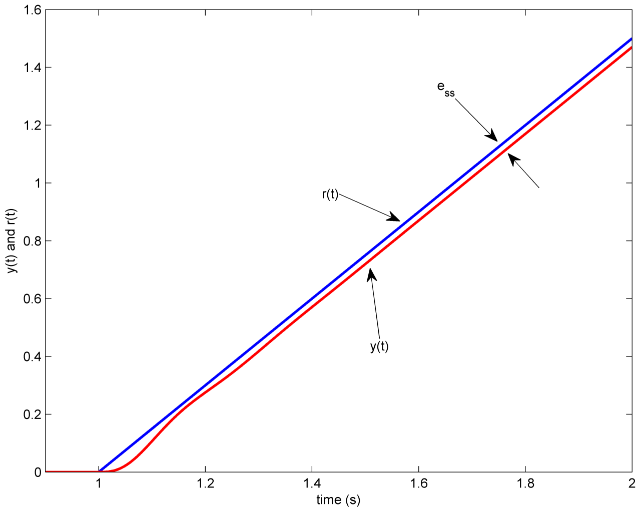


Figure 2.3: Steady-state error in ramp response.

We can find the the steady-state error of this system using the final-value theorem:

$$e_{ss} = \lim_{s \to 0} s\, E(s) \tag{2.1.13}$$

In this equation, we need to substitute the transfer function for E(s) from 2.1.12. The E(s) transfer function requires, R(s), C(s) and P(s). For simplicity, let C(s)=1 as a compensator. The P(s) and R(s) were given by equations 2.1.3 and 2.1.5, respectively. Then, the error becomes:

$$E(s) = \frac{R_0}{s\left(1 + \frac{K}{s\,(\tau s+1)}\right)} \tag{2.1.14}$$

Applying the final-value theorem gives

$$e_{ss} = R_0 \left(\lim_{s \to 0} \frac{(\tau\, s + 1)\; s}{\tau\, s^2 + s + K} \right) \tag{2.1.15}$$

When evaluated, the resulting steady-state error due to a step response is

$$e_{ss} = 0 \tag{2.1.16}$$

Based on this zero steady-state error for a step input, we can conclude that the SRV02 is a *Type 1* system.

2.1.1.3 SRV02 Position Control Specifications

The desired time-domain specifications for controlling the position of the SRV02 load shaft are:

$$e_{ss} = 0 \tag{2.1.17}$$

$$t_p = 0.20 \text{ s} \tag{2.1.18}$$

and

$$PO = 5.0 \text{ \%} \tag{2.1.19}$$

Thus, when tracking the load shaft reference, the transient response should have a peak time less than or equal to 0.20 seconds, an overshoot less than or equal to 5 %, and the steady-state response should have no error.

2.1.2 PV Controller Design

2.1.2.1 Closed Loop Transfer Function

The proportional-velocity (PV) compensator to control the position of the SRV02 has the following structure

$$V_m(t) = k_p\,(\theta_d(t) - \theta_l(t)) - k_v\left(\frac{d}{dt}\,\theta_l(t)\right) \tag{2.1.20}$$

where k_p is the proportional control gain, k_v is the velocity control gain, $\theta_d(t)$ is the setpoint or reference load shaft angle, $\theta_l(t)$ is the measured load shaft angle, and $V_m(t)$ is the SRV02 motor input voltage. The block diagram of the PV control is given in Figure 2.4. We need to find the closed-loop transfer function $\Theta_l(s)/\Theta_d(s)$ for the closed-loop position control of the SRV02. Taking the Laplace transform of equation 2.1.20 gives

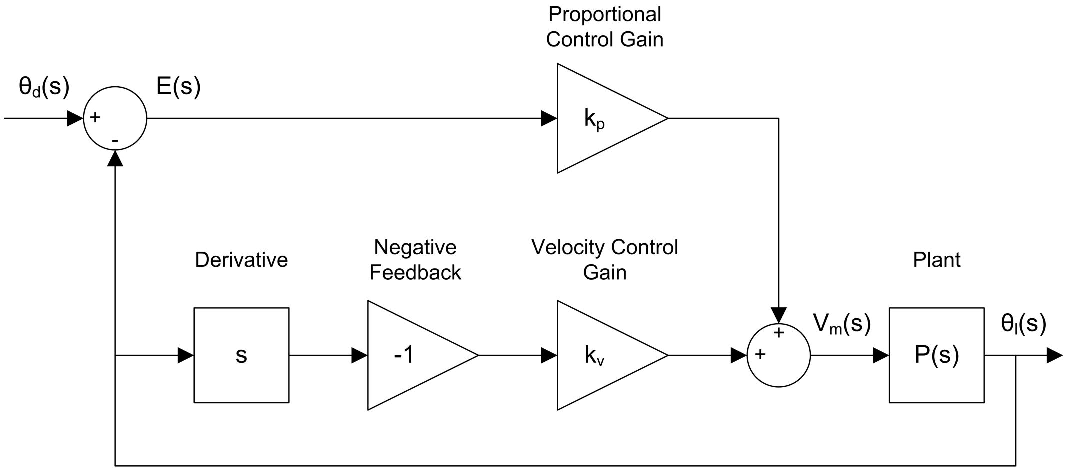


Figure 2.4: Block diagram of SRV02 PV position control.

$$V_m(s) = k_p\,(\Theta_d(s) - \Theta_l(s)) - k_v\,s\,\Theta_l(s) \tag{2.1.21}$$

From the Plant block in Figure 2.4 and equation 2.1.3, we can write

$$\frac{\Theta_l(s)}{V_m(s)} = \frac{K}{s\,(\tau\,s+1)} \tag{2.1.22}$$

Substituting equation 2.1.21 into 2.1.22 and solving for $\Theta_l(s)/\Theta_d(s)$ gives the SRV02 position closed-loop transfer function as:

$$\frac{\Theta_l(s)}{\Theta_d(s)} = \frac{K\,k_p}{\tau\,s^2 + (1+K\,k_v)\,s + K\,k_p} \tag{2.1.23}$$

2.1.2.2 Controller Gain Limits

In control design, a factor to be considered is saturation. This is a nonlinear element and is represented by a saturation block as shown in Figure 2.5. In a system like the SRV02, the computer calculates a numeric control voltage value. This value is then converted into a voltage, $V_{dac}(t)$, by the digital-to-analog converter of the data-acquisition device in the computer. The voltage is then amplified by a power amplifier by a factor of K_a. If the amplified voltage, $V_{amp}(t)$, is greater than the maximum output voltage of the amplifier or the input voltage limits of the motor (whichever is smaller), then it is saturated (limited) at V_{max}. Therefore, the input voltage $V_m(t)$ is the effective voltage being applied to the SRV02 motor. The limitations of the actuator must be taken into account when designing a controller. For instance, the voltage entering the SRV02 motor should never exceed

$$V_{max} = 10.0\text{ V} \tag{2.1.24}$$

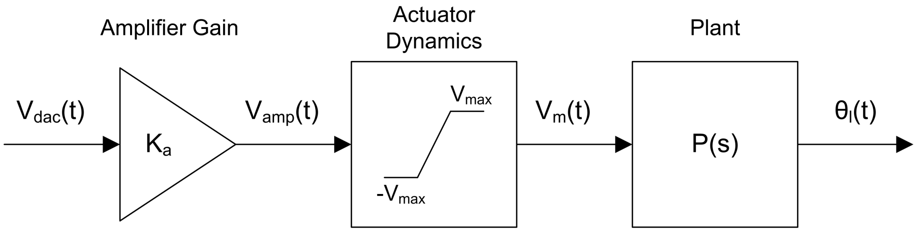


Figure 2.5: Actuator saturation.

2.1.2.3 Ramp Steady State Error Using PV Control

From our previous steady-state analysis, we found that the closed-loop SRV02 system is a Type 1 system. In this section, we will investigate the steady-state error due to a *ramp* input when using PV controller.

Given the following ramp setpoint (input)

$$R(s) = \frac{R_0}{s^2} \tag{2.1.25}$$

we can find the error transfer function by substituting the SRV02 closed-loop transfer function in equation 2.1.23 into the formula given in 2.1.11. Using the variables of the SRV02, this formula can be rewritten as $E(s) = \Theta_d(s) - \Theta_l(s)$. After rearranging the terms we find:

$$E(s) = \frac{\Theta_d(s)\, s\, (\tau\, s + 1 + K\, k_v)}{\tau\, s^2 + s + K\, k_p + K\, k_v\, s} \tag{2.1.26}$$

Substituting the input ramp transfer function 2.1.25 into the $\Theta_d(s)$ variable gives

$$E(s) = \frac{R_0\, (\tau\, s + 1 + K\, k_v)}{s\, (\tau\, s^2 + s + K\, k_p + K\, k_v\, s)} \tag{2.1.27}$$

2.1.3 PIV Controller

Adding an integral control can help eliminate any steady-state error. We will add an integral signal (middle branch in Figure 2.6) to have a proportional-integral-velocity (PIV) algorithm to control the position of the SRV02. The motor voltage will be generated by the PIV according to:

$$V_m(t) = k_p\, (\theta_d(d) - \theta_l(t)) + k_i \int (\theta_d(t) - \theta_l(t))\, dt - k_v \left(\frac{d}{dt}\, \theta_l(t) \right) \tag{2.1.28}$$

where k_i is the integral gain. We need to find the closed-loop transfer function $\Theta_l(s)/\Theta_d(s)$ for the closed-loop position control of the SRV02. Taking the Laplace transform of equation 2.1.28 gives

$$V_m(s) = \left(k_p + \frac{k_i}{s} \right) (\Theta_d(s) - \Theta_l(s)) - k_v\, s\, \Theta_l(s) \tag{2.1.29}$$

From the Plant block in Figure 2.6 and equation 2.1.3, we can write

$$\frac{\Theta_l(s)}{V_m(s)} = \frac{K}{(\tau\, s + 1)\ s} \tag{2.1.30}$$

Substituting equation 2.1.29 into 2.1.30 and solving for $\Theta_l(s)/\Theta_d(s)$ gives the SRV02 position closed-loop transfer function as:

$$\frac{\Theta_l(s)}{\Theta_d(s)} = \frac{K\, (k_p\, s + k_i)}{s^3\, \tau + (1 + K\, k_v)\, s^2 + K\, k_p\, s + K\, k_i} \tag{2.1.31}$$

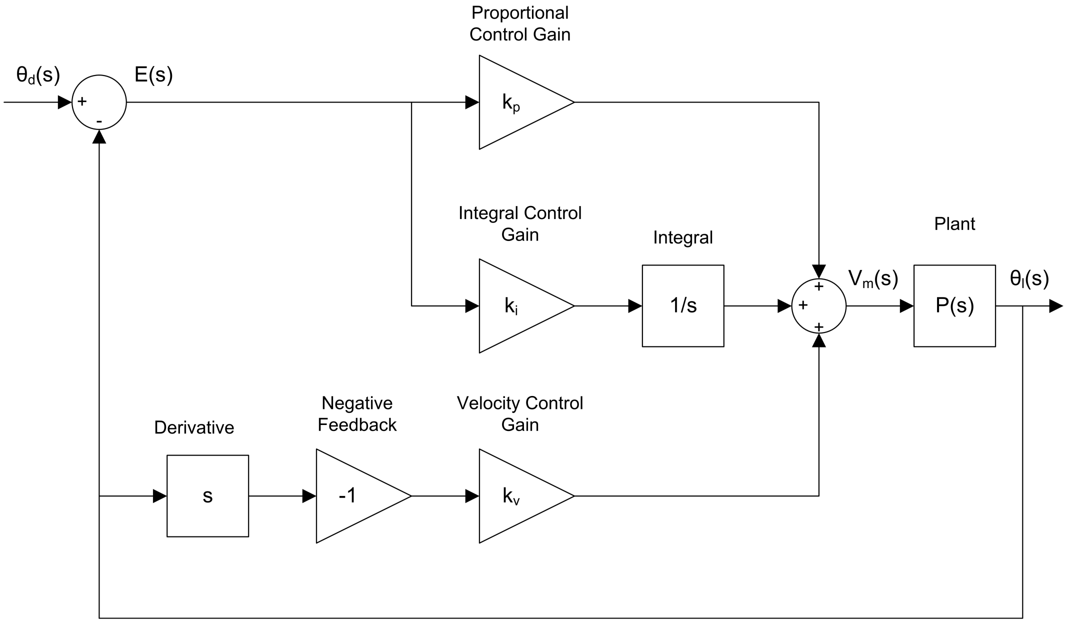


Figure 2.6: Block diagram of PIV SRV02 position control.

2.1.3.1 Ramp Steady-State Error using PIV Controller

To find the steady-state error of the SRV02 for a ramp input under the control of the PIV substitute the closed-loop transfer function from equation 2.1.31 into equation 2.1.11

$$E(s) = \frac{\Theta_d(s)\, s^2\, (\tau\, s + 1 + K\, k_v)}{s^3\, \tau + s^2 + K\, k_p\, s + K\, k_i + K\, k_v\, s^2} \tag{2.1.32}$$

Then, substituting the reference ramp transfer function 2.1.25 into the $\Theta_d(s)$ variable gives

$$E(s) = \frac{R_0\, (\tau\, s + 1 + K\, k_v)}{s^3\, \tau + s^2 + K\, k_p\, s + K\, k_i + K\, k_v\, s^2} \tag{2.1.33}$$

2.1.3.2 Integral Gain Design

It takes a certain amount of time for the output response to track the ramp reference with zero steady-state error. This is called the *settling time* and it is determined by the value used for the integral gain.

In steady-state, the ramp response error is constant. Therefore, to design an integral gain the velocity compensation (the V signal) can be neglected. Thus, we have a PI controller left as:

$$V_m(t) = k_p\, (\theta_d(t) - \theta_l(t)) + k_i \int (\theta_d(t) - \theta_l(t))\, dt \tag{2.1.34}$$

When in steady-state, the expression can be simplified to

$$V_m(t) = k_p\, e_{ss} + k_i \int_0^{t_i} e_{ss}\, dt \tag{2.1.35}$$

where the variable t_i is the integration time.

2.2 Pre-Lab Questions

Before you start the lab experiments given in Section 2.3, you should study the background materials provided in Section 2.1 and work through the questions in this Section.

1. Calculate the maximum overshoot of the response (in radians) given a step setpoint of 45 degrees and the overshoot specification given in Section 2.1.1.3.
   Hint: By substituting $y_{max} = \theta(t_p)$ and step setpoint $R_0 = \theta_d(t)$ into equation 2.1.6, we can obtain $\theta(t_p) = \theta_d(t)\left(1 + \frac{PO}{100}\right)$. Recall that the desired response specifications include 5% overshoot.
2. The SRV02 closed-loop transfer function was derived in equation 2.1.23 in Section 2.1.2.1. Find the control gains k_p and k_v in terms of ω_n and ζ. **Hint:** Remember the standard second order system equation.
3. Calculate the minimum damping ratio and natural frequency required to meet the specifications given in Section 2.1.1.3.
4. Based on the nominal SRV02 model parameters, K and τ, found in Laboratory 1: SRV02 Modeling, calculate the control gains needed to satisfy the time-domain response requirements given in Section 2.1.1.3.
5. In the PV controlled system, for a reference step of $\pi/4$ (i.e. 45 degree step) starting from $\Theta_l(t) = 0$ position, calculate the *maximum* proportional gain that would lead to providing the maximum voltage to the motor. Ignore the velocity control ($k_v = 0$). Can the desired specifications be obtained based on this maximum available gain and what you calculated in question 4?
6. For the PV controlled closed-loop system, find the steady-state error and evaluate it numerically given a ramp with a slope of $R_0 = 3.36$ rad/s. Use the control gains found in question 4.
7. What should be the integral gain k_i so that when the SRV02 is supplied with the maximum voltage of $V_{max} = 10\,V$ it can eliminate the steady-state error calculated in question 6 in 1 second? **Hint:** Start from equation 2.1.35 and use $t_i = 1$, $V_m(t) = 10$, the k_p you found in question 4 and e_{ss} found in question 6. Remember that e_{ss} is constant.

2.3 Lab Experiments

The main goal of this laboratory is to explore position control of the SRV02 load shaft using PV and PIV controllers.

In this laboratory, you will conduct three experiments:

1. Step response with PV controller,
2. Ramp response with PV controller, and
3. Ramp response with no steady-state error.

You will need to design the third experiment yourself. In each experiment, you will first simulate the closed-loop response of the system. Then, you will implement the controller using the SRV02 hardware and software to compare the real response to the simulated one.

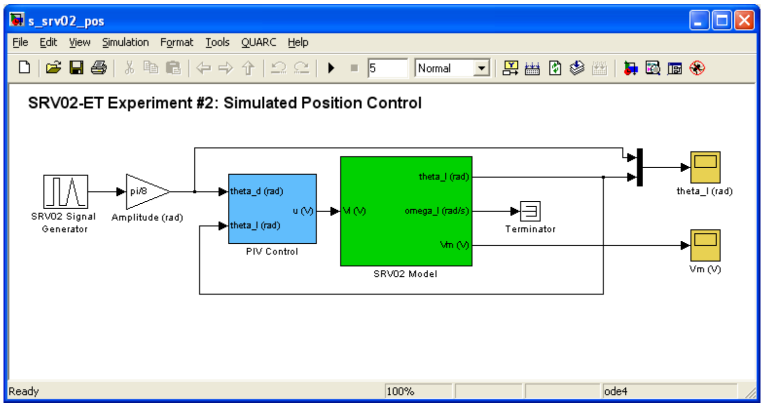
2.3.1 Step Response Using PV Controller

2.3.1.1 Simulation

First, you will simulate the closed-loop response of the SRV02 with a PV controller to step input. Our goals are to confirm that the desired reponse specifications in an ideal situation are satisfied and to verify that the motor is not saturated. Then, you will explore the effect of using a high-pass filter, instead of a direct derivative, to create the velocity signal V in the controller.

