QUANSER
INNOVATE. EDUCATE.

# STUDENT WORKBOOK

## Flexible Joint Experiment for MATLAB®/Simulink® Users

Standardized for ABET* Evaluation Criteria

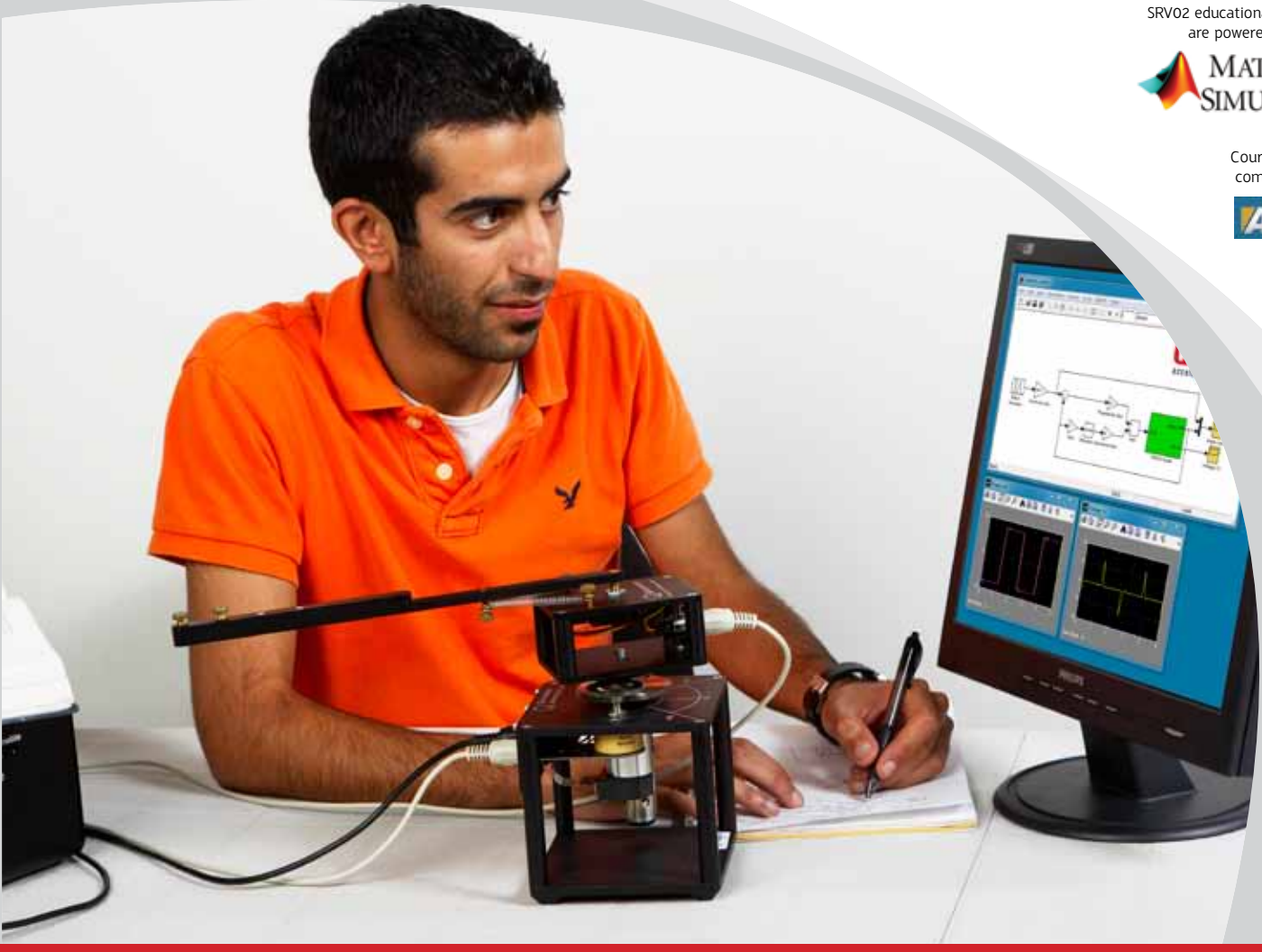
Developed by:
Jacob Apkarian, Ph.D., Quanser
Paul Karam, B.A.SC., Quanser
Michel Lévis, M.A.SC., Quanser

SRV02 educational solutions are powered by:

MATLAB SIMULINK

Course material complies with:

ABET



**CAPTIVATE. MOTIVATE. GRADUATE.**

*ABET Inc., is the recognized accreditor for college and university programs in applied science, computing, engineering, and technology. Among the most respected accreditation organizations in the U.S., ABET has provided leadership and quality assurance in higher education for over 75 years.

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 his help to include embedded outcomes assessment.

# CONTENTS

# 1 INTRODUCTION

The objective of this experiment is to control the position of the servo while minimizing the motions the flexible rotary link.

**Topics Covered**

- Modeling the Rotary Flexible Joint using Lagrange.
- Find the linear state-space model of the system.
- Do some basic model validation.
- Design a state-feedback controller using Pole-Placement (PP).
- Simulate the closed-loop flexible joint system.
- Implement the designed controller on the device.
- Compare the simulated and measured closed-loop results.
- Assess the behaviour of implementing a partial-state feedback controller.

**Prerequisites**

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

- Basics of Simulink®.
- Transfer function fundamentals.
- State-space modeling, e.g., obtaining state equations from a set of differential equations.
- *SRV02 QUARC Integration Laboratory* ([3]) in order to be familiar using QUARC®with the SRV02.

# 2 MODELING

## 2.1 Background

### 2.1.1 Model

The Rotary Flexible Joint model is shown in Figure 2.1. The base of the module is mounted on the load gear of the SRV02 system. The servo angle, $\theta$, increases positively when it rotates counter-clockwise (CCW). The servo (and thus the link) turn in the CCW direction when the control voltage is positive, i.e., $V_m > 0$.

The total length of the link can be varied by changing the mounting position of the shorter top arm. The main bottom arm, which is connected to the pivot, has a length of $L_1$ and a mass of $m_1$. The length and mass of the top link is $L_2$ and $m_2$. The distance between the pivot and the middle of the top arm, which can be changed, is denoted by the variable $d_{12}$. The moment of inertia of the entire link is specified by $J_l$ and it changes depending on the position of the top arm. See the *Rotary Flexible Joint User Manual* (in [7]) for the values of these parameters. The deflection angle of the link is denoted as $\alpha$ and increases positively when rotated CCW.