Experimental Setup

The *s_srv02_pos* Simulink®diagram shown in Figure 2.7 will be used to simulate the closed-loop position control response with the PV and PIV controllers. The SRV02 Model uses a *Transfer Fcn* block from the Simulink®library. The PIV Control subsystem contains the PIV controller detailed in Section 2.1.3. When the integral gain is set to zero, it essentially becomes a PV controller.



SRV02-ET Experiment #2: Simulated Position Control

Figure 2.7: Simulink model used to simulate the SRV02 closed-loop position response.

IMPORTANT: Before you can conduct these experiments, you need to make sure that the lab files are configured according to your SRV02 setup. If they have not been configured already, then you need to go to Section 2.4.2 to configure the lab files first.

Closed-loop Response with the PV Controller

1. Enter the proportional and velocity control gains found in Pre-Lab question 4 in Matlab®as *kp* and *kv*.
2. To generate a step reference, ensure the *SRV02 Signal Generator* is set to the following:
   - Signal type = *square*
   - Amplitude = 1
   - Frequency = 0.4 Hz
3. In the Simulink®diagram, set the *Amplitude (rad)* gain block to $\pi/8(rad)$ to generate a step with an amplitude of 45 degrees (i.e., square wave goes between $\pm\pi/8$ which results in a step amplitude of $\pi/4$) .
4. Inside the *PIV Control* subsystem, set the *Manual Switch* to the upward position so the *Derivative block* is used.
5. Open the load shaft position scope, *theta_l (rad)*, and the motor input voltage scope, *Vm (V)*.
6. Start the simulation. By default, the simulation runs for 5 seconds. The scopes should be displaying responses similar to figures 2.8 and 2.9 Note that in the *theta_l* (rad) scope, the yellow trace is the setpoint position while the purple trace is the simulated position (generated by the *SRV02 Model* block). This simulation is called the *Ideal PV* response as it uses the PV compensator with the derivative block.

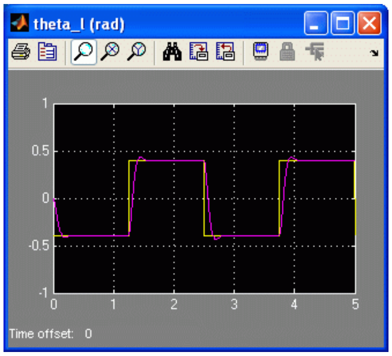


Figure 2.8: Ideal PV position response.

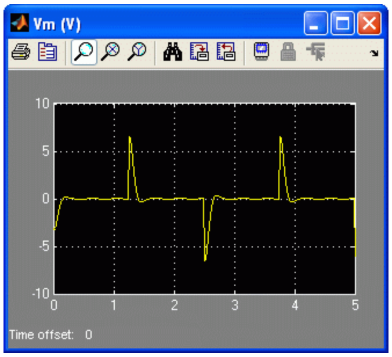


Figure 2.9: Ideal PV motor input voltage.

7. Generate a Matlab®figure showing the *Ideal PV* position response and the ideal input voltage. After each simulation run, each scope automatically saves their response to a variable in the Matlab®workspace. That is, the *theta_l (rad)* scope saves its response to the variable called data_pos and the *Vm (V)* scope saves its data to the *data_vm* variable. The *data_pos* variable has the following structure: *data_pos(:,1)* is the time vector, *data_pos(:,2)* is the setpoint, and *data_pos(:,3)* is the simulated angle. For the *data_vm* variable, *data_vm(:,1)* is the time and *data_vm(:,2)* is the simulated input voltage.
8. Measure the steady-state error, the percent overshoot and the peak time of the simulated response. Does the response satisfy the specifications given in Section 2.1.1.3? **Hint:** Use the Matlab®*ginput* command to take measurements off the figure.

   Using a High-pass Filter Instead of Direct Derivative
9. When implementing a controller on actual hardware, it is generally not advised to take the direct derivative of a measured signal. Any noise or spikes in the signal becomes amplified and gets multiplied by a gain and fed into the motor which may lead to damage. To remove any high-frequency noise components in the velocity signal, a low-pass filter is placed in series with the derivative, i.e. taking the high-pass filter of the measured signal. However, as with a controller, the filter must also be tuned properly. In addition, the filter has some adverse affects. Go in the *PIV Control* block and set the Manual Switch block to the down position to enable the high-pass filter.
10. Start the simulation. The response in the scopes should still be similar to figures 2.8 and 2.9. This simulation is called the *Filtered PV* response as it uses the PV controller with the high-pass filter block.

11. Generate a Matlab®figure showing the *Filtered PV* position and input voltage responses.

12. Measure the steady-state error, peak time, and percent overshoot. Are the specifications still satisfied without saturating the actuator? Recall that the peak time and percent overshoot should not exceed the values given in Section 2.1.1.3. Discuss the changes from the ideal response. **Hint:** The different in the response is minor. Make sure you use *ginput* to take precise measurements.

2.3.1.2 Implementing Step Response using PV Controller

In this experiment, we will control the angular position of the SRV02 load shaft, i.e. the disc load, using the PV controller. Measurements will then be taken to ensure that the specifications are satisfied.

Experimental Setup

The *q_srv02_pos* Simulink®diagram shown in Figure 2.10 is used to implement the position control experiments. The *SRV02-ET* subsystem contains QUARC blocks that interface with the DC motor and sensors of the SRV02 system, as discussed in Section A. The *PIV Control* subsystem implements the PIV controller detailed in Section 2.1.3, except a high-pass filter is used to obtain the velocity signal (as opposed to taking the direct derivative).

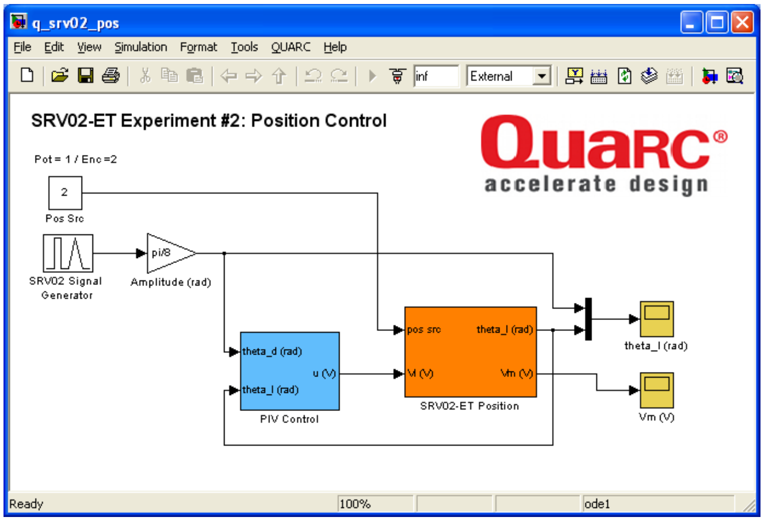


Figure 2.10: Simulink model used with QUARC to run the PV and PIV position controllers on the SRV02.

IMPORTANT: Before you can conduct these experiments, you need to make sure that the lab files are configured according to your SRV02 setup. If they have not been configured already, then you need to go to Section 2.4.3 to configure the lab files first.

1. Run the *setup_srv02_exp02_pos.m* script.

2. Enter the proportional and velocity control gains found in Pre-Lab question 4.

3. Set Signal Type in the SRV02 Signal Generator to *square* to generate a step reference.

4. Set the *Amplitude (rad)* gain block to $\pi/8$ to generate a step with an amplitude of 45 degrees.

5. Open the load shaft position scope, *theta_l (rad)*, and the motor input voltage scope, *Vm (V)*.

6. Click on QUARC | Build to compile the Simulink®diagram.

7. Select QUARC | Start to begin running the controller. The scopes should display responses similar to figures 2.11 and 2.12 Note that in the theta_l (rad) scope, the yellow trace is the setpoint position while the purple trace is the measured position.

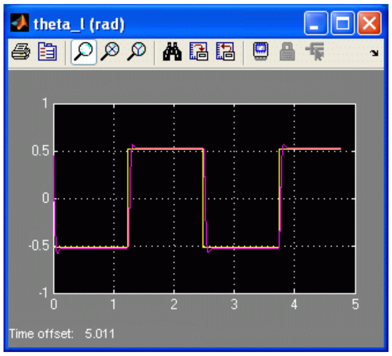


Figure 2.11: Measured PV step response.

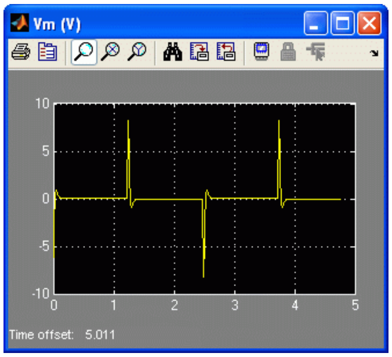


Figure 2.12: PV control input voltage.

8. When a suitable response is obtained, click on the Stop button in the Simulink® diagram toolbar (or select QUARC | Stop from the menu) to stop running the code. Generate a Matlab®figure showing the PV position response and its input voltage.

   As in the *s_srv02_pos* Simulink diagram, when the controller is stopped each scope automatically saves their response to a variable in the Matlab®workspace. Thus the *theta_l (rad)* scope saves its response to the *data_pos* variable and the *Vm (V)* scope saves its data to the *data_vm* variable.

9. Measure the steady-state error, the percent overshoot, and the peak time of the SRV02 load gear. Does the response satisfy the specifications given in Section 2.1.1.3?

10. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.

11. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

2.3.2 Ramp Response Using PV Controller

2.3.2.1 Simulation

In this simulation, the goal is to verify that the system with the PV controller can meet the zero steady-state error specification without saturating the motor.

As in the Step Response experiment in Section 2.3.1, in this experiment you need to use the *s_srv02_pos* Simulink®diagram shown in Figure 2.7 in Section 2.3.1.1 again.

1. Enter the proportional and velocity control gains found in Pre-Lab question 4.

2. Set the *SRV02 Signal Generator* parameters to the following to generate a triangular reference (which corresponds to a ramp input):

   - Signal Type = triangle
   - Amplitude = 1
   - Frequency = 0.8 Hz

3. Setting the frequency to 0.8 Hz will generate an increasing and decreasing ramp signal with the same slope used in the Pre-Lab question 6. The slope is calculated from the *Triangular Waveform* amplitude, *Amp*, and frequency, *f*, using the expression.

$$R_0 = 4\,Amp\,f \tag{2.3.36}$$

4. In the Simulink®diagram, set the *Amplitude (rad)* gain block to $\pi/3$.

5. Inside the *PIV Control* subsystem, set the *Manual Switch* to the down position so that the *High-Pass Filter* block is used.
6. Open the load shaft position scope, *theta_l (rad)*, and the motor input voltage scope, *Vm (V)*.
7. Start the simulation. The scopes should display responses similar to figures 2.13 and 2.14.

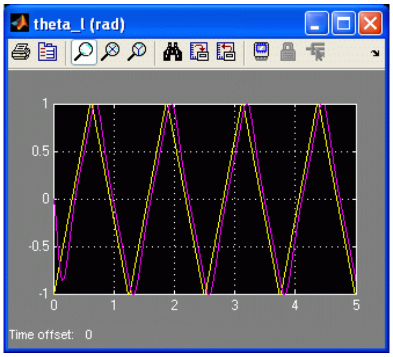


Figure 2.13: Ramp response using PV.

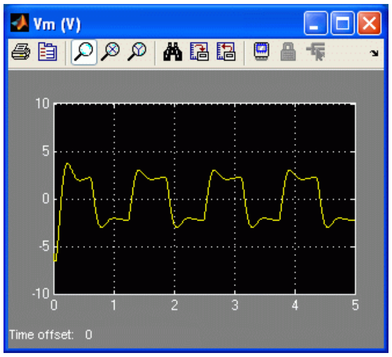


Figure 2.14: Input voltage of ramp tracking using PV.

8. Generate a Matlab®figure showing the *Ramp PV* position response and its corresponding input voltage trace.
9. Measure the steady-state error. Compare the simulation measurement with the steady-state error calculated in Pre-Lab question 6.

2.3.2.2 Implementing Ramp Response Using PV

In this experiment, we will control the angular position of the SRV02 load shaft, i.e. the disc load, using a PV controller. The goal is to examine how well the system can track a triangular (ramp) position input. Measurements will then be taken to ensure that the specifications are satisfied.

As in the Step Response experiment in Section 2.3.1, in this experiment you also need to use the *q_srv02_pos* Simulink®diagram shown in Figure 2.10 to implement the position control experiments.

1. Run the *setup_srv02_exp02_pos.m* script.
2. Enter the proportional and velocity control gains found in Pre-Lab question 4.
3. Set the *SRV02 Signal Generator* parameters to the following to generate a triangular reference (i.e., ramp reference):
   - Signal Type = triangle
   - Amplitude = 1
   - Frequency = 0.8 Hz
4. In the Simulink®diagra, set the *Amplitude (rad)* gain block to $\pi/3$.
5. Open the load shaft position scope, *theta_l (rad)*, and the motor input voltage scope, *Vm (V)*.
6. Click on QUARC | Build to compile the Simulink®diagram.
7. Select QUARC | Start to run the controller. The scopes should display responses similar to figures 2.15 and 2.16.
8. Generate a Matlab®figure showing the *Ramp PV* position response and its corresponding input voltage trace.
9. Measure the steady-state error and compare it with the steady-state error calculated in Pre-Lab question 6.

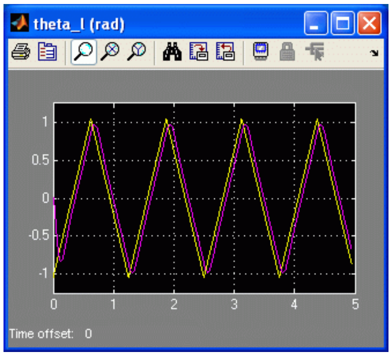


Figure 2.15: Measured SRV02 PV ramp response.

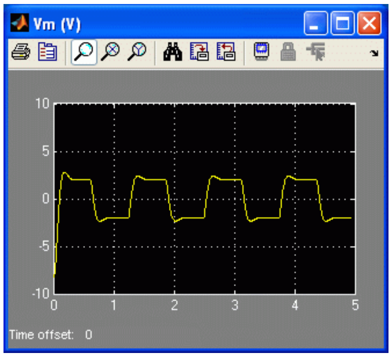


Figure 2.16: Input voltage of PV ramp response.

2.3.3 Ramp Response with No Steady-State Error

Design an experiment to see if the steady-state error can be eliminated when tracking a ramp input. First simulate the response, then implement it using the SRV02 system.

1. How can the PV controller be modified to eliminate the steady-state error in the ramp response? State your hypothesis and describe the anticipated cause-and-effect leading to the expected result. **Hint:** Look through Section 2.
2. List the independent and dependent variables of your proposed controller. Explain their relationship.
3. Your proposed control, like the PV compensator, are model-based controllers. This means that the control gains generated are based on mathematical representation of the system. Given this, list the assumptions you are making in this control design. State the reasons for your assumptions.
4. Give a brief, general overview of the steps involved in your experimental procedure for two cases: (1) Simulation, and (2) Implementation.
5. For each case, generate a Matlab®figure showing the position response of the system and its corresponding input voltage.
6. In each case, measure the steady-state error.
7. For each case comment on whether the steady-state specification given in Section 2.1.1.3 was satisfied without saturating the actuator.
8. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.
9. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

2.3.4 Results

Fill out Table 2.1 below with your answers to the Pre-Lab questions and your results from the lab experiments.

Section / Question	Description	Symbol	Value	Unit
Question 4	**Pre-Lab: Model Parameters** Open-Loop Steady-State Gain Open-Loop Time Constant	 K τ		
Question 4	**Pre-Lab: PV Gain Design** Proportional gain Velocity gain	 k_p k_v		
Question 5	**Pre-Lab: Control Gain Limits** Maximum proportional gain	 $k_{p,max}$		
Question 6	**Pre-Lab: Ramp Steady-State Error** Steady-state error using PV	 e_{ss}		
Question 7	**Pre-Lab: Integral Gain Design** Integral gain	 k_i		
2.3.1.1	**Step Response Simulation** Peak time Percent overshoot Steady-state error	 t_p PO e_{ss}		
2.3.1.1	**Filtered Step Response Using PV** Peak time Percent overshoot Steady-state error	 t_p PO e_{ss}		
2.3.1.2	**Step Response Implementation** Peak time Percent overshoot Steady-state error	 t_p PO e_{ss}		
2.3.2.1	**Ramp Response Simulation with PV** Steady-state error	 e_{ss}		
2.3.2.2	**Ramp Response Implementation with PV** Steady-state error	 e_{ss}		
2.3.3	**Ramp Response Simulation with with no steady-state error** Steady-state error	 e_{ss}		
2.3.3	**Ramp Response Implementation with with no steady-state error** Steady-state error	 e_{ss}		

Table 2.1: Summary of results for the SRV02 Position Control laboratory.

2.4 System Requirements

Before you begin this laboratory make sure:

- QUARC®is installed on your PC, as described in Reference [1].
- You have a QUARC compatible data-aquisition (DAQ) card installed in your PC. For a listing of compliant DAQ cards, see Reference [5].
- SRV02 and amplifier are connected to your DAQ board as described Reference [6].

2.4.1 Overview of Files

Table 2.2: Files supplied with the SRV02 Position Control laboratory.

File Name	Description
02 - SRV02 Position Control - Student Manual.pdf	This laboratory guide contains pre-lab questions and lab experiments demonstrating how to design and implement a position controller on the Quanser SRV02 rotary plant using QUARC.
setup_srv02_exp02_pos.m	The main Matlab®script that sets the SRV02 motor and sensor parameters as well as its configuration-dependent model parameters. **Run this file only to setup the laboratory.**
config_srv02.m	Returns the configuration-based SRV02 model specifications *Rm, kt, km, Kg, eta_g, Beq, Jeq,* and *eta_m,* the sensor calibration constants K_POT, K_ENC, and K_TACH, and the amplifier limits VMAX_AMP and IMAX_AMP.
d_model_param.m	Calculates the SRV02 model parameters K and tau based on the device specifications *Rm, kt, km, Kg, eta_g, Beq, Jeq,* and *eta_m.*
calc_conversion_constants.m	Returns various conversions factors.
s_srv02_pos.mdl	Simulink file that simulates a closed-loop PIV controller for the SRV02 system.
q_srv02_pos.mdl	Simulink file that implements a closed-loop PIV position controller on the SRV02 system using QUARC®.

2.4.2 Setup for Position Control Simulations

Follow these steps to configure the lab properly:

1. Load the Matlab®software.
2. Browse through the Current Directory window in Matlab®and find the folder that contains the SRV02 position controller files, e.g. *q_srv02_pos.mdl*.
3. Double-click on the *s_srv02_pos.mdl* file to open the Simulink®diagram shown in Figure 2.7.
4. Double-click on the *setup_srv02_exp02_spd.m* file to open the setup script for the position control Simulink models.

5. **Configure setup script:** The controllers will be run on an SRV02 in the high-gear configuration with the disc load. In order to simulate the SRV02 properly, make sure the script is setup to match this configuration, i.e. the EXT_GEAR_CONFIG should be set to 'HIGH' and the LOAD_TYPE should be set to 'DISC'. Also, ensure the ENCODER_TYPE, TACH_OPTION, K_CABLE, AMP_TYPE, and VMAX_DAC parameters are set according to the SRV02 system that is to be used in the laboratory.

6. Finally, make sure CONTROL_TYPE is set to 'MANUAL'.

7. Run the script by selecting the Debug | Run item from the menu bar or clicking on the Run button in the tool bar. The messages shown below, should be generated in the Matlab® Command Window. The model parameters and specifications are loaded but the PIV gains are all set to zero - they need to be changed.

```
SRV02 model parameters:
   K = 1.53 rad/s/V
   tau = 0.0254 s
Specifications:
   tp = 0.2 s
   PO = 5 %
Calculated PV control gains:
   kp = 0 V/rad
   kv = 0 V.s/rad
Integral control gain for triangle tracking:
   ki = 0 V/rad/s
```

2.4.3 Setup for Position Control Implementation

Before beginning the lab experiments on the SRV02 device, the q_srv02_pos Simulink® diagram and the setup_srv02_exp02_pos script must be configured.

Follow these steps to get the system ready for this lab:

1. Setup the SRV02 in the high-gear configuration and with the disc load as described in Reference [6].
2. Load the Matlab® software.
3. Browse through the *Current Directory* window in Matlab® and find the folder that contains the SRV02 position control files, e.g. *q_srv02_pos.mdl*.
4. Double-click on the *q_srv02_pos.mdl* file to open the Position Control Simulink® diagram shown in Figure 2.7.
5. **Configure DAQ:** Double-click on the HIL Initialize block in the *SRV02-ET* subsystem (which is located inside the *SRV02-ET* Position subsystem) and ensure it is configured for the DAQ device that is installed in your system. See Section A for more information on configuring the HIL Initialize block.
6. **Configure Sensor:** The position of the load shaft can be measured using various sensors. Set the Pos Src Source block in q_srv02_pos, as shown in Figure 2.7, as follows:

   - 1 to use the potentiometer
   - 2 to use to the encoder

   Note that when using the potentiometer, there will be a discontinuity.
7. Configure setup script: Set the parameters in the setup_srv02_exp02_pos.m script according to your system setup. See Section 2.4.2 for more details.

2.5 Lab Report

This laboratory contains three experiments, namely,

1. step response,
2. ramp response with PV controller, and
3. ramp response with no steady-state error.

When you are writing your lab report, follow the outline corresponding to the experiment you conducted to build the *content* of your report. Also, in Section 2.5.4 you can find some basic tips for the *format* of your report.

2.5.1 Template for Content (Step Response Experiment)

I. PROCEDURE

1. *Closed-loop response with the PV controller*
   - Briefly describe the main goal of the simulation.
   - Briefly describe the simulation procedure (Section 2.3.1.1)
2. *Step response with PV controller using high-pass filter*
   - Briefly describe the main goal of this simulation.
   - Briefly describe the simulation procedure (Section 2.3.1.1)
3. *Implementing Step Response using PV Controller*
   - Briefly describe the main goal of this experiment.
   - Briefly describe the experimental procedure (Section 2.3.1.2)

II. RESULTS
Do not interpret or analyze the data in this section. Just provide the results.

1. Response plot from step 7 in Section 2.3.1.1, *Simulated step response*
2. Response plot from step 11 in Section 2.3.1.1, *Filtered PV response*
3. Response plot from step 8 in Section 2.3.1.2, *Step response of implemented PV controller*
4. Provide applicable data collected in this laboratory (from Table 2.1).

III. ANALYSIS
Provide details of your calculations (methods used) for analysis for each of the following:

1. Step 8 in Section 2.3.1.1, *Step response with PV controller*
2. Step 12 in Section 2.3.1.1, *Step response with PV controller using high-pass filter*
3. Step 9 in Section 2.3.1.2, *Step response with PV controller*

IV. CONCLUSIONS
Interpret your results to arrive at logical conclusions for the following:

1. Step 8 in Section 2.3.1.1, *Step response simulation with PV controller*
2. Step 12 in Section 2.3.1.1, *Step response simulation with PV controller using High-pass filter*
3. Step 9 in Section 2.3.1.2, *Step response with the implemented PV controller*

2.5.2 Template for Content (Ramp Response with PV)

I. PROCEDURE

1. *Ramp response with PV controller*
   - Briefly describe the main goal of this simulation.
   - Briefly describe the procedure (Section 2.3.2.1)
2. *Implementing Ramp Response Using PV*
   - Briefly describe the main goal of this experiment.
   - Briefly describe the experimental procedure (Section 2.3.2.2)

II. RESULTS
Do not interpret or analyze the data in this section. Just provide the results.

1. Response plot from step 8 in Section 2.3.2.1, *Simulated PV controller with ramp input*
2. Response plot from step 8 in Section 2.3.2.2, *Ramp response of implemented PV controller*
3. Provide applicable data collected in this laboratory (from Table 2.1).

III. ANALYSIS
Provide details of your calculations (methods used) for analysis for each of the following:

1. Step 9 in Section 2.3.2.1, *Simulated PV controller with ramp input*
2. Step 9 in Section 2.3.2.2, *Ramp response of implemented PV controller*

IV. CONCLUSIONS
Interpret your results to arrive at logical conclusions for the following:

1. Step 9 in Section 2.3.2.2, *Ramp response with the implemented PV controller*

2.5.3 Template for Content (Ramp Response with No Steady-State Error)

I. PROCEDURE

1. State the hypothesis of your experiment and describe the anticipated cause-and-effect leading to the expected result (Step 1 in Section 2.3.3).
2. List the independent and dependent variables for the controller. Explain their relationship (Step 2 in Section 2.3.3).
3. List the assumptions you made in this experiment. State the reasons for your assumptions (Step 3 in Section 2.3.3).
4. Briefly list the steps of your experimental procedure for two cases: (1) Simulation, and (2) Implementation (Step 4 in Section 2.3.3).

II. RESULTS
Do not interpret or analyze the data in this section. Just provide the results.

1. Response plot from step 5 in Section 2.3.3, *Simulated controller with ramp input*
2. Response plot from step 5 in Section 2.3.3, *Implemented controller with ramp input*
3. Provide applicable data collected in this laboratory (from Table 2.1).

III. ANALYSIS
Provide details of your calculations (methods used) for analysis for each of the following:

1. Step 6 in Section 2.3.3, *Simulated controller with ramp input*
2. Step 6 in Section 2.3.3, *Implemented controller with ramp input*

IV. CONCLUSIONS
Interpret your results to arrive at logical conclusions for the following:

1. Step 7 in Section 2.3.3, *for both simulated and implemented controllers*

2.5.4 Tips for Report Format

PROFESSIONAL APPEARANCE

- Has cover page with all necessary details (title, course, student name(s), etc.)
- Each of the required sections is completed (Procedure, Results, Analysis and Conclusions).
- Typed.
- All grammar/spelling correct.
- Report layout is neat.
- Does not exceed specified maximum page limit, if any.
- Pages are numbered.
- Equations are consecutively numbered.
- Figures are numbered, axes have labels, each figure has a descriptive caption.
- Tables are numbered, they include labels, each table has a descriptive caption.
- Data are presented in a useful format (graphs, numerical, table, charts, diagrams).
- No hand drawn sketches/diagrams.
- References are cited using correct format.

LABORATORY 3

SRV02 SPEED CONTROL

The objective of this laboratory is to develop feedback systems that control the speed of the rotary servo load shaft. A proportional-integral (PI) controller and a lead compensator are designed to regulate the shaft speed according to a set of specifications.

Topics Covered

- Design of a proportional-integral (PI) controller that regulates the angular speed of the servo load shaft.
- Design of a lead compensator.
- Simulation of the PI and lead controllers using the plant model to ensure the specifications are met without any actuator saturation.
- Implemention of the controllers on the Quanser SRV02 device to evaluate their performance.

Prerequisites

In order to successfully carry out this laboratory, the user should be familiar with the following:

- Data acquisition device (e.g. Q2-USB), the power amplifier (e.g. VoltPAQ-X1), and the main components of the SRV02 (e.g. actuator, sensors), as described in References [2], [4], and [6], respectively.
- Wiring and operating procedure of the SRV02 plant with the amplifier and data-aquisition (DAQ) device, as discussed in Reference [6].
- Transfer function fundamentals, e.g. obtaining a transfer function from a differential equation.
- Laboratory described in Appendix A to get familiar with using QUARC®with the SRV02.

3.1 Background

3.1.1 Desired Response

3.1.1.1 SRV02 Speed Control Specifications

The time-domain requirements for controlling the speed of the SRV02 load shaft are:

$$e_{ss} = 0 \tag{3.1.1}$$

$$t_p \leq 0.05 \text{ s, and} \tag{3.1.2}$$

$$PO \leq 5 \text{ \%} \tag{3.1.3}$$

Thus, when tracking the load shaft reference, the transient response should have a peak time less than or equal to 0.05 seconds, an overshoot less than or equal to 5 %, and zero steady-state error.

In addition to the above time-based specifications, the following frequency-domain requirements are to be met when designing the *Lead Compensator*:

$$PM \geq 75.0 \text{ deg} \tag{3.1.4}$$

and

$$\omega_g = 75.0 \text{ rad/s} \tag{3.1.5}$$

The phase margin mainly affects the shape of the response. Having a higher phase margin implies that the system is more stable and the corresponding time response will have less overshoot. The overshoot will not go beyond 5% with a phase margin of at least 75.0 degrees.

The crossover frequency is the frequency where the gain of the Bode plot is 1 (or 0 dB). This parameter mainly affects the speed of the response, thus having a larger ω_g decreases the peak time. With a crossover frequency of 75.0 radians the resulting peak time will be less than or equal to 0.05 seconds.

3.1.1.2 Overshoot

In this laboratory we will use the following step setpoint (input):

$$\omega_d(t) = \begin{cases} 2.5 \text{ rad/s} & t \leq t_0 \\ 7.5 \text{ rad/s} & t > t_0 \end{cases} \tag{3.1.6}$$

where t_0 is the time the step is applied. Initially, the SRV02 should be running at 2.5 rad/s and after the step time it should jump up to 7.5 rad/s. From the standard definition of overshoot in step response, we can calculate the maximum overshoot of the response (in radians):

$$\omega(t_p) = \omega_d(t_0) + (\omega_d(t) - \omega_d(t_0))\left(1 + \frac{PO}{100}\right) \tag{3.1.7}$$

with the given values the maximum overshoot of the response is

$$\omega(t_p) = 7.75 \text{ rad/s} \tag{3.1.8}$$

The closed-loop speed response should therefore not exceed the value given in Equation 3.1.8.

3.1.1.3 Steady State Error

Consider the speed control system with unity feedback shown in Figure 3.1. Let the compensator be $C(s) = 1$.

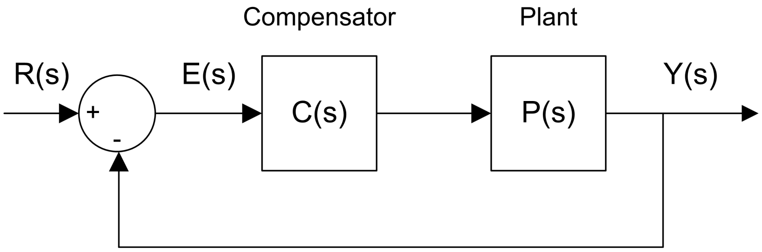


Figure 3.1: Unity feedback loop.

We can find the steady-state error using the final value theorem:

$$e_{ss} = \lim_{s\to 0} s\,E(s) \tag{3.1.9}$$

where

$$E(s) = \frac{R(s)}{1 + C(s)\,P(s)} \tag{3.1.10}$$

The voltage-to-speed transfer function for the SRV02 was found in Section 1 as:

$$P(s) = \frac{K}{\tau\,s + 1} \tag{3.1.11}$$

Substituting $R(s) = \frac{R_0}{s}$ and $C(s) = 1$ gives:

$$E(s) = \frac{R_0}{s\left(1 + \frac{K}{\tau\,s+1}\right)} \tag{3.1.12}$$

Applying the final-value theorem to the system gives

$$e_{ss} = R_0 \left(\lim_{s\to 0} \frac{\tau\,s + 1}{\tau\,s + 1 + K}\right) \tag{3.1.13}$$

When evaluated, the resulting steady-state error due to a step response is

$$e_{ss} = \frac{R_0}{1 + K} \tag{3.1.14}$$

3.1.2 PI Control Design

3.1.2.1 Closed Loop Transfer Function

The proportional-integral (PI) compensator used to control the velocity of the SRV02 has the following structure:

$$V_m(t) = k_p\,(b_{sp}\,\omega_d(t) - \omega_l(t)) - k_i \int (\omega_d(t) - \omega_l(t))\,dt \tag{3.1.15}$$

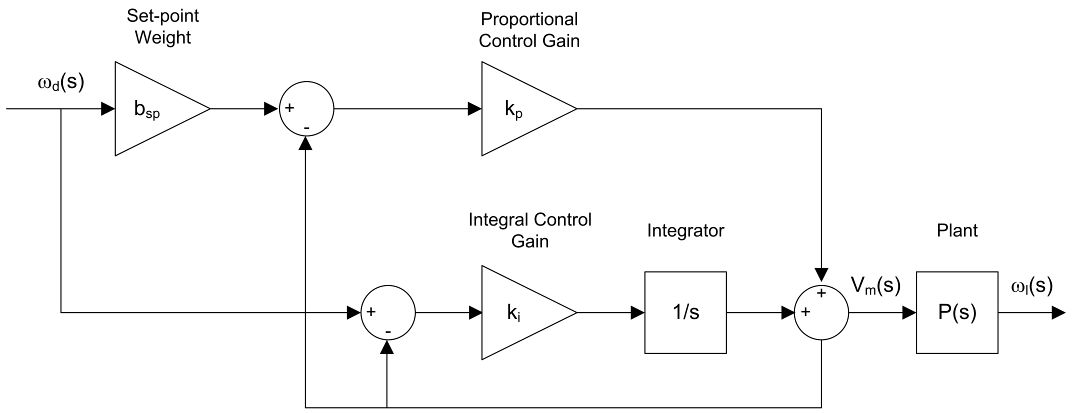


Figure 3.2: Block diagram of SRV02 PI speed control.

where k_p is the proportional control gain, k_i is the integral control gain, $\omega_d(t)$ is the setpoint or reference angular speed for the load shaft, $\omega_l(t)$ is the measured load shaft angular speed, b_{sp} is the setpoint weight, and $V_m(t)$ is the voltage applied to the SRV02 motor. The block diagram of the PI control is given in Figure 3.2.

We can take Laplace transform of the controller given in Equation 3.1.15:

$$V_m(s) = k_p\,(b_{sp}\,\Omega_d(s) - \Omega_l(s)) + \frac{k_i\,(\Omega_d(s) - \Omega_l(s))}{s} \tag{3.1.16}$$

To find the closed-loop speed transfer function, $\Omega_l(s)/\Omega_d(s)$, we can use the process transfer function from Equation 3.1.11 and solve for $\Omega_l(s)/\Omega_d(s)$ as:

$$\frac{\Omega_l(s)}{\Omega_d(s)} = \frac{K\,(k_p\,s\,b_{sp} + k_i)}{s^2\,\tau + (1 + K\,k_p)\,s + K\,k_i} \tag{3.1.17}$$

3.1.2.2 Finding PI Gains to Satisfy Specifications

In this section, we will first calculate the minimum damping ratio and natural frequency required to meet the specifications given in Section 3.1.1.1. Then, using these values we will calculate the necessary control gains k_p and k_i to achieve the desired performance with a PI controller.