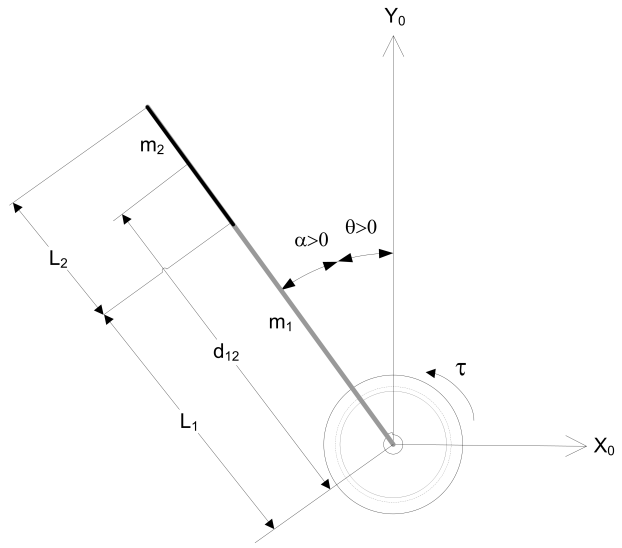



Figure 2.1: Rotary Flexible Joint Angles

The flexible joint system can be represented by the diagram shown in Figure 2.2. Our control variable is the input servo motor voltage, $V_m$. This generates a torque, $\tau$, at the load gear of the servo that rotates the base of the link. The viscous friction coefficient of the servo is denoted by $B_{eq}$. This is the friction that opposes the torque being applied at the servo load gear. The friction acting on the link is represented by the viscous damping coefficient $B_l$. Finally, the flexible joint is modeled as a linear spring with the stiffness $K_s$.

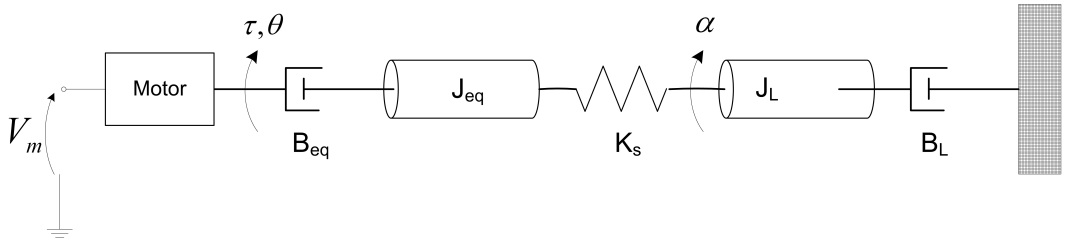



Figure 2.2: Rotary Flexible Joint Model

### 2.1.2 Finding the Equations of Motion

Instead of using classical mechanics, the Lagrange method is used to find the equations of motion of the system. This systematic method is often used for more complicated systems such as robot manipulators with multiple joints.

More specifically, the equations that describe the motions of the servo and the link with respect to the servo motor voltage, i.e. the dynamics, will be obtained using the Euler-Lagrange equation:

$$\frac{\partial^2 L}{\partial t \partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i \tag{2.1}$$

The variables $q_i$ are called *generalized coordinates*. For this system let

$$q(t)^\top = [\theta(t)\ \alpha(t)] \tag{2.2}$$

where, as shown in Figure 2.2, $\theta(t)$ is the servo angle and $\alpha(t)$ is the flexible joint angle. The corresponding velocities are

$$\dot{q}(t)^\top = \left[\frac{\partial \theta(t)}{\partial t}\ \frac{\partial \alpha(t)}{\partial t}\right]$$

**Note:** The dot convention for the time derivative will be used throughout this document, i.e., $\dot{\theta} = \frac{d\theta}{dt}$ and $\dot{\alpha} = \frac{d\alpha}{dt}$. The time variable $t$ will also be dropped from $\theta$ and $\alpha$, i.e., $\theta := \theta(t)$ and $\alpha := \alpha(t)$.

With the generalized coordinates defined, the Euler-Lagrange equations for the rotary flexible joint system are

$$\frac{\partial^2 L}{\partial t \partial \dot{\theta}} - \frac{\partial L}{\partial \theta} = Q_1 \tag{2.3}$$

and

$$\frac{\partial^2 L}{\partial t \partial \dot{\alpha}} - \frac{\partial L}{\partial \alpha} = Q_2 \tag{2.4}$$

The Lagrangian of a system is defined

$$L = T - V \tag{2.5}$$

where $T$ is the total kinetic energy of the system and $V$ is the total potential energy of the system. Thus the Lagrangian is the difference between a system's kinetic and potential energies.

The generalized forces $Q_i$ are used to describe the non-conservative forces (e.g., friction) applied to a system with respect to the generalized coordinates. In this case, the generalized force acting on the rotary arm is

$$Q_1 = \tau - B_{eq}\dot{\theta} \tag{2.6}$$

and acting on the link is

$$Q_2 = -B_l \dot{\alpha}. \tag{2.7}$$

The torque applied at the base of the rotary arm (i.e., at the load gear) is generated by the servo motor as described by the equation

$$\tau = \frac{\eta_g K_g \eta_m k_t (V_m - K_g k_m \dot{\theta})}{R_m}. \tag{2.8}$$

See [5] for a description of the corresponding SRV02 parameters (e.g. such as the back-emf constant, $k_m$). The servo damping (i.e. friction), $B_{eq}$, opposes the applied torque. The flexible joint is not actuated, the only force acting on the link is the damping, $B_l$.

Again, the Euler-Lagrange equations is a systematic method of finding the equations of motion (EOMs) of a system. Once the kinetic and potential energy are obtained and the Lagrangian is found, then the task is to compute various derivatives to get the EOMs.

### 2.1.3 Potential and Kinetic Energy

**Kinetic Energy**
Translational kinetic equation is defined as

$$T = \frac{1}{2}mv^2, \tag{2.9}$$

where $m$ is the mass of the object and $v$ is the linear velocity.

Rotational kinetic energy is described as

$$T = \frac{1}{2}J\omega^2 \tag{2.10}$$

where $J$ is the moment of inertia of the object and $\omega$ is its angular rate.