The minimum damping ratio and natural frequency needed to satisfy a given percent overshoot and peak time are:

$$\zeta = -\ln\left(\frac{PO}{100}\right)\sqrt{\frac{1}{\ln\left(\frac{PO}{100}\right)^2 + \pi^2}} \tag{3.1.18}$$

and

$$\omega_n = \frac{\pi}{t_p\,\sqrt{1-\zeta^2}} \tag{3.1.19}$$

Substituting the percent overshoot specifications given in 3.1.3 into Equation 3.1.18 gives the required damping ratio

$$\zeta = 0.690 \tag{3.1.20}$$

Then, by substituting this damping ratio and the desired peak time, given in 3.1.2, into Equation 3.1.19, the minimum natural frequency is found as:

$$\omega_n = 86.7 \text{ rad/s} \tag{3.1.21}$$

Now, let's look at how we can calculate the gains. When the setpoint weight is zero, i.e. $b_{sp} = 0$, the closed-loop SRV02 speed transfer function has the structure of a *standard second-order system*. We can find expressions for the control gains k_p and k_i by equating the characteristic equation (denominator) of the SRV02 closed-loop transfer function to the *standard characteristic equation*: $s^2 + 2\,\zeta\,\omega_n\,s + \omega_n^2$.

The denominator of the transfer function can be re-structured into the following:

$$s^2 + \frac{(1 + K\,k_p)\;s}{\tau} + \frac{K\,k_i}{\tau} \tag{3.1.22}$$

equating the coefficients of this equation to the coefficients of the standard characteristic equation gives:

$$\frac{K\,k_i}{\tau} = \omega_n^2 \tag{3.1.23}$$

and

$$\frac{1 + K\,k_p}{\tau} = 2\,\zeta\,\omega_n \tag{3.1.24}$$

Then, the proportional gain k_p can be found as:

$$k_p = \frac{-1 + 2\,\zeta\,\omega_n\,\tau}{K} \tag{3.1.25}$$

and the integral gain k_i is

$$k_i = \frac{\omega_n^2\,\tau}{K} \tag{3.1.26}$$

3.1.3 Lead Control Design

Alternatively, a lead or lag compensator can be designed to control the speed of the servo. The lag compensator is actually an approximation of a PI control and this, at first, may seem like the more viable option. However, due to the saturation limits of the actuator the lag compensator cannot achieve the desired zero steady-state error specification. Instead, a lead compensator with an integrator, as shown in Figure 3.3, will be designed.

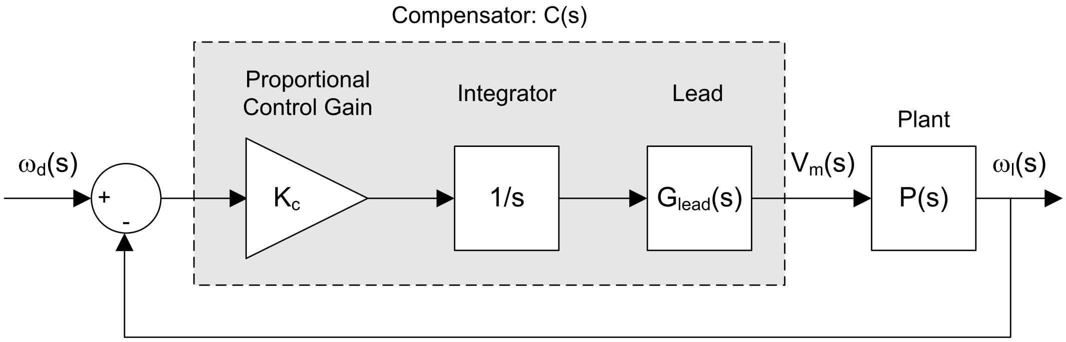


Figure 3.3: Closed-loop SRV02 speed control with lead compensator.

To obtain zero steady-state error, an integrator is placed in series with the plant. This system is denoted by the transfer function

$$P_i(s) = \frac{P(s)}{s} \tag{3.1.27}$$

where $P(s)$ is the plant transfer function in Equation 3.1.11.

The phase margin and crossover frequency specifications listed in equations 3.1.4 and 3.1.5 of Section 3.1.1.1 can then be satisfied using a proportional gain K_c and the lead transfer function

$$G_{lead}(s) = \frac{1 + a\,T\,s}{1 + T\,s} \tag{3.1.28}$$

The a and T parameters change the location of the pole and the zero of the lead compensator which changes the gain and phase margins of the system. The design process involves examining the stability margins of the *loop transfer function*, $L(s) = C(s) \cdot P(s)$, where the compensator is given by:

$$C(s) = \frac{K_c\;(1 + a\,T\,s)}{(1 + T\,s)\;s} \tag{3.1.29}$$

3.1.3.1 Finding Lead Compensator Parameters

The Lead compensator is an approximation of a proportional-derivative (PD) control. A PD controller can be used to add damping to reduce the overshoot in the transient of a step response and effectively making the system more stable. In other words, it increases the phase margin. In this particular case, the lead compensator is designed for the following system:

$$L_p(s) = \frac{K_c\, P(s)}{s} \tag{3.1.30}$$

The proportional gain Kc is designed to attain a certain crossover frequency. Increasing the gain crossover frequency essentially increases the bandwidth of the system which decreases the peak time in the transient response (i.e. makes the response faster). However, as will be shown, adding a gain $K_c > 1$ makes the system less stable. The phase margin of the $L_p(s)$ system is therefore lower than the phase margin of the $P_i(s)$ system and this translates to having a large overshoot in the response. The lead compensator is used to dampen the overshoot and increase the overall stability of the system, i.e increase its phase margin.

The frequency response of the lead compensator given in 3.1.28 is

$$G_{lead}\,(\omega\, j) = \frac{1 + a\, T\, \omega j}{1 + T\, \omega\, j} \tag{3.1.31}$$

and its corresponding magnitude and phase equations are

$$|G_{lead}(\omega\, j)| = \sqrt{\frac{T^2\, \omega^2\, a^2 + 1}{1 + T^2\, \omega^2}} \tag{3.1.32}$$

and

$$\phi_G = arctan(a\, T\omega) - arctan(T\, \omega) \tag{3.1.33}$$

The Bode plot of the lead compensator is shown in Figure 3.4.

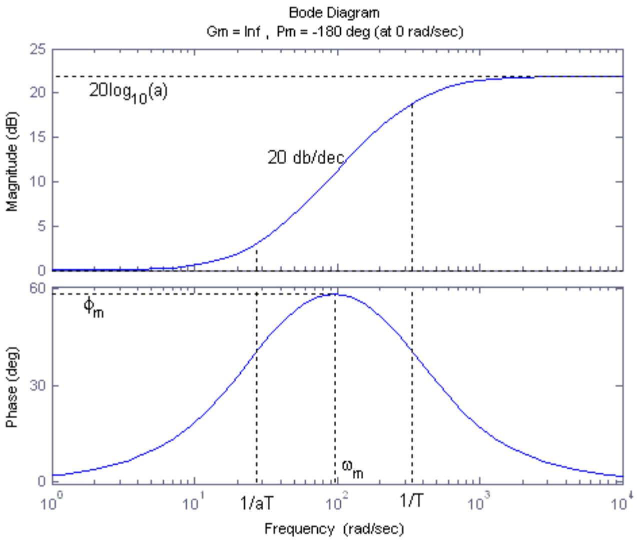


Figure 3.4: Bode of lead compensator.

3.1.3.2 Lead Compensator Design using MATLAB

In this section, we will use Matlab®to design a lead compensator that will satisfy the frequency-based specifications given in Section 3.1.1.1.

1. **Bode plot of the open-loop uncompensated system**, $P_i(s)$, must first be found. To generate the Bode plot of $P_i(s)$, enter the following commands in Matlab®. **NOTE:** If your system has not been set up yet, then you

need to first run the the *setup_srv02_exp03_spd.m* script. This script will store the model parameter *K* and *tau* in the Matlab®workspace. These parameters are used with the commands *tf* and *series* to create the $P_i(s)$ transfer function. The *margin* command generates a Bode plot of the system and it lists the gain and phase stability margins as well as the phase and gain crossover frequencies.

```
% Plant transfer function
P = tf([K],[tau 1]);
% Integrator transfer function
I = tf([1],[1 0]);
% Plant with Integrator transfer function
Pi = series(P,I);
% Bode of Pi(s)
figure(1)
margin(Pi);
set (1,'name','Pi(s)');
```

The entire Lead compensator design is given in the *d_lead.m* script file. Run this script after running the setup_srv02_exp03_spd.m script when CONTROL_TYPE = 'AUTO' to generate a collection of Bode diagrams including the Bode of $P_i(s)$ given in Figure 3.5.

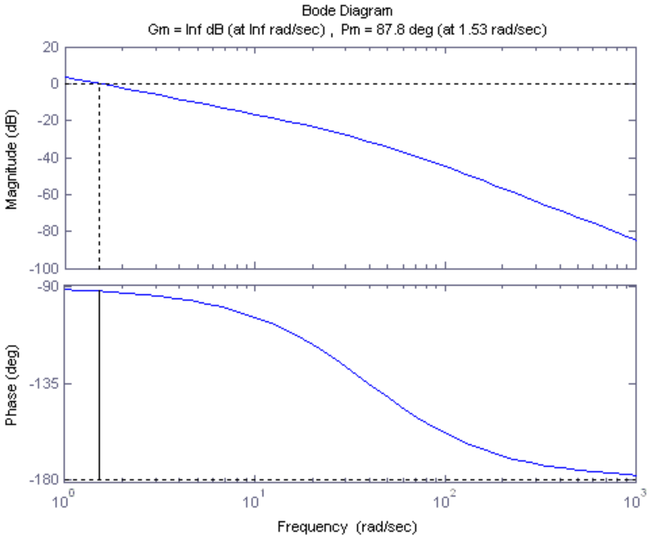


Figure 3.5: Bode of Pi(s) system.

2. **Find how much more gain is required** such that the gain crossover frequency is 50.0 rad/s (use the *ginput* Matlab®command). As mentioned before, the lead compensator adds gain to the system and will increases the phase as well. Therefore, gain K_c is not to be designed to meet the specified 75.0 rad/s fully.

   As given in Figure 3.5, the crossover frequency of the uncompensated system is 1.53 rad/s. To move the crossover frequency to 50.0 rad/s, a gain of

$$K_c = 34.5 \text{ dB} \tag{3.1.34}$$

or

$$K_c = 53.1 \text{ V/rad} \tag{3.1.35}$$

in the linear range is required. The Bode plot of the loop transfer function $L_p(s)$ (from Section 3.1.3) is given in Figure 3.6. This initial estimate of the gain was found using the *ginput* command. The gain was then adjusted according to the crossover frequency calculated in the generated Bode plot of the $L_p(s)$ system. The commands used to generate the Bode plot are given in the d_lead.m script.

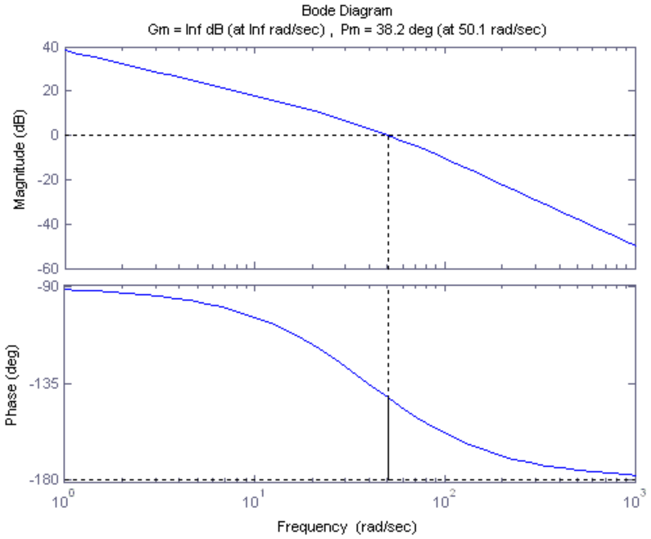


Figure 3.6: Bode of Lp(s) system.

3. **Gain needed for specified phase margin** must be found next so that the lead compensator can achieve the specified phase margin of 75 degrees. Also, to ensure the desired specifications are reached, we'll add another 5 degrees to the maximum phase of the lead.

   To attain the necessary phase margin, the maximum phase of the lead can be calculated using

$$\phi_m = PM_{des} - PM_{meas} + 5 \tag{3.1.36}$$

   Given that the desired phase margin in Equation 3.1.4 and the phase margin of $L_p(s)$ is

$$PM_{meas} = 21.5 \text{ deg} \tag{3.1.37}$$

   the maximum lead phase has to be about

$$\phi_m = 41.8 \text{ deg} \tag{3.1.38}$$

   or

$$\phi_m = 0.728 \text{ rad} \tag{3.1.39}$$

   The lead compensator, as explained in Section 3.1.3.1, has two parameters: a and T. To attain the maximum phase ϕ_m shown in Figure 3.4, the Lead compensator has to add $20 \log_{10}(a)$ of gain. This is determined using the equation

$$a = -\frac{1 + \sin(\phi_m)}{-1 + \sin(\phi_m)} \tag{3.1.40}$$

   The gain needed is found by inserting the max phase into this equation to get

$$a = 4.96 \tag{3.1.41}$$

   which is

$$20 \log 10(a) = 13.9 \text{ dB} \tag{3.1.42}$$

4. **The frequency at which the lead maximum phase occurs** must be placed at the new gain crossover frequency $\omega_{g,new}$. This is the crossover frequency after the lead compensator is applied. As illustrated in Figure 3.4, ω_m occurs halfway between 0 dB and $20 \log_{10}(a)$, i.e. at $10 \log_{10}(a)$. So, the new gain crossover frequency in the $L_p(s)$ system will be the frequency where the gain is $-10 \log_{10}(a)$.

   From Figure 3.6, it is found that the frequency where the $-10 \log_{10}(a)$ gain in the $L_p(s)$ system occurs is at about 80.9 rad/s. Thus, the maximum phase of the lead will be set to

$$\omega_m = 80.9 \text{ rad/s} \tag{3.1.43}$$

As illustrated earlier in Figure 3.4 in Section 3.1.3.1, the maximum phase occurs at the maximum phase frequency ω_m. Parameter T given by:

$$T = \frac{1}{\omega_m \sqrt{a}} \tag{3.1.44}$$

is used to attain a certain maximum phase frequency. This changes where the Lead compensator breakpoint frequencies $1/(a * T)$ and $1/T$ shown in Figure 3.4 occur. The slope of the lead compensator gain changes at these frequencies. We can find the parameter T by substituting $\omega_m = 80.9$ and the lead gain value from Equation 3.1.41 into Equation 3.1.44:

$$T = 0.00556 \text{ s/rad} \tag{3.1.45}$$

Therefore, the lead breakpoint frequencies are:

$$\frac{1}{a\,T} = 36.1 \text{ rad/s} \tag{3.1.46}$$

and

$$\frac{1}{T} = 180.9 \text{ rad/s} \tag{3.1.47}$$

5. **Bode plot of the lead compensator** $C_{lead}(s)$, defined in 3.1.28 can be generated using the *d_lead.m* script.

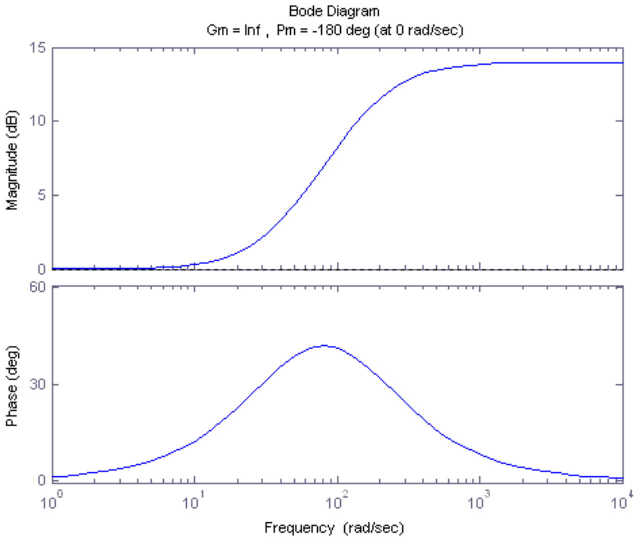


Figure 3.7: Bode of lead compensator $C_{lead}(s)$.

6. **Bode plot of the loop transfer function** $L(s)$, as described in 3.1.30, can be generated using the *d_lead.m* script. The phase margin of $L(s)$ is 68.1 degrees and is below the desired phase margin of 75.0 degrees, as specified in Section 3.1.1.

7. **Check response** by simulating the system to make sure that the time-domain specifications are met. Keep in mind that the goal of the lead design is the same as the PI control, the response should meet the desired steady-state error, peak time, and percentage overshoot specifications given in Section 3.1.1. Thus, if the crossover frequency and/or phase margin specifications are not quite satisfied, the response should be simulated to verify if the time-domain requirements are satisfied. If so, then the design is complete. If not, then the lead design needs to be re-visited.

You will work on this later in the laboratory as described in Section 3.3.2.1.

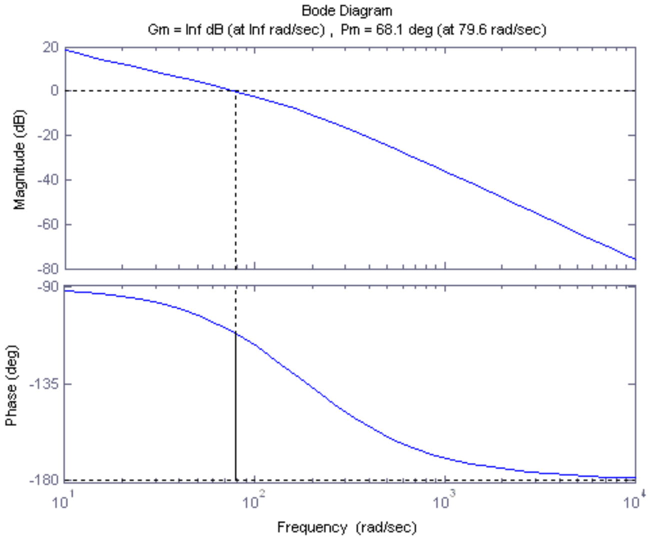


Figure 3.8: Bode of loop transfer function $L(s)$.

3.1.4 Sensor Noise

When using analog sensors, such as a tachometer, there is often some inherent noise in the measured signal.

The peak-to-peak noise of the measured SRV02 load gear signal can be calculated using

$$e_\omega = \frac{1}{100} K_n \, \omega_l \qquad (3.1.48)$$

where K_n is the peak-to-peak ripple rating of the sensor and ω_l is the speed of SRV02 load gear. The rated peak-to-peak noise of the SRV02 tachometer is given in Appendix B of Reference [6] as:

$$K_n = 7 \text{ \%} \qquad (3.1.49)$$

Based on this specification, the peak-to-peak noise, when the load shaft runs at 7.5 rad/s, will be

$$e_\omega = 0.525 \text{ rad/s} \qquad (3.1.50)$$

Thus, the signal will oscillate ± 0.2625 rad/s about the 7.5 rad/s setpoint, or approximately between 7.24 rad/s and 7.76 rad/s. Then, taking the noise into account, what would be the maximum peak in the speed response that is to be expected?

Equation 3.1.7 was used to find the peak value of the load gear response for a given percent overshoot. To take into account the noise in the signal, this formula is modified as follows:

$$\omega(t_p) = \omega_d(t_0) + (\omega_d(t) - \omega_d(t_0)) \left(1 + \frac{PO}{100}\right) + \frac{1}{2} e_\omega \qquad (3.1.51)$$

Given a reference signal that goes between 2.5 rad/s to 7.5 rad/s, as described in Section 3.1.1.1, and the peak-to-peak ripple estimate in Equation 3.1.50, the peak speed of the load gear, including the noise, can be found as:

$$\omega(t_p) = 8.01 \text{ rad/s} \qquad (3.1.52)$$

Using

$$PO = \frac{100\ (\omega(t_p) - \omega_d(t))}{\omega_d(t) - \omega_d(t_0)} \qquad (3.1.53)$$

the new maximum percent overshoot for a 5.0 rad/s step is

$$PO \leq 10.2 \text{ \%} \qquad (3.1.54)$$

3.2 Pre-Lab Questions

1. Based on the steady-state error result of a step response from Equation ,what *type* of system is the SRV02 when performing speed control (Type 0, 1, or 2) and why?

2. The nominal SRV02 model parameters, K and τ, found in SRV02 Modeling Laboratory (Section 1) should be about 1.53 (rad/s-V) and 0.0254 sec, respectively. Calculate the PI control gains needed to satisfy the time-domain response requirements.

3. Find the frequency response magnitude, $|P_i(\omega)|$, of the transfer function $P_i(s)$ given in Equation 3.1.27.

4. Calculate the DC gain of $P_i(s)$ given in Equation 3.1.27. **Hint:** The DC gain is the gain when the frequency is zero, i.e. $\omega = 0\, rad/s$. However, because of its integrator, $P_i(s)$ has a singularity at zero frequency. Therefore, the DC gain is not technically defined for this system. Instead, approximate the DC gain by using $\omega = 1\, rad/s$. Make sure the DC gain estimate is evaluated numerically in dB using the nominal model parameters, $K = 1.53$ and $\tau = 0.0254$, (or use what you found for K and τ in Section 1).

5. The gain crossover frequency, ω_g, is the frequency at which the gain of the system is 1 or 0 dB. Express the crossover frequency symbolically in terms of the SRV02 model parameters K and τ. Then, evaluate the expression using the nominal SRV02 model parameters $K = 1.53$ and $\tau = 0.0254$, (or use what you found for K and τ in Section 1).

3.3 Lab Experiments

The main goal of this laboratory is to explore closed-loop speed control of the SRV02 load shaft.

In this laboratory you will conduct two experiments:

1. Step response with PI control, and
2. Step response with Lead control

In each of the experiments, you will first simulate the closed-loop response of the system. Then, you will implement the controller using the SRV02 hardware and software to compare the real response to the simulated one.

3.3.1 Step Response with PI Control

3.3.1.1 Simulation

First you will simulate the closed-loop speed response of the SRV02 with a PI controller to step input. Our goals are to confirm that the desired reponse specifications in an ideal situation are satisfied and to verify that the motor is not saturated. Then, you will explore the effect of the setpoint weight.

Experimental Setup

The s_srv02_spd Simulink®diagram shown in Figure 3.9 is used to simulate the closed-loop speed response of the SRV02 when using either the PI or Lead controls. The SRV02 Model uses a *Transfer Fcn* block from the Simulink®library to simulate the system. The PI compensator subsystem contains the PI control detailed in Section 3.1.2 and the *Lead Compensator* block has the compensator described in Section 3.1.3.

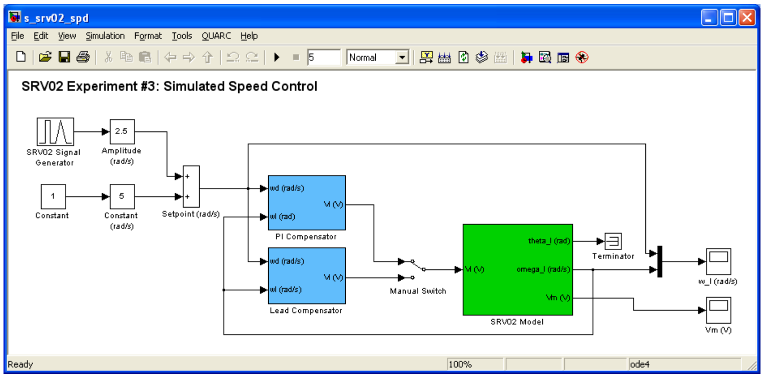


Figure 3.9: Simulink diagram used to simulate the closed-loop SRV02 speed response.

IMPORTANT: Before you can conduct these experiments, you need to make sure that the lab files are configured according to your SRV02 setup. If they have not been configured already, then you need to go to Section 3.4.2 to configure the lab files first.

1. Enter the proportional and integral control gains found in Section 3.1.2.2 as *kp* and *ki* in Matlab®.
2. The speed reference signal is to be a 0.4 Hz square wave that goes between 2.5 rad/s and 7.5 rad/s (i.e. between 23.9 rpm and 71.6 rpm). Set the *SRV02 Signal Generator block* parameters to the following:
   - Signal type = *square*
   - Amplitude = 1

- Frequency = 0.4 Hz

3. In the Speed Control Simulink®model, set the *Amplitude (rad/s)* gain block to 2.5 rad/s and the *Offset (rad/s)* block to 5.0 rad/s.

4. Set the *Manual Switch* to the upward position to activate the PI control.

5. Open the load shaft position scope, *w_l (rad)*, and the motor input voltage scope, *Vm (V)*.

6. Start the simulation. By default, the simulation runs for 5 seconds. The scopes should be displaying responses similar to figures 3.10 and 3.11. Note that in the *w_l (rad)* scope, the yellow trace is the setpoint position while the purple trace is the simulated speed (generated by the *SRV02 Model block*).

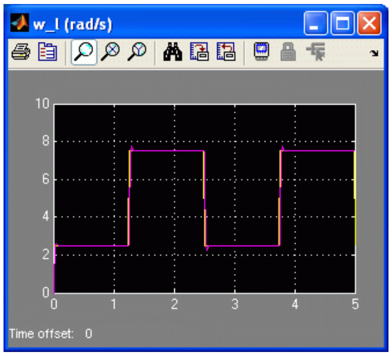


Figure 3.10: Simulated PI speed response.

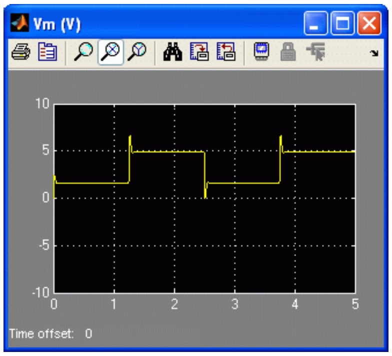


Figure 3.11: Simulated PI motor input voltage.

7. Generate a Matlab®figure showing the simulated PI speed response and its input voltage. After each simulation run, each scope automatically saves their response to a variable in the Matlab workspace. The *w_l (rad)* scope saves its response to the variable called *data_spd* and the *Vm (V)* scope saves its data to the *data_vm* variable. The *data_spd* variable has the following structure: *data_spd(:,1)* is the time vector, *data_spd(:,2)* is the setpoint, and *data_spd(:,3)* is the simulated angular speed. For the *data_vm* variable, *data_vm(:,1)* is the time and *data_vm(:,2)* is the simulated input voltage.

8. Measure the steady-state error, the percent overshoot, and the peak time of the simulated response. Does the response satisfy the specifications given in Section 3.1.1.1?

3.3.1.2 Implementing PI Speed Control

Experimental Setup

The q_srv02_spd Simulink®diagram shown in Figure 3.12 is used to perform the speed control exercises in this laboratory. The SRV02-ET Speed subsystem contains QUARC®blocks that interface with the DC motor and sensors of the SRV02 system, as discussed in Section A. The PI control subsystem implements the PI control detailed in Section 3.1.2 and the Lead Compensator block implements the lead control described in Section 3.1.3.

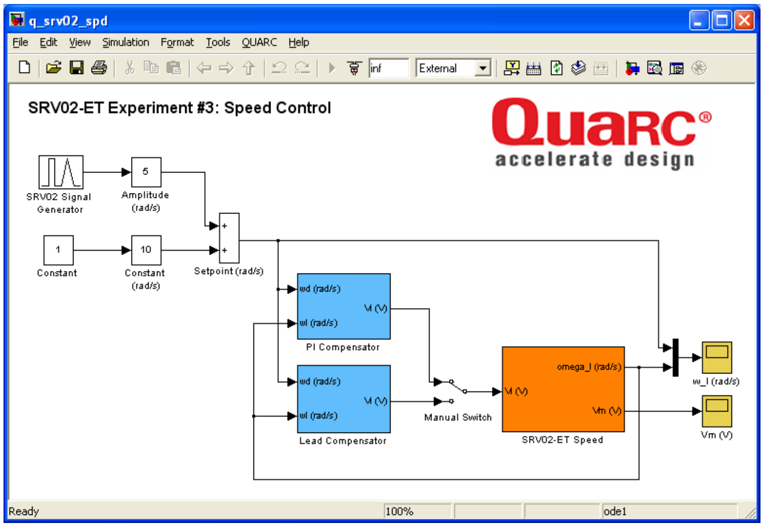


Figure 3.12: Simulink model used with QUARC to run the PI and lead speed controllers on the SRV02.

1. Run the setup_srv02_exp03_spd.m script.
2. Enter the proportional and integral control gains found in Section 3.2 as *kp* and *ki* in Matlab®.
3. To generate a square reference speed signal, set the *SRV02 Signal Generator* block parameters to the following:
    - Signal type = *square*
    - Amplitude = 1
    - Frequency = 0.4 Hz
4. In the Speed Control Simulink®model, set the *Amplitude (rad/s)* gain block to 2.5 rad/s and the *Offset (rad/s)* constant block to 5.0 rad/s.
5. Open the load shaft speed scope, *w_l (rad/s)*, and the motor input voltage scope, *Vm (V)*.
6. Set the *Manual Switch* to the upward position to activate the PI control.
7. Click on QUARC | Build to compile the Simulink®diagram.
8. Select QUARC | Start to run the controller. The scopes should be displaying responses similar to figures Figure 3.13 and Figure 3.14. Note that in the *w_l (rad/s)* scope, the yellow trace is the setpoint position while the purple trace is the measured position.

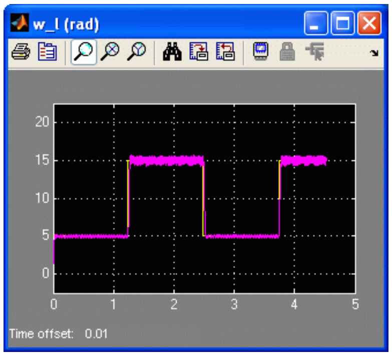


Figure 3.13: Measured PI speed step response.

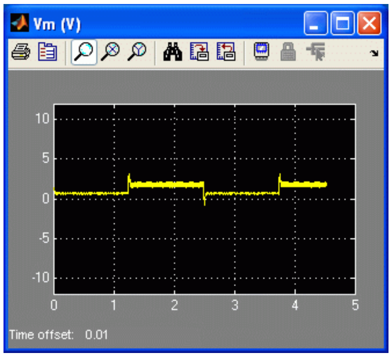


Figure 3.14: PI motor input voltage.

9. When a suitable response is obtained, click on the *Stop* button in the Simulink® diagram toolbar (or select QUARC | Stop from the menu) to stop running the code. Generate a Matlab®figure showing the PI speed response and its input voltage. As in the *s_srv02_spd* Simulink®diagram, when the controller is stopped each scope automatically saves their response to a variable in the Matlab®workspace. Thus, the *theta_l (rad)* scope saves its response to the *data_spd* variable and the *Vm (V)* scope saves its data to the *data_vm* variable.

10. Due to the noise in the measured speed signal, it is difficult to obtain an accurate measurement of the specifications. In the Speed Control Simulink®mode, set the *Amplitude (rad)* block to 0 rad/s and the *Offset (rad)* block to 7.5 rad/s in order to generate a constant speed reference of 7.5 rad/s. Generate a Matlab®figure showing the noise in the signal.

11. Measure the peak-to-peak ripple found in the speed signal, $e_{\omega,meas}$, and compare it with the estimate in Section 3.1.4. Then, find the steady-state error by comparing the average of the measured signal with the desired speed. Is the steady-state error specification satisfied?

12. Measure the percent overshoot and the peak time of the SRV02 load gear step response. Taking into account the noise in the signal, does the response satisfy the specifications given in Section 3.1.1.1?

13. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.

14. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

3.3.2 Step Response with LEAD Control

3.3.2.1 Simulation

You will simulate the closed-loop speed response of the SRV02 with a Lead controller to step input. Our goals are to confirm that the desired reponse specifications in an ideal situation are satisfied and to verify that the motor is not saturated.

As in the step response with PI control experiment in Section 3.3.1.1, in this experiment you need to use the s_srv02_spd Simulink®diagram shown in Figure 3.12 again.

1. Enter the Lead control parameters found in Section 3.1.3.2. These are denoted as *Kc*, *a*, and *T* in Matlab®.

2. Set the *SRV02 Signal Generator* block parameters to the following:
   - Signal type = *square*
   - Amplitude = 1
   - Frequency = 0.4 Hz

3. In the Speed Control Simulink®model, set the *Amplitude (rad/s)* gain block to 2.5 rad/s and the *Offset (rad/s)* constant block to 5.0 rad/s.

4. To engage the lead control, set the *Manual Switch* to the downward position.

5. Open the load shaft position scope, *w_l (rad)*, and the motor input voltage scope, *Vm (V)*.

6. Start the simulation. By default, the simulation runs for 5 seconds. The scopes should be displaying responses similar to Figures 3.10 and 3.11.

7. Verify if the time-domain specifications in Section 3.1.1.1 are satisfied and that the motor is not being saturated. To calculate the steady-state error, peak time, and percent overshoot, use the simulated response data stored in the *data_spd* variable.

8. If the specifications are not satisfied, go back in the lead compensator design. You may have to, for example, add more maximum phase in order to increase the phase margin. If the specifications are met, move on to the next step.

9. Generate a Matlab®figure showing the *Simulated Lead* speed response and its input voltage.

3.3.2.2 Implementing LEAD Speed Control

In this section the speed of the SRV02 is controlled using the lead compensator. Measurements will be taken to see if the specifications are satisfied.

1. Run the setup_srv02_exp03_spd.m script.
2. Enter the *Kc*, *a*, and *T*, lead parameters found in Section 3.1.3.2 in Matlab®.
3. Set the *SRV02 Signal Generator* block parameters to the following:
   - Signal type = *square*
   - Amplitude = 1
   - Frequency = 0.4 Hz
4. In the Speed Control Simulink®model, set the *Amplitude (rad/s)* gain block to 2.5 rad/s and the *Offset (rad/s)* constant block to 5.0 rad/s.
5. To engage the lead compensator, set the *Manual Switch* in the Speed Control Simulink® diagram to the downward position.
6. Open the load shaft speed scope, *w_l (rad/s)*, and the motor input voltage scope, *Vm (V)*.
7. Click on QUARC | Build to compile the Simulink®diagram.
8. Select QUARC | Start to run the controller. The scopes should be displaying responses similar to figures 3.13 and 3.14.
9. When a suitable response is obtained, click on the *Stop* button in the Simulink diagram toolbar (or select QUARC | Stop from the menu) to stop running the code. Generate a Matlab®figure showing the lead speed response and its input voltage.
10. Measure the steady-state error, the percent overshoot, and the peak time of the SRV02 load gear. For the steady-state error, it may be beneficial to give a constant reference and take its average as done in Section 3.3.1.2. Does the response satisfy the specifications given in Section 3.1.1.1?
11. Using both your simulation and implementation results, comment on any differences between the PI and lead controls.
12. Click the Stop button on the Simulink®diagram toolbar (or select QUARC | Stop from the menu) to stop the experiment.
13. Turn off the power to the amplifier if no more experiments will be performed on the SRV02 in this session.