**Potential Energy**
Potential energy comes in different forms. Typically in mechanical system we deal with *gravitational* and *elastic* potential energy. The *relative* gravitational potential energy of an object is

$$V_g = mg\Delta h, \tag{2.11}$$

where $m$ is the object mass and $\Delta h$ is the change in altitude of the object (from a reference point). The potential energy of an object that rises from the table surface (i.e., the reference) up to 0.25 meter is $\Delta h = 0.25 - 0 = 0.25$ and the energy stored is $V_g = 0.25mg$.

The equation for elastic potential energy, i.e., the energy stored in a spring, is

$$V_e = \frac{1}{2}K\Delta x^2 \tag{2.12}$$

where $K$ is the spring stiffness and $\Delta x$ is the linear or angular change in position. If an object that is connected to a spring moves from in its initial reference position to $0.1$ m, then the change in displacement is $\Delta x = 0.1 - 0 = 0.1$ and the energy stored equals $V_e = 0.005K$.

### 2.1.4 Linear State-Space Model

The linear state-space equations are

$$\dot{x} = Ax + Bu \tag{2.13}$$

and

$$y = Cx + Du \tag{2.14}$$

where $x$ is the state, $u$ is the control input, $A$, $B$, $C$, and $D$ are state-space matrices. For the Rotary Flexible Joint system, the state and output are defined

$$x^\top = [\theta\ \alpha\ \dot{\theta}\ \dot{\alpha}] \tag{2.15}$$

and

$$y^\top = [x_1\ x_2]. \tag{2.16}$$

In the output equation, only the position of the servo and link angles are being measured. Based on this, the $C$ and $D$ matrices in the output equation are

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.17}$$

and

$$D = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{2.18}$$

The velocities of the servo and link angles can be computed in the digital controller, e.g., by taking the derivative and filtering the result though a high-pass filter.

### 2.1.5 Finding Second Order System Parameters

Consider a second-order system described by

$$J\ddot{x} + B\dot{x} + Kx = 0. \tag{2.19}$$

Assuming the initial conditions $x(0^-) = x_0$ and $\dot{x}(0^-) = 0$, the Laplace transform of Equation 2.19 is

$$X(s) = \frac{\frac{x_0}{J}}{s^2 + \frac{B}{J}s + \frac{K}{J}} \tag{2.20}$$

The prototype second-order equation is defined

$$s^2 + 2\zeta\omega_n s + \omega_n^2,$$

where $\zeta$ is the damping ratio and $\omega_n$ is the natural frequency. Equating the characteristic equation in 2.20 to this gives

$$\omega_n^2 = \frac{K}{J}$$

and

$$2\zeta\omega_n = \frac{B}{J}$$

Based on the measured damping ratio and natural frequency, the friction (or stiffness) of the system is

$$K = J\omega_n^2 \tag{2.21}$$

and the viscous damping is

$$B = 2\zeta\omega_n J. \tag{2.22}$$

The natural frequency and damping ratio of a system can be found experimentally, e.g., from a free-oscillation response or frequency response.

### 2.1.6 Power Spectrum

Fourier is a way to represent a signal in terms of sinusoidals. The Fourier transform, or spectrum, of a signal $g(t)$ is denoted as $G(\omega)$ and it shows the relative amplitudes and frequencies of the sinusoidals that are used to represent that signal [9].

The Fourier transform contains both the magnitude and phase information. The *power spectrum* of a signal is based on the magnitude of the signal. Figure 2.3, for example, show the power spectrum of the compound sine wave signal

$$g(t) = 3\sin 2\pi t + 2\sin 4\pi t + 0.5\sin 10\pi t.$$

The spectrum shows the peaks occurring at the sine wave frequencies 1, 2, and 5 Hz. Similarly, this can be used to find the resonant frequencies of an actual system.

The power of a signal is the time-average of the its squared value [9] and, for a continuous signal $g(t)$, is defined

$$P_g = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} g(t)^2 dt.$$

We want to define the power of the signal from its transform. First consider the truncated signal of $g(t)$ defined as

$$g_T(t) = \begin{cases} g(t) & |t| \le T \\ 0 & |t| > T \end{cases}$$

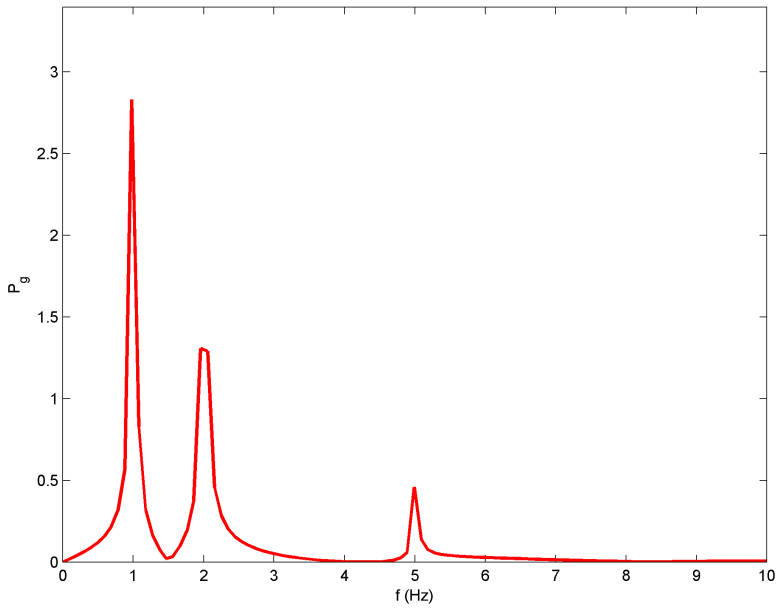



Figure 2.3: Power spectrum of compound sine wave

From Parseval's theorem, the energy of this signal equals [9]

$$E_{g_T} = \int_{-\infty}^{\infty} g_T(t)^2 dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} |G_T(\omega)|^2 d\omega$$

The power of a signal can then be expressed as

$$\begin{aligned} P_g &= \lim_{T\to\infty} \frac{E_{g_T}}{T} \\ &= \frac{1}{2\pi} \lim_{T\to\infty} \int_{-\infty}^{\infty} \frac{|G_T(\omega)|^2}{T} d\omega. \end{aligned}$$