3.3.3 Results

Fill out Table 3.1 below with your answers to the Pre-Lab questions and your results from the lab experiments.

Section / Question	**Description**	**Symbol**	**Value**	**Unit**
Question 2	**Pre-Lab: PI Gains**			
	Proportional Gain	k_p		
	Integral Gain	k_i		
	Open-Loop Time Constant	τ		
	Open-Loop Steady-state Gain	K		
Question 4	**Pre-Lab: DC Gain Estimate**			
	DC Gain Estimate of $P_i(s)$	$\vert P_i(1)\vert$		
Question 5	**Pre-Lab: Gain Crossover Frequency**			
	Gain crossover frequency	ω_g		
Section 3.3.1.1	**In-Lab: PI Step Response Simulation**			
	Peak time	t_p		
	Percent overshoot	PO		
	Steady-state error	e_{ss}		
Section 3.3.1.2	**In-Lab: PI Speed Control Implementation**			
	Measured peak-to-peak ripple	$e_{\omega,meas}$		
	Steady-state error	e_{ss}		
	Peak time	t_p		
	Percent overshoot	PO		
Section 3.3.2.1	**In-Lab: Step Response Simulation with Lead Control**			
	Peak time	t_p		
	Percent overshoot	PO		
	Steady-state error	e_{ss}		
Section 3.3.2.2	**In-Lab: Lead Speed Control Implementation**			
	Peak time	t_p		
	Percentage overshoot	PO		
	Steady-state error	e_{ss}		

Table 3.1: Summary of results for the Speed Control laboratory.

3.4 System Requirements

Before you begin this laboratory make sure:

- QUARC®is installed on your PC, as described in Reference [1].
- You have a QUARC®compatible data-aquisition (DAQ) card installed in your PC. For a listing of compliant DAQ cards, see Reference [5].
- SRV02 and amplifier are connected to your DAQ board as described Reference [6].

3.4.1 Overview of Files

Table 3.2: Files supplied with the SRV02 Speed Control laboratory.

File Name	**Description**
03 - SRV02 Speed Control - Student Manual.pdf	This laboratory guide contains pre-lab questions and lab experiments demonstrating how to design and implement a speed controller on the Quanser SRV02 rotary plant using QUARC®.
setup_srv02_exp03_spd.m	The main Matlab script that sets the SRV02 motor and sensor parameters as well as its configuration-dependent model parameters. *Run this file only to setup the laboratory.*
config_srv02.m	Returns the configuration-based SRV02 model specifications *Rm*, *kt*, *km*, *Kg*, *eta*_g, *Beq*, *Jeq*, and *eta*_m, the sensor calibration constants K_POT, K_ENC, and K_TACH, and the amplifier limits VMAX_AMP and IMAX_AMP.
d_model_param.m	Calculates the SRV02 model parameters *K* and *tau* based on the device specifications *Rm*, *kt*, *km*, *Kg*, *eta_g*, *Beq*, *Jeq*, and *eta_m*.
calc_conversion_constants.m	Returns various conversions factors.
s_srv02_spd.mdl	Simulink file that simulates the closed-loop SRV02 speed control using either the PI control or the lead compensator.
q_srv02_spd.mdl	Simulink file that runs the PI or Lead speed control on the actual SRV02 system using QUARC®.

3.4.2 Setup for Speed Control Simulation

Follow these steps to configure the Matlab®setup script and the Simulink®diagram for the Speed Control simulation laboratory:

1. Load the Matlab®software.
2. Browse through the Current Directory window in Matlab®and find the folder that contains the SRV02 speed controller files, e.g. s_srv02_spd.mdl.
3. Double-click on the s_srv02_spd.mdl file to open the SRV02 Speed Control Simulation Simulink diagram shown in Figure 3.9.

4. Double-click on the setup_srv02_exp03_spd.m file to open the setup script for the position control Simulink models.

5. **Configure setup script:** The controllers will be run on an SRV02 in the high-gear configuration with the disc load, as in Section 1. In order to simulate the SRV02 properly, make sure the script is setup to match this configuration, e.g. the EXT_GEAR_CONFIG should be set to 'HIGH' and the LOAD_TYPE should be set to 'DISC'. Also, ensure the ENCODER_TYPE, TACH_OPTION, K_AMP, AMP_TYPE, and VMAX_DAC parameters are set according to the SRV02 system that is to be used in the laboratory.
   Finally, make sure CONTROL_TYPE is set to 'MANUAL'.

6. Run the script by selecting the Debug | Run item from the menu bar or clicking on the Run button in the tool bar. The messages shown below, should be generated in the Matlab® Command Window. The correct model parameters are loaded but the control gains and related parameters loaded are default values that need to be changed. That is, the PI control gains are all set to zero, the lead compensator parameters *a* and *T* are both set to 1, and the compensator proportional gain *Kc* is set to zero.

   ```
   SRV02 model parameters:
      K = 1.53 rad/s/V
      tau = 0.0254 s
   PI control gains:
      kp = 0 V/rad
      ki = 0 V/rad/s
   Lead compensator parameters:
      Kc = 0 V/rad/s
      1/(a*T) = 1 rad/s
      1/T = 1 rad/s
   ```

3.4.3 Setup for Speed Control Implementation

Before beginning the in-lab exercises on the SRV02 device, the q_srv02_spd Simulink® diagram and the setup_srv02_exp03_spd script must be configured.

Follow these steps to get the system ready for this lab:

1. Setup the SRV02 in the high-gear configuration and with the disc load as described in Reference [6].

2. Load the Matlab software.

3. Browse through the *Current Directory* window in Matlab® and find the folder that contains the SRV02 speed control files, e.g. q_srv02_spd.mdl.

4. Double-click on the q_srv02_spd.mdl file to open the Speed Control Simulink diagram shown in Figure 3.12.

5. **Configure DAQ:** Double-click on the HIL Initialize block in the *SRV02-ET* subsystem (which is located inside the *SRV02-ET* Speed subsystem) and ensure it is configured for the DAQ device that is installed in your system. See Section A for more information on configuring the HIL Initialize block.

6. **Configure Sensor:** To perform the speed control experiment, the angular rate of the load shaft should be measured using the tachometer. This has already been set in the *Spd Src* Source block inside the *SRV02-ET Speed* subsystem.

7. **Configure setup script:** Set the parameters in the setup_srv02_exp03_spd.m script according to your system setup. See Section 3.4.2 for more details.

3.5 Lab Report

This laboratory contains two experiments, namely,

1. step response with PI control, and
2. step response with lead control.

When you are writing your report, follow the outline corresponding to the experiment you conducted to build the *content* of your report. Also, in Section 3.5.3 you can find some basic tips for the *format* of your report.

3.5.1 Template for Content (PI Control Experiments)

I. PROCEDURE

I.1. Step Response with PI Control

1. *Simulation*
    - Briefly describe the main goal of this simulation.
    - Briefly describe the procedure (Section 3.3.1.1)
2. *Implementation*
    - Briefly describe the main goal of this experiment.
    - Briefly describe the experimental procedure (Section 3.3.1.2)

II. RESULTS
Do not interpret or analyze the data in this section. Just provide the results.

1. Response plot from step 7 in Section 3.3.1.1, *Step response simulation with PI Control*
2. Response plot from step 9 in Section 3.3.1.2, *Step response implementation with PI Control*
3. Signal noise plot from step 10 in Section 3.3.1.2, *Step response implementation with PI Control*
4. Provide data collected in this laboratory (from Table 3.1).

III. ANALYSIS
Provide details of your calculations (methods used) for analysis for each of the following:

III.1. Step Response with PI Control

1. Step 8 in Section 3.3.1.1, *Step response simulation with PI Control*
2. Step 11 in Section 3.3.1.2, *Step response implementation with PI Control*
3. Step 12 in Section 3.3.1.2, *Step response implementation with PI Control*

IV. CONCLUSIONS
Interpret your results to arrive at logical conclusions for the following:

1. Step 8 in Section 3.3.1.1, *Step response simulation with PI Control*
2. Step 11 in Section 3.3.1.2, *Step response implementation with PI Control*
3. Step 12 in Section 3.3.1.2, *Step response implementation with PI Control*

3.5.2 Template for Content (Lead Control Experiments)

I. PROCEDURE

I.1. Step Reponse with Lead Control

1. *Simulation*
    - Briefly describe the main goal of this simulation.
    - Briefly describe the procedure (Section 3.3.2.1)
2. *Implementation*
    - Briefly describe the main goal of this experiment.
    - Briefly describe the experimental procedure (Section 3.3.2.2)

II. RESULTS
Do not interpret or analyze the data in this section. Just provide the results.

1. Response plot from step 9 in Section 3.3.2.1, *Step response simulation with Lead Control*
2. Response plot from step 9 in Section 3.3.2.2, *Step response implementation with Lead Control*
3. Provide data collected in this laboratory (from Table 3.1).

III. ANALYSIS
Provide details of your calculations (methods used) for analysis for each of the following:

III.1. Step Response with Lead Control

1. Step 7 in Section 3.3.2.1, *Step response simulation with Lead Control*
2. Step 10 in Section 3.3.2.2, *Step response implementation with Lead Control*

IV. CONCLUSIONS
Interpret your results to arrive at logical conclusions for the following:

1. Step 7 in Section 3.3.2.1, *Step response simulation with Lead Control*
2. Step 10 in Section 3.3.2.2, *Step response implementation with Lead Control*
3. Step 11 in Section 3.3.2.2, *Step response implementation with Lead Control*

3.5.3 Tips for Report Format

PROFESSIONAL APPEARANCE

- Has cover page with all necessary details (title, course, student name(s), etc.)
- Each of the required sections is completed (Procedure, Results, Analysis and Conclusions).
- Typed.
- All grammar/spelling correct.
- Report layout is neat.
- Does not exceed specified maximum page limit, if any.
- Pages are numbered.
- Equations are consecutively numbered.
- Figures are numbered, axes have labels, each figure has a descriptive caption.
- Tables are numbered, they include labels, each table has a descriptive caption.
- Data are presented in a useful format (graphs, numerical, table, charts, diagrams).
- No hand drawn sketches/diagrams.
- References are cited using correct format.

Appendix A

SRV02 QUARC INTEGRATION

In this section, we explain how to send command voltages to the Quanser®SRV02 and measure the position and speed of its load shaft in real-time using your computer.

System Requirements

Before you begin this laboratory make sure:

- QUARC®is installed on your PC, as described in [1].
- You have a QUARC®compatible data-acquisition (DAQ) card installed in your PC. For a listing of compliant DAQ cards, see [5].
- SRV02 and amplifier are connected to your DAQ board as described in [6].

Prerequisites

The user should be familiar with the following:

- System hardware:
  - Data acquisition card (e.g. Q2-USB in [2])
  - Amplifier (e.g. VoltPAQ in [4]).
  - Main components of the SRV02, i.e. DC motor and sensors such as the potentiometer [6].
- Basics of Simulink®.

A.1 Applying Voltage to SRV02 Motor

Here are the basic steps to apply a voltage to the SRV02 motor using QUARC®:

1. Make a Simulink®model that interacts with your installed data-acquisition device using blocks from the *QUARC Targets* library. This is explained in Section A.1.1,
2. From the Simulink®model, build real-time code as shown in Section A.1.2, and
3. Execute the code as explained in Section A.1.3.

A.1.1 Making the Simulink Model

In this section, we will make a Simulink®model as shown in Figure A.1 using QUARC®blocks to feed a sinusoidal voltage to the SRV02 dc motor. The blocks from the *QUARC Targets* library are used to interact with a data-acquisition board, e.g. Quanser Q2-USB or Q8-USB device.

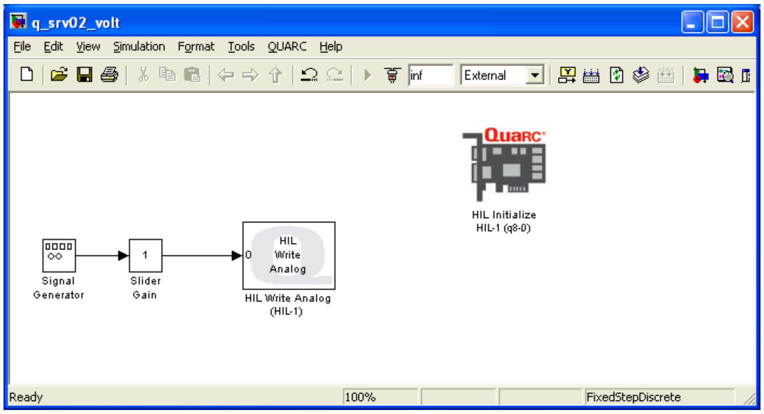
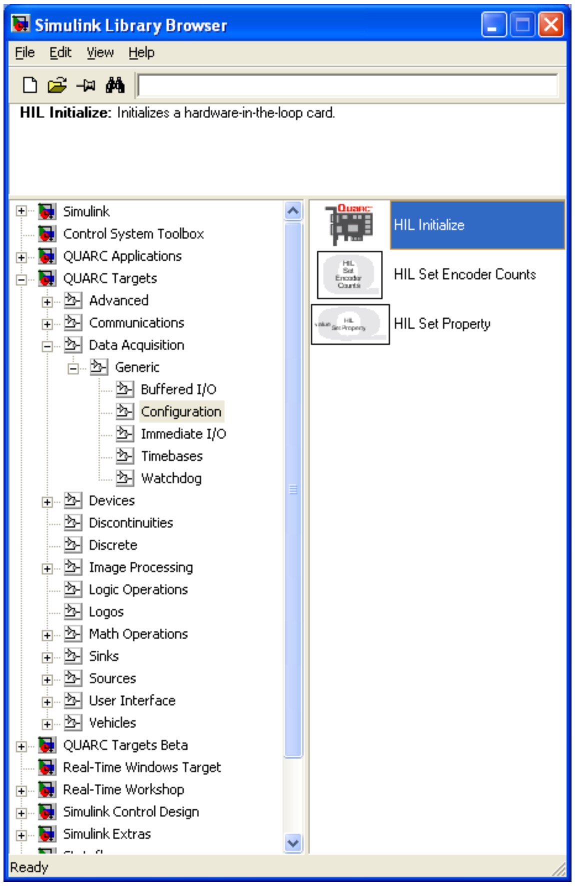


Figure A.1: Simulink model used with QUARC to apply voltage to SRV02

Follow these steps to make the Simulink®diagram:

1. Load the Matlab®software.
2. Create a new Simulink®diagram. To do this, go to File | New | Model item in the menu bar.
3. Open the Simulink Library Browser window by clicking on the View | Library Browser item in the Simulink menu bar or clicking on the Simulink icon.
4. Expand the *QUARC Targets* item and go to the Data Acquisition | Generic | Configuration folder, as shown in Figure A.2.
5. Click-and-drag the *HIL Initialize block* from the library window into the blank Simulink®model. This block is used to configure your data-acquisition device, e.g. the Quanser Q2-USB or Q8-USB hardware-in-the-loop (HIL) boards.
6. In the Library Browser, go to the Data Acquisition | Generic | Immediate I/O category. This contains various blocks used to interact with actuators and sensors.
7. Click-and-drag the *HIL Write Analog* block from the library into the Simulink®diagram. This block is used to output a voltage from an Analog Output channel, i.e. digital-to-analog (D/A) channel, on the data-acquisition device.



HIL Initialize: Initializes a hardware-in-the-loop card.

Figure A.2: *QUARC Targets* in Simulink®Library Browser

8. Add the *Signal Generator* block, found in the Simulink®| Source folder, and the *Slider Gain* block, from the Simulink®| Math Operations category, into the Simulink®model. Connect the blocks as shown in A.1.

9. Double-click on the *HIL Initialize* block:

   - In the *Board type* field, select the board that is installed in your PC. For example, if you have a Quanser Q2-USB board then select *q2_usb*.
   - If more than one of the same board is installed (e.g. two Q2-USB devices), ensure the *Board number* field is set correctly, e.g. if two boards are used, then choose either 0 or 1.

10. Go to the *Analog Output* pane shown in Figure A.3. Make sure the *Analog output channels* field includes 0. This will ensure 0 V is output from Analog Output channel #0 when the QUARC controller is stopped. For more information, click on the *Help* button. Otherwise, click on the OK button and proceed.