The *power spectral density (PSD)* function is

$$S_g(\omega) = \lim_{T\to\infty} \frac{|G_T(\omega)|^2}{T} \tag{2.23}$$

Defining the signal power in terms of the PSD and considering only positive frequencies we get

$$\begin{aligned} P_g &= \frac{1}{2\pi} \int_{-\infty}^{\infty} S_g(\omega) d\omega \\ &= \frac{1}{\pi} \int_{0}^{\infty} S_g(\omega) d\omega \end{aligned}$$

Expressing this in term of Hertz we get the expression

$$P_g = 2 \int_{0}^{\infty} S_g(\omega) df \tag{2.24}$$

In practice, signals are sampled and the algorithm is performed in discrete mode. The Fast-Fourier Transform (FFT) is a computational algorithm used to find the Fourier transform of a signal, i.e., taking the FFT of a signal $g(t)$ generates $G(\omega)$. To find the power spectrum, code similar to the following would be used

```
y = fft(g);
Sg = |y|/N;
Pg = 2*Sg(1:N/2);
```

where $N$ is the number of samples, or length, of the signal $g(t)$. If $g(t)$ was sampled at a frequency interval of $T_s$ and had a duration of $T$, it would have $N = T/T_s$ samples. Notice that because this is implemented in discrete mode, we use the amount of samples, $N$, instead of the signal time duration, $T$, as used in the PSD expression in Equation 2.23.

The power spectrum can be used to find resonant frequencies of a system. One common technique is feeding a sine sweep (or chirp) signal to the system and measuring the corresponding output response. By taking its FFT and finding the spectrum, the user will see various peaks. These peaks represent the sine wave frequencies that describe the system.

## 2.2 Pre-Lab Questions

1. Energy is stored in the flexible joint springs as it rotates by an angle of $\alpha$ (see Figure 2.1). Find the potential energy of the flexible joint using the parameters described in Section 2.1.
2. Find the total kinetic energy of the system contributed by the rotary servo, $\theta$, and the rotation of the link, $\alpha$. Use the parameters shown in Figure 2.2.
3. Compute the Lagrangian of the system.
4. Find the first Euler-Lagrange equation given in 2.3. Keep the equations in terms of applied torque, $\tau$ (i.e., not in terms of DC motor voltage). Also make sure your equations follow the general form: $J\ddot{x} + B\dot{x} + Kx = u$.
5. Find the second Euler-Lagrange Equation 2.4.
6. Find the equations of motion: $\ddot{\theta} = f_1(\theta, \dot{\theta}, \alpha, \dot{\alpha}, \tau)$ and $\ddot{\alpha} = f_2(\theta, \dot{\theta}, \alpha, \dot{\alpha}, \tau)$. Assume the viscous damping of the link is negligible, i.e., $B_l = 0$.
7. Given state $x$ defined in Equation 2.15, find the linear state-space matrices $A$ and $B$.
8. Express the moment of inertia of the rotary arm in terms of the length and masses given in Figure 2.1. As already mentioned, the total arm length varies depending on where the short arm is mounted on top of the long arm. The inertia can be computed in the lab once you know how your ROTFLEX is configured.

   **Hint:** The total inertia depends on where the top arm is anchored. Use the parallel-axis theorem to determine the inertia of the top link with respect to the pivot.

## 2.3 Lab Experiments

In the first part of this laboratory, the stiffness of the flexible joint is determined by measuring its natural frequency. In the second part, the state-space model is finalized and validated against actual measurements.

### 2.3.1 Finding Stiffness

To find the stiffness we need to find the natural frequency of the flexible joint. This is the frequency where the link attached to the flexible joint begins to oscillate the most. To find this frequency, we use a Sine Sweep signal. The Sine Sweep is a sine wave that goes through a range of frequencies, i.e., from low to high. We can then generate a power spectrum from the measured signal and identify the frequency with the largest amplitude - the natural frequency.

**Physical Parameters for the Lab**

In order to do some of the laboratory exercises, you will need these values:

$$
\begin{aligned}
B_{eq} &= 0.004 \text{ N} \cdot \text{m / (rad/s)} \\
J_{eq} &= 2.08 \times 10^{-3} \text{ kg} \cdot \text{m}^2 \\
m_1 &= 0.064 \text{ kg} \\
L_1 &= 0.298 \text{ m} \\
m_2 &= 0.030 \text{ kg} \\
L_2 &= 0.156 \text{ m} \\
B_l &= 0
\end{aligned}
$$

**Note:** The equivalent viscous damping, $B_{eq}$, and moment of inertia, $J_{eq}$, parameters are for the SRV02 when there is no load (i.e., the parameter found in the SRV02 Modeling Laboratory was for servo with the disc load).

**Experimental Setup**

The q_rotflex_id Simulink diagram shown in Figure 2.4 is used to find the natural frequency of the flexible joint. The QUARC blocks are used to interface with the DC motor and encoders of the system. The Sine Sweep block generates a sine wave that goes from a low frequency to a high frequency for a predetermined duration. For more information about the QUARC® software, see Reference [3]. This model outputs the deflection angle of the link.

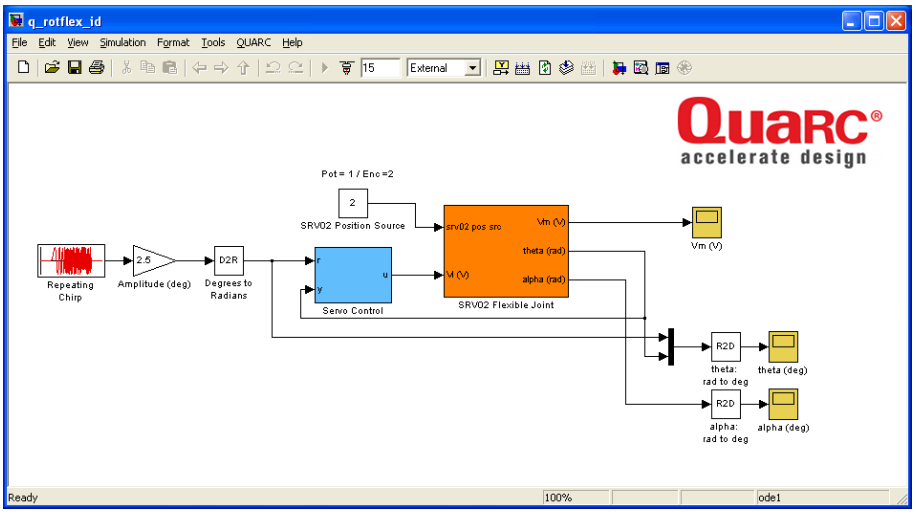



Figure 2.4: q_rotflex_id Simulink diagram used to find flexible joint stiffness

**IMPORTANT:** Before you can conduct this experiment, you need to make sure that the lab files are configured

according to your system setup. If they have not been configured already, then you need to go to Section 4.2 to configure the lab files first.

1. In the *q_rotflex_id* Simulink diagram, go to QUARC | Build to build the QUARC controller.
2. Go to QUARC | Start to run the controller. The SRV02 tracks a sine sweep that goes from 1 Hz to 5 Hz. The *alpha (deg)* Scope should be reading a response similarly as shown in Figure 2.5. Note that the controller is set to run for 15 seconds.

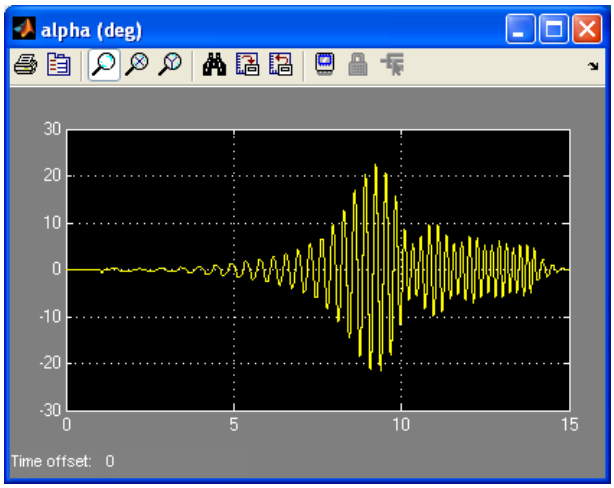



Figure 2.5: Typical Flexible Joint Sine Sweep Response

3. After the controller stops (i.e., after 15 sec), the data is automatically saved in the Matlab workspace to the variable *data_alpha*. The time is stored in *data_alpha(:,1)* vector and the link angle is stored the *data_alpha(:,2)* vector. Plot the response in a Matlab figure.
4. Using the sweep response data and Matlab commands, generate a plot of the power spectrum. The background theory and *pseudo code* to find the power spectrum was given in Section 2.1.6. You can also use any available Matlab examples to help you out. Show the commands you used and the plot generated.

   **Hint:** When using the discrete Matlab FFT command *fft(x,n)*, the size of $n$ should be base of 2. Use the *nextpow2* function to find this size. For example, if your signal size is 250 samples long, this will return 8 which gives $n = 2^8 = 256$.
5. Find the natural frequency of the link. Because the damping in relatively low, assume the damped natural frequency (which is being measured) is equivalent to the undamped natural frequency.
6. Find the total moment of inertia of the link, $J_l$, using the equation you developed in the Exercise 8 in Section 2.2. The corresponding value of $d_{12}$ is given in the Rotary Flexible Joint User Manual [7]. Once the inertia is computed, find the stiffness of the flexible joint, $K_s$.

## 2.3.2 Model Validation

By running a simulation and the actual device in parallel, we can verify whether the dynamic model (which drives the simulation) accurately represents our system.

**Experimental Setup**

The q_rotflex_val Simulink diagram shown in Figure 2.6 applies either a step or pulse input to both the Quanser Flexible Joint hardware and to the Flexible Joint model and reads the servo and link angles. The *SRV02 Flexible Joint* subsystem contains the QUARC blocks that interface to the actual hardware. The Simulink *State-Space* block reads the $A$, $B$, $C$, and $D$ state-space matrices that are loaded in the Matlab workspace. This model outputs the deflection angle of the link.

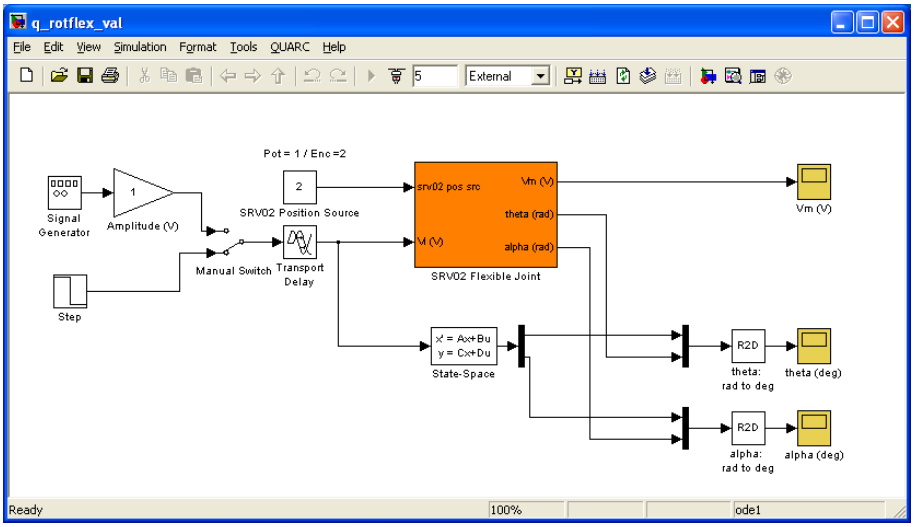



Figure 2.6: q_rotflex_val Simulink diagram used validate the model

**IMPORTANT:** Before you can conduct this experiment, you need to make sure that the lab files are configured according to your system setup. If they have not been configured already, then go to Section 4.3 to configure the lab files first.

1. Run the *setup_rotflex.m*.
2. When prompted, enter the stiffness you found in Section 2.3.1. This is saved to the Matlab variable *Ks*.

   ```
   Enter link stiffness (Ks):
   ```

3. Depending on your stiffness, the Matlab prompt should generate a gain similarly as shown below (this gain is generated for a *Ks* of 1):

   ```
   K =

         0     0     0     0


   cls_poles =

         0     0     0     0
   ```

   This means the script ran correctly.

4. In Matlab, open the M-File called *ROTFLEXE_ABCD_eqns_student.m*. The script has the following state-space matrices entered:

   ```
   A = [0 0 1 0;
       0 0 0 1;
       0 500 -5 0;
       0 -750 5 0];

   B = [0 0 500 -500]';

   C = zeros(2,4);

   D = zeros(2,1);
   ```

5. Enter the state-space matrices you found in Section 2.2 for $A$, $B$, $C$, and $D$. In Matlab, the stiffness and link moment of inertia are defined as *Ks* and *Jl* and the SRV02 inertia and damping are denoted *Jeq* and *Beq*.