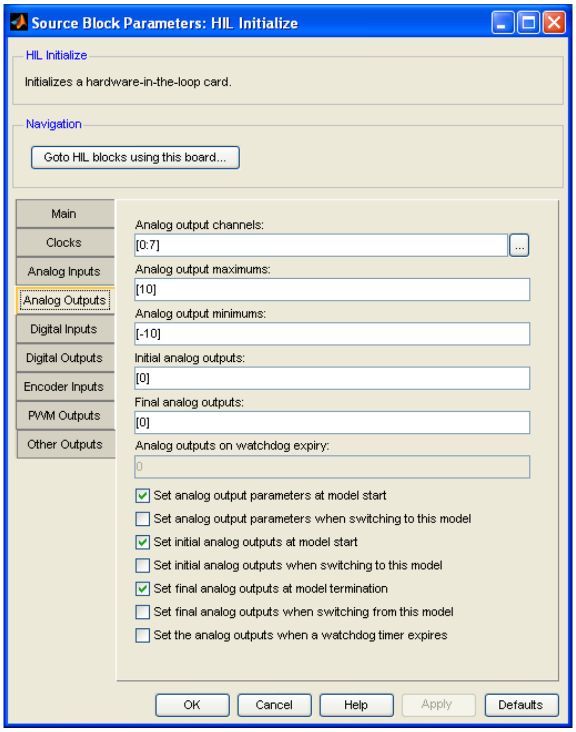
    Note: If you are using a NI DAQ device, make sure you enter [0] in the Analog Output channel box. The analog output channels for the NI boards are not selected by default.

11. Double-click on the *HIL Write Analog* block.

    - Make sure *Board name* is set to HIL-1 (i.e. points to the HIL Initialize block).
    - Set *Channels* to 0 (default setting). Recall that, as instructed in Reference [6], the dc motor is connected to Analog Output Channel #0 on the hardware-in-the-loop board. Therefore, *Channels* should be set to 0.
    - Set *Sample time* to -1 (default setting). This implies that the sampling interval is inherited from the previous block.

12. Click on the OK button to save and close the *HIL Write Analog* block properties.

13. Save the Simulink®mode by selecting the File | Save item in the menu bar or clicking on the Save icon.



HIL Initialize

Initializes a hardware-in-the-loop card.

Navigation

Goto HIL blocks using this board...

Analog output channels: [0:7]

Analog output maximums: [10]

Analog output minimums: [-10]

Initial analog outputs: [0]

Final analog outputs: [0]

Analog outputs on watchdog expiry: 0

☑ Set analog output parameters at model start

□ Set analog output parameters when switching to this model

☑ Set initial analog outputs at model start

□ Set initial analog outputs when switching to this model

☑ Set final analog outputs at model termination

□ Set final analog outputs when switching from this model

□ Set the analog outputs when a watchdog timer expires

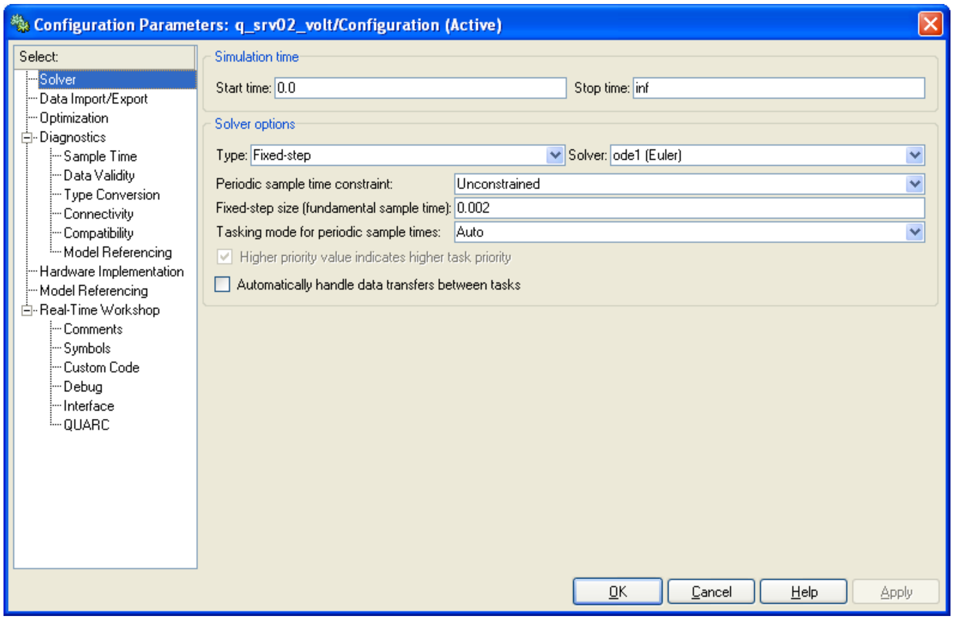
Figure A.3: Configuring Analog Output channel on DAQ device

A.1.2 Compiling the Model

The Simulink®model we made in Section A.1 can now be used by QUARC to generate code. When you execute this code, a voltage is sent to the SRV02.

Follow these steps to generate code from a Simulink®diagram:

1. In the Simulink®diagram made in Section A.1, go to the QUARC | Set default options item to set the correct Real-Time Workshop parameters and setup the Simulink®model for external use (as opposed to the simulation mode).
2. To view the compiler options shown in Figure A.4, go to QUARC | Options in the Simulink®model tool bar.
3. Real-Time Workshop pane:
   - *System target file* is set to Target Language Compiler file quarc_windows.tlc.
   - *Make command* is set to make_rtw and the *Template makefile* is set to quarc_default_tmf file.
4. Solver pane:
   - *Stop time* is set to *inf* in order for the code to be executed continuously until it is stopped manually by the user. Alternatively, the *Stop time* parameter can be set to the desired duration (code will cease executing when the stop time value is reached).
   - *Type* parameter is set to Fixed-step. When compiling real-time code, the solver must be fixed-step as opposed to variable step which can be used in simulations.
   - *Solver* is set to discrete. There are no continuous blocks inside the designed Simulink® model, therefore having a discrete solver is fine. However, if an Integrator block or another continuous system were be added, then the Solver field would have to be changed to an integration method such as *ode1 (Euler)*.
   - *Fixed-step size* field sets the sampling interval, or sampling time, of the controller. By default this is set to 0.002 seconds, which is a sampling rate of 500 Hz.



Configuration Parameters: q_srv02_volt/Configuration (Active)

Simulation time

Start time: 0.0 Stop time: inf

Solver options

Type: Fixed-step Solver: ode1 (Euler)

Periodic sample time constraint: Unconstrained

Fixed-step size (fundamental sample time): 0.002

Tasking mode for periodic sample times: Auto

☑ Higher priority value indicates higher task priority

☐ Automatically handle data transfers between tasks

Figure A.4: Default QUARC solver settings

5. Click on the OK button to close the Configuration Parameters window.
6. Select the QUARC | Build item. Various lines in the Matlab®Command Window should be displayed as the model is being compiled.
7. Once done compiling, a QUARC Windows executable file along with a folder containing various ``C" and Matlab®files are generated. Note that once the executable is created, the folder is no longer needed. If you like, you can remove the executable and associated code folder may be removed from the current directory by clicking on QUARC | Clean item.

See [5] for more information about configuring QUARC®.

A.1.3 Running QUARC Code

Once the Simulink®model has been compiled, the code can be executed and the voltage set in the Simulink®model can be sent to the SRV02 motor. Here are the steps to follow:

1. Power ON your power amplifier (e.g. Quanser VoltPAQ).
2. To begin executing the code, click on the QUARC | Start item in the Simulink®model. The SRV02 external gears on the SRV02 should begin rotating back-and-forth. This command actually does two things: it connects to the target and then executes the code. Alternatively, users may decide to go through these steps individually:

   Option 1: In the Simulink®model tool bar, click on the *Connect to target* icon and then click on the *Run* icon. These buttons are shown in Figure A.5.

   Option 2: Select the Simulation | Connect to target item from the menu bar and the select Simulation | Start Real-Time Code item.

3. Double-click on the Signal Generator block to open its parameter window.
4. Set the *Frequency* field to 0.5 Hz and click on the OK button. Notice how the parameter change effects the SRV02 immediately: the gears on the SRV02 begin to switch back-and-forth slowly.

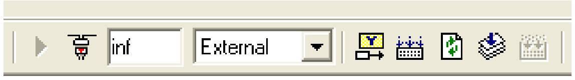


Figure A.5: Simulink®model toolbar: connect to target and compilation

5. Vary the value of the Slider Gain block between 0 and 2. Examine how the angular rate of the SRV02 gears change proportionally with the amplitude of the sine wave.
6. Select the QUARC | Stop item to stop the code execution (or click on the Stop button in the Simulink®model tool bar).
7. Power OFF the amplifier if no more experiments will be run in this session.

A.2 Reading Position using Potentiometer

In this section, the Simulink®model previously designed in Section A.1 is modified to obtain readings from the potentiometer sensor.

A.2.1 Reading the Potentiometer Voltage

Follow the procedure below to design the Simulink®diagram as shown in Figure A.6 to read the potentiometer voltage.

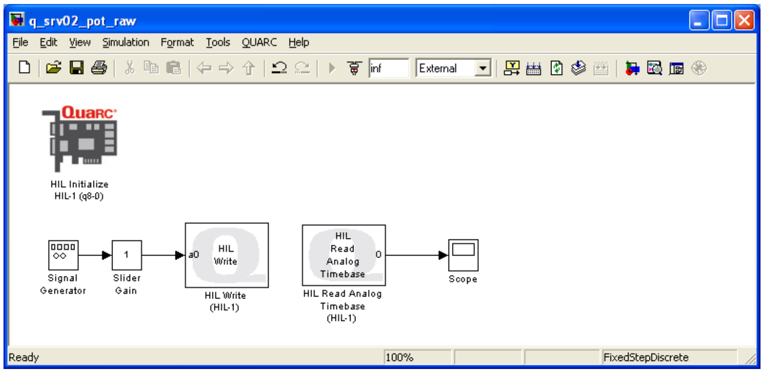


Figure A.6: Simulink®model used with QUARC to read potentiometer

1. From the Generic | Timebases category in the Simulink®Library Browser, click-and-drag the *HIL Read Analog Timebase* block into the Simulink®diagram created previously in Section A.1. This block will be used to read a voltage from an Analog Input channel, i.e. analog-to-digital (A/D), on the data-acquisition device.

   Note: Using a *Timebase* type block causes the running model to be triggered by the hardware timer on the data-acquisition board as opposed to the system clock. This increases performance of the controller by reducing jitter and allowing for greater sampling rates. If you want to use system clock instead (some DAQ cards do not support hardware-based timing, see [5] for details), then use blocks from the *Immediate I/O* category instead.

2. Recall that, as instructed in [6], the potentiometer is connected to Analog Input #0 on the DAQ board. Therefore, the default setting of 0 in the block is fine.

3. Add a Scope block from the Simulink®| Sinks folder in the Library Browser and connect it to the output of the *HIL Read Analog Timebase* block, as shown in Figure A.6.

4. Set the *Frequency* parameter in the Signal Generator block to 1.0 Hz and the Slider Gain block to 1.

5. Double-click on the *Scope* block to open the scope.

6. Save the Simulink®model (if you want to save the model as a different file, go to *Save As*).

7. Power ON the power amplifier.

8. Go to QUARC | Build to compile the code.

9. Click on QUARC | Start to execute the code. As the SRV02 rotates back-and-forth, the Scope should display the potentiometer readings similar to Figure A.7. Notice that the readings shown are the voltage output of the potentiometer and not an angular measurement in degrees or radians.

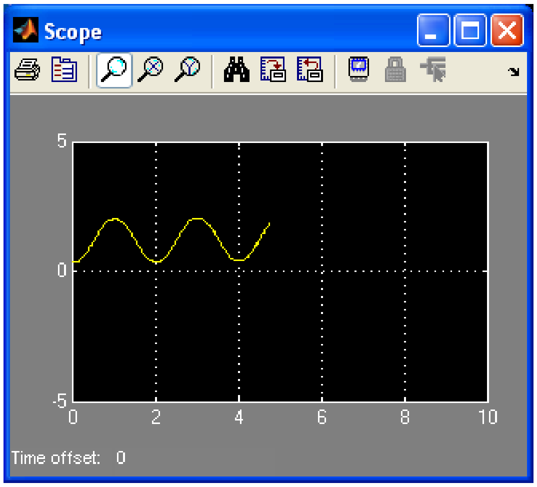


Figure A.7: Reading raw voltage from the SRV02 potentiometer

Note: If the SRV02 gears are rotating but the scope is not plotting any data, then you must re-configure the RTW Signal & Triggering properties inside the Simulink®diagram. See the *Signal Triggering* section under *QUARC Basics* in [5] for more information.

10. Vary the sine wave frequency between 0.1 and 2 Hz and the Slider Gain value between 0 and 2. Notice how the controller is updated in real-time and the measurement in the scope changes with the new parameters.

    Note: The potentiometer has an electrical range of 352 degrees across ± 5 V [6]. As a result, the potentiometer outputs a discontinous voltage signal as seen in Figure A.8. Then, a disadvantage is that you cannot surpass the ± 180 degree range when controlling the position or take the derivative of this sensor to control velocity.

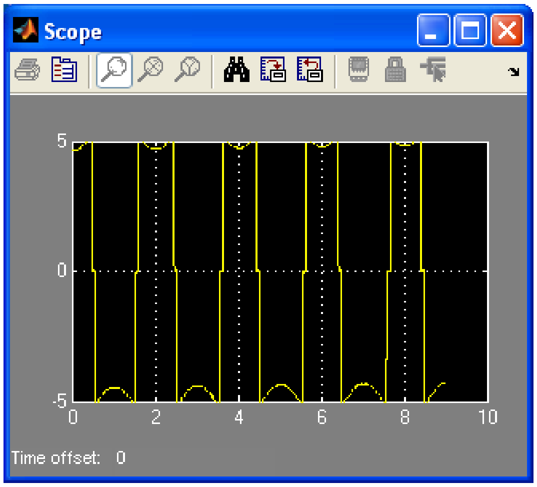


Figure A.8: Potentiometer reading when encountering its discontinuity

11. Stop the code by clicking on QUARC | Stop.

A.2.2 Measuring Position

The Simulink®model will now be modified to read the *load angle* from the potentiometer and the voltage commanded to the motor as shown in Figure A.9.

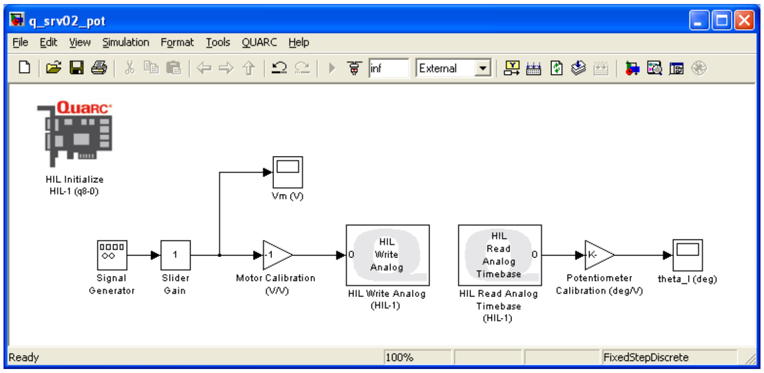


Figure A.9: Simulink®model used with QUARC to send voltage to the SRV02 and read its load shaft angle using potentiometer.

Modify the Simulink®diagram built in the previous section following these steps:

1. Label the Scope block that is presently connected to the *HIL Read Analog Timebase* block to *theta_l (deg)*. This will display the angular measurement of the load gear in degrees.
2. Add a Scope block from the Simulink®| Sinks folder in the Library Browser and connect it to the output of the Slider Gain block, as shown in Figure A.9, above. Label the Scope block *Vm (V)* which stands for the input motor voltage.
3. From the Simulink®| Math Operations category in the Library Browser, add two Gain blocks into the Simulink®model.
4. Connect the gain blocks as illustrated in Figure A.9.
   - One gain block between the *Amplifier Pre-Compensation* and *HIL Write Analog* blocks and label it *Motor Calibration (V/V)*.
   - Another gain block between the *HIL Read Analog Timebase* and *theta_l* Scope blocks and denote it *Potentiometer Calibration (deg/V)*.
5. Re-compile the Simulink®model, i.e. click on QUARC | Build.
6. Click on QUARC | Start to run the code.
7. Open the *Vm (V)* and *theta_l (deg)* scopes.
8. In the Signal Generator, set the *Frequency* parameter to 0.25 Hz.
9. Set the Slider Gain block to 1.
10. Currently when the voltage goes positive the load gear rotates in the clockwise direction. However, the desired convention is for the load gear to rotate in the counter-clockwise direction when the voltage goes positive. Thus, set the *Motor Calibration (V/V)* block to -1.
11. As described in [6], the potentiometer outputs between ± 5 V when it rotated 352 degrees. Enter the value 352/10 in the *Potentiometer Calibration (deg/V)*block.
12. In the *theta_l (deg)* scope, click on the *Autoscale* icon so the y-axis range is increased and the full signal can be viewed.
13. Set the Slider Gain block to 0.
14. Rotate the load gear manually and examine the corresponding response in the *theta_l (deg)* scope. Confirm that, indeed, the correct measurement is being taken.
15. Position the load gear such that 0 is read in the *theta_l (deg)* scope .

16. Set the Slider Gain block to 1.

17. Examine the relationship between the input voltage and load position. When the input voltage increases in the positive direction, the potentiometer angle decreases. Add a negative sign to the *Potentiometer Calibration (deg/V)* block so the entered value becomes -352/10. Now, the angular position of the load gear, displayed in the *theta_l (deg)* scope, reads increasingly positive when the commanded motor input voltage, i.e. signal in the *Vm (V)* scope, is positive.

    Note: The theoretical model assumes that the velocity increases when the voltage is positive. The feedback control design is based on the model. Therefore, it is imperative that the actual system follows the same conventions as the model that will be developed.