6. Run the *ROTFLEX_ABCD_eqns_student.m* script to load the state-space matrices in the Matlab workspace. Show the numerical matrices that are displayed in the Matlab prompt.

7. The input of the state-space model you found in Section 2.2 is the torque acting at the servo load gear (i.e., at the pivot of the flexible joint). However, *we do not control torque directly - we control the servo input voltage*. Recall the voltage-torque relationship given in Equation 2.8 in Section 2.1.2. In the *System Model* section of the setup_rotflex.m script, the actuator dynamics are added to your state-space matrices with the code:

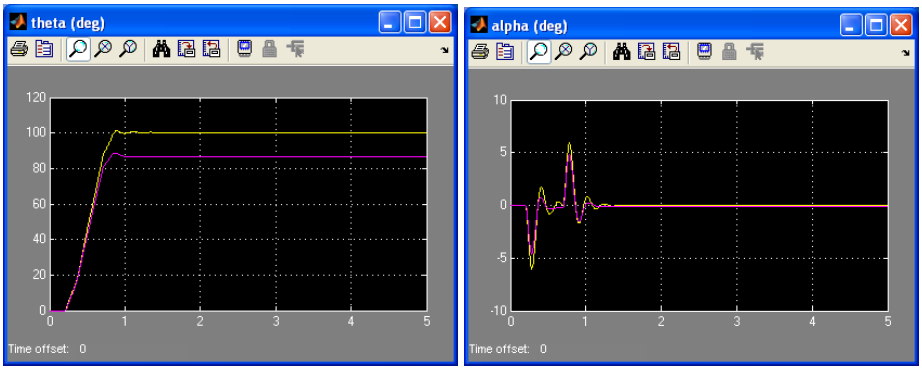
```
Ao = A;
Bo = B;
B = eta_g*Kg*eta_m*kt/Rm*Bo;
A(3,3) = Ao(3,3) - Bo(3)*eta_g*Kg^2*eta_m*kt*km/Rm;
A(4,3) = Ao(4,3) - Bo(4)*eta_g*Kg^2*eta_m*kt*km/Rm;
```

8. Run the setup_rotflex.m script again so your Flexible Joint model is based on DC motor voltage.

9. In the *q_rotflex_val* Simulink diagram, go to QUARC | Build to build the QUARC controller.

10. Make sure the area around the Flexible Joint experiment is clear. Set the *Manual Switch* to the downward position to feed a Step input.

11. Go to QUARC | Start to run the q_rotflex_val controller. Typical scope responses are shown in Figure 2.7. The *theta (deg)* scope displays the simulated servo angle in yellow and the measured angle in purple. Similarly, the *alpha (deg)* scope shows the simulated link angle in yellow and the measured angle in purple.




(a) Servo Angle (b) Flexible Link Angle

Figure 2.7: Typical Flexible Joint model validation response in scopes

12. If your simulation and measured response match, go to the next step. If they do not, then there is an issue with your model. Here are some issues to investigate:

    - Ensure the state-space model was entered properly in the script.
    - The stiffness, $K_s$, found in Section 2.3.1 is not correct. Review your calculations or redo the experiment.
    - Review your model derivation in Section 2.2, e.g., might be a mistake in solving the EOMs.

13. The *theta (deg)* and *alpha (deg)* scopes save their data to the variables *data_theta* and *data_alpha*. For *data_theta*, the time is in *data_theta(:,1)*, the simulated servo angle is in *data_theta(:,2)*, and the measured SRV02 angle is in *data_theta(:,3)*. Similarly for the flexible joint response saved in *data_alpha*. Plot the response in a Matlab figure.

14. How well does your model represent the actual system? We want a model that is fairly representative but, having said that, keep in mind that no model is perfect. This is just a quick test to see how well your model represents the actual device. As shown in figures 2.7a and 2.7b, the simulation does not match the measured response perfectly.

15. Give at least one reason why the model does not represent the system accurately.

16. In Matlab, find the open-loop poles (i.e., eigenvalues) of the system using the state-space matrix $A$ that is loaded.

    **Note:** These will be required for a pre-lab question in Section 3.3.

## 2.4 Results

Fill out Table 1 with your answers from your modeling lab results - both simulation and implementation.

| Description | Symbol | Value | Unit |
|---|---|---|---|
| **Finding Stiffness** | | | |
| Natural frequency | $\omega_n$ | | rad/s |
| Stiffness | $K_s$ | | N m/rad |
| **Model Validation** | | | |
| State-Space Matrix | A | | |
| State-Space Matrix | B | | |
| State-Space Matrix | C | | |
| State-Space Matrix | D | | |
| Open-loop poles | OL | | |

Table 1: Results

# 3 CONTROL DESIGN

## 3.1 Specifications

The time-domain requirements are:

**Specification 1:** Servo angle 4% settling time: $t_s <= 0.5$ s.

**Specification 2:** Servo angle percentage overshoot: $PO <= 5$ %.

**Specification 3:** Maximum link angle deflection: $|\alpha| <= 12.5$ deg.

**Specification 4:** Maximum control effort / voltage: $|V_m| <= 10$ V.

Desired closed-loop poles:

- Damping ratio: $\zeta = 0.6$.
- Natural frequency: $\omega_n = 20$ rad/s.
- Non-dominant poles: $\{p_3 = -30, p_4 = -40\}$.

Design a feedback controller that places the poles of the closed-loop system to the desired locations indicated. The servo settling time, overshoot, link deflection, and control effort time-domain requirements are to be satisfied when the servo is tracking a $\pm 20$ degree angle square wave. Note for the settling time specifications, the servo angle must reach 4% of its final, steady-state angle in the alloted time. For instance if the measured steady-state servo angle is 20 degrees, then it must settle within $20 \pm 0.8$ degrees in 0.5 seconds.

## 3.2 Background

In Section 2.2, we found a linear state-space model that represents the Rotary Flexible Joint system. This model is used to investigate the stability properties of the Flexible Joint system in Section 3.2.1. In Section 3.2.2, the notion of controllability is introduced. The procedure to transform matrices to their companion form is described in Section 3.2.3. Once in their companion form, it is easier to design a gain according to the pole-placement principles, which is discussed in Section 3.2.4. Lastly, Section 3.2.6 describes the state-feedback control used to control the servo position while minimizing link deflection that is introduced by the flexible joint.