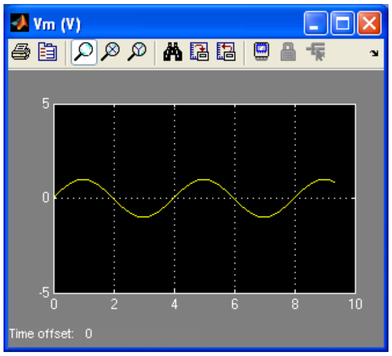


Figure A.10: Sinusoidal input voltage.

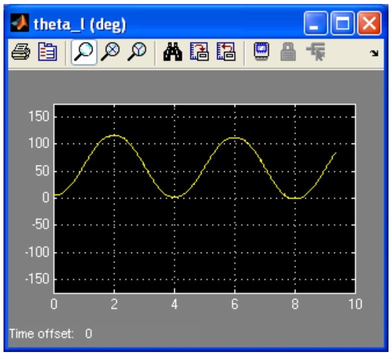


Figure A.11: Load gear position reading using potentiometer.

18. Select the QUARC | Stop item to stop the code from running.

19. Power OFF the amplifier if no more experiments will be run in this session.

A.3 Measuring Speed using Tachometer

In this section, we will modify the Simulink®diagram designed in Section A.2 to include the readings from the tachometer.

Before continuing, please ensure that the SRV02 unit being used has a tachometer, i.e. SRV02-T option.

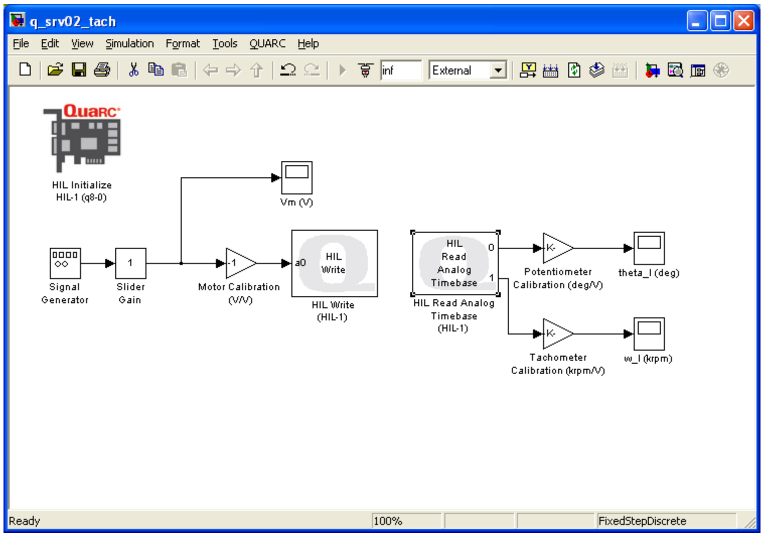


Figure A.12: Simulink®model used with QUARC to send voltage to SRV02 and read the potentiometer and tachometer sensors.

Using the Simulink®diagram built in Section A.2, go through this procedure to add the tachometer functionality:

1. Double-click on the *HIL Read Analog Timebase* block to open its properties.
2. As detailed in [6], the tachometer is connected to Analog Input #1 on the hardware-in-the-loop board. To add a channel, set the *Analog channels* field to [0,1] and click on the OK button.
3. Add a Scope block and a Gain block from Library Browser into the Simulink®model.
4. Connect the Scope and Gain blocks as depicted in Figure A.12, above.
   - Connect the Gain block to the output of Channel #1 from the *HIL Read Analog Timebase* block and label it *Tachometer Calibration (krpm/V)*.
   - Connect the output of this gain block to the input of the Scope block and label the scope *w_l (rpm)*. The speed of the load gear is denoted by the ω_l. This scope will display the measured speed of the load gear in thousand revolutions per minute.
5. Set the Signal Generator *Frequency* parameter to 1.0 Hz and the Slider Gain block to 1.
6. Open the *Vm (V)* and the *w_l (krpm)* scopes.
7. Power ON the amplifier.
8. Go to QUARC | Build to compile the code.
9. Click on QUARC | Start to execute the code. As the SRV02 rotates back-and-forth, the *w_l (rpm/V)* scope should display the tachometer readings. Since the *Tachometer Calibration (krpm/V)* gain has not been configured yet, the scope is displaying the tachometer output voltage, which is proportional to the speed of the load shaft.

10. The back-emf constant of the tachometer sensor is 1.5 mV/rpm. However, the measurement is taken directly from the motor itself (see [6]). Thus, to read the velocity of the gear the tachometer calibration gain must be divided by the gear ratio. Enter 1 / 1.5 / 70 in the *Tachometer Calibration (krpm/V)* gain block when using the SRV02 in the high-gear configuration (or 1 / 1.5 / 14 if using the low-gear configuration).

    Note: The measurement will be very small. Click on the *Autoscale* icon in the scope to zoom up on the signal. Alternatively, the y-range of the scope can be set manually. To do this, right-click on the y-axis, select Axes Properties from the drop-down menu, and set the desired y-range values.

11. Examine the relationship between the input voltage and load speed. When the input voltage increases in the positive direction, the tachometer velocity decreases. Similar to the potentiometer, the speed of the load shaft should go positive when the input voltage is positive. To fix this, add a negative sign to the *Tachometer Calibration (krpm/V)* block so the entered value becomes -1 / 1.5 / 70 (or -1 / 1.5 / 14 for low-gear). The input voltage and load velocity scopes should read as shown in Figure A.13 and Figure A.14.

    Note: If you want the measurement to be in RPM instead of kRPM, enter -1000 / 1.5 / 70 gear (or -1000 / 1.5 / 14 for low-gear).

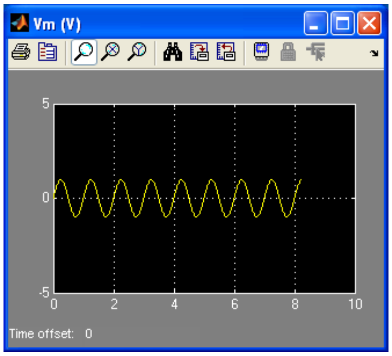


Figure A.13: Sinusoidal input voltage.

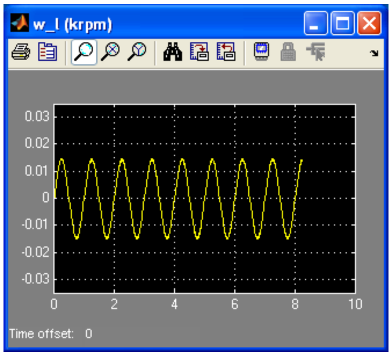


Figure A.14: High gear velocity reading using tachometer.

12. Select the QUARC | Stop item to stop the code from running.

13. Power OFF the amplifier if no more experiments will be run in this session.

A.4 Measuring Position using Encoder

The Simulink®diagram designed previously is modified to include an encoder measurement, as illustrated in Figure A.15 below.

Before continuing, please ensure that the SRV02 unit being used has an encoder, i.e. SRV02-E option.

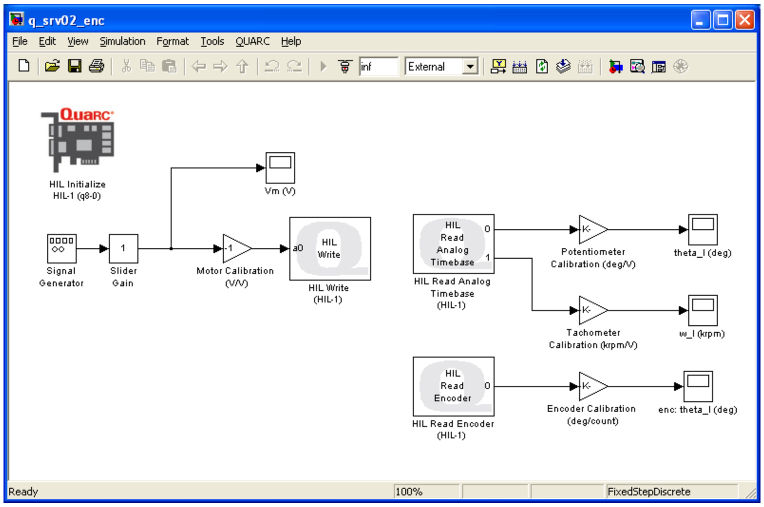


Figure A.15: Simulink®model used with QUARC to send voltage to SRV02 and reads the potentiometer, tachometer, and encoder sensors.

Using the Simulink®model designed in either Section A.2 or A.3, follow this procedure to add encoder functionality:

1. From the QUARC Targets | Data Acquisition | Generic | Immediate I/O category in the Library Browser, add a HIL Read Encoder block.
2. Recall that, as instructed in [6], the encoder is connected to Encoder Input #0 on the data acquisition board. The HIL Read Encoder block is already configured for to read channel 0 and the default encoder configurations in the HIL Initialize block are fine (but keep in mind that these can be changed).
3. Add a Gain block and a Scope block from the *Math Operations* and *Sinks* folders in the Library Browser, respectively, into the Simulink®model.
4. Connect the scope and gain blocks as depicted in Figure A.15.
    - Connect the HIL Read Timebase block output *e0* to a Gain block and label it *Encoder Calibration (deg/count)*.
    - Connect the output of this gain block to the input of the Scope block and label the scope *enc: theta_l (deg)*. This scope will display the measured angular position of the load gear in degrees.
5. Set the *Frequency* parameter in the Signal Generator block to 1.0 Hz and the Slider Gain block to 1.
6. Open the *Vm (V)* and *enc: theta_l (deg)* scopes.
7. Save the Simulink®model (you may want to save the model as a different file).
8. Power ON the power amplifier.
9. Go to QUARC | Build to compile the code.

10. Click on QUARC | Start to execute the code. As the SRV02 rotates back-and-forth, the *enc: theta_l (deg)* should display the encoder readings. Since the *Encoder Calibration (deg/count)* gain has not been configured yet, the scope is displaying the number of counts from the encoder output, which is proportional to the position of the load shaft.

    Note: The measurement will be very large. Click on the Autoscale icon in the scope to zoom out and view the entire signal. Alternatively, the y-range of the scope can be set manually. To do this, right-click on the y-axis, select Axes Properties from the drop-down menu, and set the desired y-range values.

11. As discussed in [6], the encoder outputs 4096 counts for every full revolution. To measure the load gear angle, set the *Encoder Calibration (deg/count)* gain block to 360 / 4096 degrees per count.

12. The measurement will be very small. Click on the *Autoscale* icon in the scope to zoom up on the signal or adjust the range of the y-axis manually. The input voltage and position scopes should appear similarly as shown in Figure A.16 and Figure A.17. Note that no further calibration is needed since the encoder position increases when the input voltage goes positive.

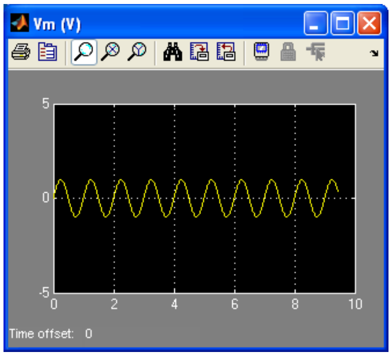


Figure A.16: Sinusoidal input voltage.

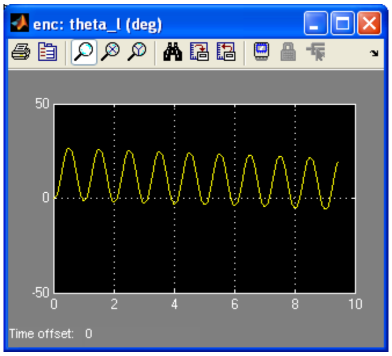


Figure A.17: Position reading using encoder.

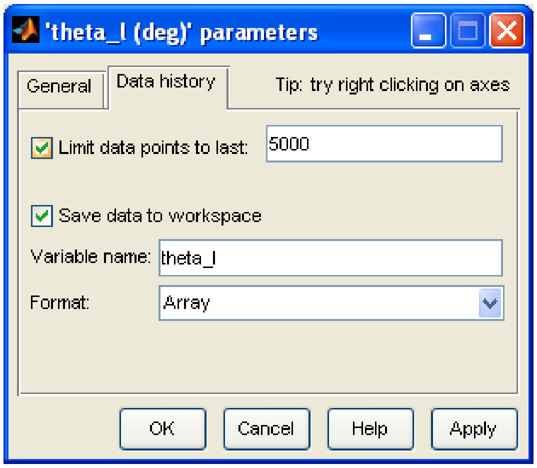
13. Select the QUARC | Stop item to stop the code from running.

14. Power OFF the amplifier if no more experiments will be run in this session.

A.5 Saving Data

The scopes in the Simulink®model can be configured to save variables in the Matlab®workspace. For instance, to configure the load gear position scope *theta_l (deg)* in the Simulink®model from Section A.2 perform the following:

1. Open the *theta_l (deg)* scope.
2. Click on *Parameters* icon and select the *Data History* tab, as shown in Figure A.18.
   - Select the *Save data to workspace* check box.
   - Set the *Variable name* field to a desired variable, e.g. *theta_l*.
   - Set Format to Array.



☑ Limit data points to last: 5000

☑ Save data to workspace

Variable name: theta_l

Format: Array

Figure A.18: Adjusting scope parameters to save data

   Note: By default, the Limit data points to last box is set to 5000. This means that only the last 5000 points of data will be saved in the variable. Thus, given that the controller runs at 500 Hz, this implies that the last 10 seconds of data shown in the *theta_l (deg)* scope will be captured.

3. Click on the OK button.
4. Save the Simulink®model.
5. Select QUARC | Build to rebuild the model.
6. Click on QUARC | Run.
7. Run the controller for a few seconds and stop QUARC.
8. When the controller is stopped, the variable *theta_l* is saved to the workspace. The variable is a two-dimensional array with a maximum of 5000 elements. The first vector is the running time and the second vector is the position of the load gear. You can plot the data into a Matlab figure using a script like:

```
t = theta_l(:,1);
th_l = theta_l(:,2);
plot(t,th_l,'r-');
```

   You can then add Matlab®commands such as *xlabel*, *ylabel*, and *title* to desribe the data and units you are plotting.

   Note: If the controller has not run for the full 10 seconds then, it will have t_f/T_s number of elements, where t_f is the duration of the controller and T_s is the sampling interval. For instance, if you ran QUARC for 4 seconds then there will be $4/0.002 = 2000$ elements.

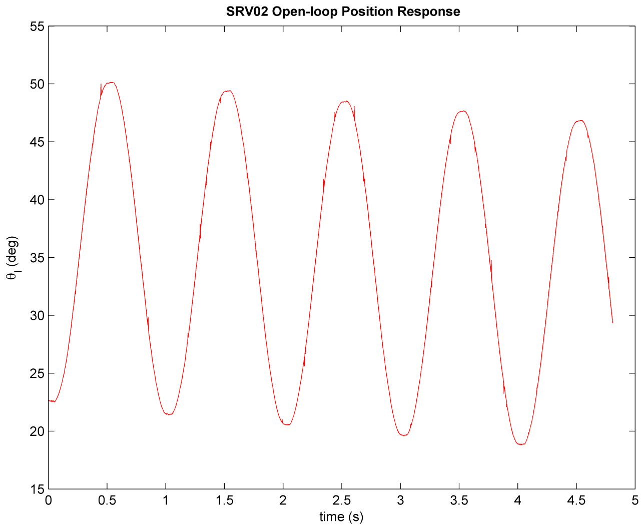


Figure A.19: Plotting saved data

9. Running the script generates a Matlab figure as shown in Figure A.19.

   Note: Use the Matlab®command *ginput* to measure points directly from the Matlab figure.

There are many different ways to save data for offline analysis, e.g. saving data to a Matlab MAT file. Go to [5] in the QUARC Basics | Data Collection category for more information.

BIBLIOGRAPHY

[1] Quanser Inc. *QUARC Installation Guide*, 2009.

[2] Quanser Inc. *Q2-USB User's Manual*, 2010.

[3] Quanser Inc. *VoltPAQ User's Manual*, 2010.

[4] Quanser Inc. *QUARC User Manual*, 2011.

[5] Quanser Inc. *SRV02 User Manual*, 2011.

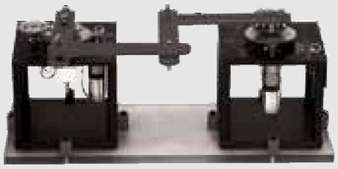
Ten modules to teach controls from the basic to advanced level

SRV02 Base Unit

Flexible Link

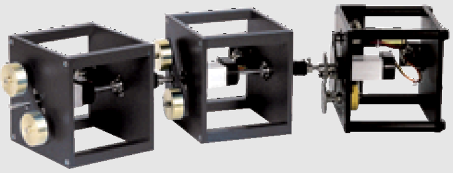
Ball and Beam

2 DOF Robot



2 DOF Inverted Pendulum

Multi-DOF Torsion



Flexible Joint

Gyro/Stable Platform

Double Inverted Pendulum

2 DOF Ball Balancer

With the SRV02 Base Unit, you can select from 10 add-on modules to create experiments of varying complexity across a wide range of topics, disciplines and courses. All of the experiments/workstations are compatible with LabVIEW™ and MATLAB®/Simulink®.

To request a demonstration or a quote, please email info@quanser.com.

QUANSER
INNOVATE. EDUCATE.

INFO@QUANSER.COM +1-905-940-3575 QUANSER.COM

Solutions for teaching and research. Made in Canada.