### 3.2.1 Stability

The stability of a system can be determined from its poles ([10]):

- Stable systems have poles only in the left-hand plane.
- Unstable systems have at least one pole in the right-hand plane and/or poles of multiplicity greater than 1 on the imaginary axis.
- Marginally stable systems have one pole on the imaginary axis and the other poles in the left-hand plane.

The poles are the roots of the system's characteristic equation. From the state-space, the characteristic equation of the system can be found using

$$\det(sI - A) = 0 \tag{3.1}$$

where *det()* is the determinant function, $s$ is the Laplace operator, and $I$ the identity matrix. These are the *eigenvalues* of the state-space matrix $A$.

### 3.2.2 Controllability

If the control input, $u$, of a system can take each state variable, $x_i$ where $i = 1 \dots n$, from an initial state to a final state then the system is controllable, otherwise it is uncontrollable ([10]).

**Rank Test** The system is controllable if the rank of its controllability matrix

$$T = \begin{bmatrix} B & AB & A^2B \dots A^nB \end{bmatrix} \tag{3.2}$$

equals the number of states in the system,

$$\text{rank}(T) = n. \tag{3.3}$$

### 3.2.3 Companion Matrix

If $(A, B)$ are controllable and $B$ is $n \times 1$, then $A$ is similar to a companion matrix ([1]). Let the characteristic equation of $A$ be

$$s^n + a_n s^{n-1} + \dots + a_1.$$

Then the companion matrices of A and B are

$$\tilde{A} = \begin{bmatrix} 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ -a_1 & -a_2 & \cdots & -a_{n-1} & -a_n \end{bmatrix} \tag{3.4}$$

and

$$\tilde{B} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{3.5}$$

Define

$$W = T\tilde{T}^{-1}$$

where $T$ is the controllability matrix defined in Equation 3.2 and

$$\tilde{T} = [\tilde{B}\ \tilde{B}\tilde{A} \dots \tilde{B}\tilde{A}^n].$$

Then

$$W^{-1}AW = \tilde{A}$$

and

$$W^{-1}B = \tilde{B}.$$

### 3.2.4 Pole Placement

If (A,B) are controllable, then pole placement can be used to design the controller. Given the control law $u = -Kx$, the state-space in Equation 2.13 becomes

$$\begin{aligned} \dot{x} &= Ax + B(-Kx) \\ &= (A - BK)x \end{aligned}$$

To illustate how to design gain $K$, consider the following system

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 3 & -1 & -5 \end{bmatrix} \tag{3.6}$$

and

$$B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.7}$$

Note that A and B are already in the companion form. We want the closed-loop poles to be at $[-1 \; -2 \; -3]$. The *desired* characteristic equation is therefore

$$(s+1)(s+2)(s+3) = s^3 + 6s^2 + 11s + 6 \tag{3.8}$$

For the gain $K = [k_1 \; k_2 \; k_3]$, apply control $u = -Kx$ and get

$$A - KB = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 3 - k_1 & -1 - k_2 & -5 - k_3 \end{bmatrix}.$$

The characteristic equation of $A - KB$ is

$$s^3 + (k_3 + 5)s^2 + (k_2 + 1)s + (k_1 - 3) \tag{3.9}$$

Equating the coefficients between Equation 3.9 and the desired polynomial in Equation 3.8

$$\begin{aligned} k_1 - 3 &= 6 \\ k_2 + 1 &= 11 \\ k_3 + 5 &= 6 \end{aligned}$$

Solving for the gains, we find that a gain of $K = [9 \; 10 \; 1]$ is required to move the poles to their desired location.

We can generalize the procedure to design a gain $K$ for a controllable (A,B) system as follows:

**Step 1** Find the companion matrices $\tilde{A}$ and $\tilde{B}$. Compute $W = T\tilde{T}^{-1}$.

**Step 2** Compute $\tilde{K}$ to assign the poles of $\tilde{A} - \tilde{B}\tilde{K}$ to the desired locations. Applying the control law $u = -Kx$ to the general system given in Equation 3.4,

$$\tilde{A} = \begin{bmatrix} 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ -a_1 - k_1 & -a_2 - k_2 & \cdots & -a_{n-1} - k_{n-1} & -a_n - k_n \end{bmatrix} \tag{3.10}$$

**Step 3** Find $K = \tilde{K}W^{-1}$ to get the feedback gain for the original system (A,B).

**Remark:** It is important to do the $\tilde{K} \rightarrow K$ conversion. Remember that (A,B) represents the actual system while the companion matrices $\tilde{A}$ and $\tilde{B}$ do not.

### 3.2.5 Desired Poles

The rotary inverted pendulum system has four poles. As depicted in Figure 3.1, poles $p_1$ and $p_2$ are the complex conjugate *dominant* poles and are chosen to satisfy the natural frequency, $\omega_n$, and damping ratio, $\zeta$, specifications given in Section 3.1. Let the conjugate poles be

$$p_1 = -\sigma + j\omega_d \tag{3.11}$$

and

$$p_2 = -\sigma - j\omega_d \tag{3.12}$$

where $\sigma = \zeta\omega_n$ and $\omega_d = \omega_n\sqrt{1-\zeta^2}$ is the *damped* natural frequency. The remaining closed-loop poles, $p_3$ and $p_4$, are placed along the real-axis to the left of the dominant poles, as shown in Figure 3.1.

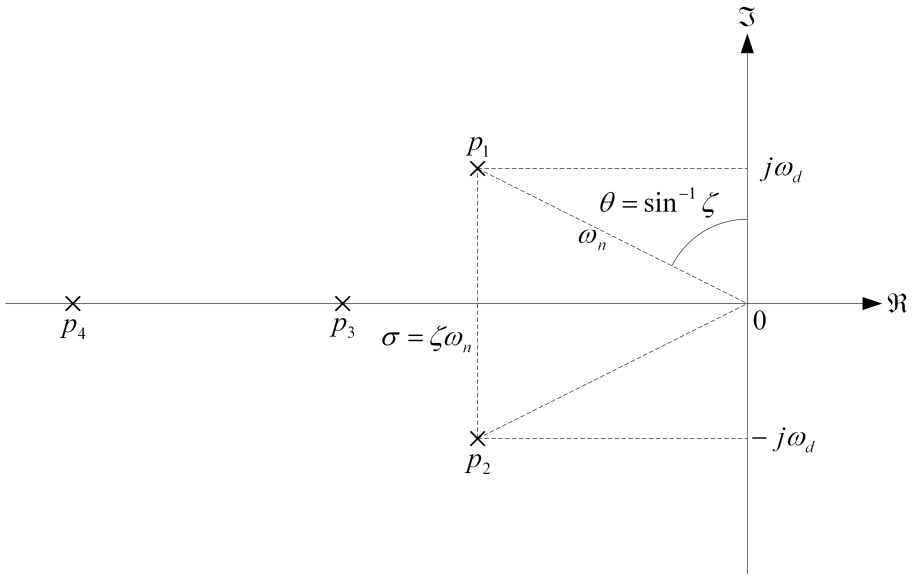



Figure 3.1: Desired closed-loop pole locations

### 3.2.6 Feedback Control

The feedback control loop that in Figure 3.2 is designed to stabilize the servo to a desired position, $\theta_d$, while minimizing the deflection of the flexible link.

The reference state is defined

$$x_d = [\theta_d \; 0 \; 0 \; 0] \tag{3.13}$$

and the controller is

$$u = K(x_d - x). \tag{3.14}$$

Note that if $x_d = 0$ then $u = -Kx$, which is the control used in the control algorithm.

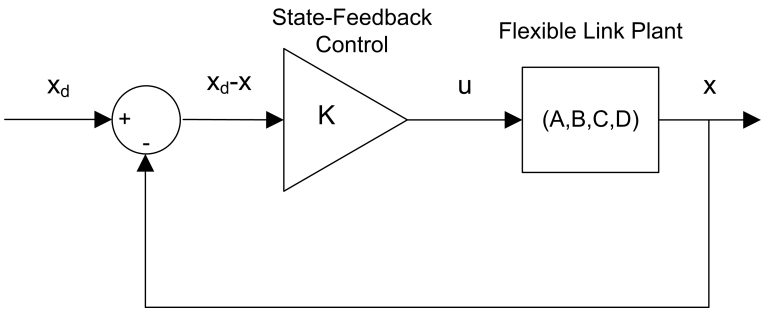



Figure 3.2: State-feedback control loop

## 3.3 Pre-Lab Questions

1. Based on your analysis in Section 2.3, is the system stable, marginally stable, or unstable? Did you expect the stability of the Rotary Flexibe Joint system to be as what was determined? If desired, use your experience with the actual device in the Modeling Laboratory in Section 2.3.
2. Using the open-loop poles, find the characteristic equation of the system.
3. Give the corresponding companion matrices $\tilde{A}$ and $\tilde{B}$. Do not compute the transformation matrix $W$ (this will be computed using software in the lab).
4. Find the location of the two dominant poles, $p_1$ and $p_2$, based on the specifications given in Section 3.1. Place the other poles at $p_3 = -30$ and $p_4 = -40$. Finally, give the *desired* characteristic equation.
5. When applying the control $u = -\tilde{K}x$ to the companion form, it changes $(\tilde{A}, \tilde{B})$ to $(\tilde{A} - \tilde{B}\tilde{K}, \tilde{B})$. Find the gain $\tilde{K}$ that assigns the poles to their new desired location.

## 3.4 Lab Experiments

### 3.4.1 Control Design

1. Run the *setup_rotflex.m* script to load the model you found in the Modeling Experiment in Section 2.3.
2. Using Matlab commands, determine if the system is controllable. Explain why.
3. Open the *d_pole_placement_student.m* script. As shown below, the companion matrices $\tilde{A}$ and $\tilde{B}$ for the model are automatically found (denoted as *Ac* and *Bc* in Matlab).

```
% Characteristic equation: s^4 + a_4*s^3 + a_3*s^2 + a_2*s + a_1
a = poly(A);
%
% Companion matrices (Ac, Bc)
Ac = [  0 1 0 0;
        0 0 1 0;
        0 0 0 1;
        -a(5) -a(4) -a(3) -a(2)];
%
Bc = [0; 0; 0; 1];
% Controllability
T = 0;
% Controllability of companion matrices
Tc = 0;
% Transformation matrices
W = 0;
```

   In order to find the gain $K$, we need to find the transformation matrix $W = T\tilde{T}^{-1}$ (note: $\tilde{T}$ is denoted as $Tc$ in Matlab). Modify the *d_pole_placement_student.m* script to calculate the controllability matrix $T$, the companion controllabilty matrix $Tc$, the inverse of $Tc$, and $W$. Show your completed script and the resulting $T$, $Tc$, $Tc^{-1}$, and $W$ matrices.

4. Enter the companion gain, $\tilde{K}$, you found in the pre-lab as *Kc* in *d_pole_placement_student.m* and modify it to find gain $K$ using the transformation detailed in Section 3.2. Run the script again to calculate the feedback gain $K$ and record its value in Table 2.
5. Evaluate the closed-loop poles of the system, i.e., the eigenvalues of $A - BK$. Record the closed-loop poles of the system when using the gain $K$ calculated above. Have the poles been placed to their desired locations? If not, then go back and re-investigate your control design until you find a gain that positions the poles to the required location.
6. In the previous exercises, gain $K$ was found manually through matrix operations. All that work can instead be done using a pre-defined *Compensator Design* Matlab command. Find gain $K$ using a Matlab pole-placement command and verify that the gain is the same as generated before.

### 3.4.2 Control Simulation

Using the linear state-space model of the system and the designed control gain, the closed-loop response can be simulated. This way, we can test the controller and see if it satisfies the given specifications before running it on the hardware platform.

**Experiment Setup**

The *s_rotflex* Simulink diagram shown in Figure 3.3 is used to simulate the closed-loop response of the Flexible Joint using the control developed in Section 3.3.

The *Smooth Signal Generator* block generates a 0.33 Hz square wave (with amplitude of 1) that is passed through a *Rate Limiter* block to smooth the signal. The *Amplitude (deg)* gain block is used to change the desired servo position command. The state-feedback gain $K$ is set in the *Controller* gain block and is read from the Matlab workspace. The Simulink *State-Space* block reads the $A$, $B$, $C$, and $D$ state-space matrices that are loaded in the Matlab workspace.

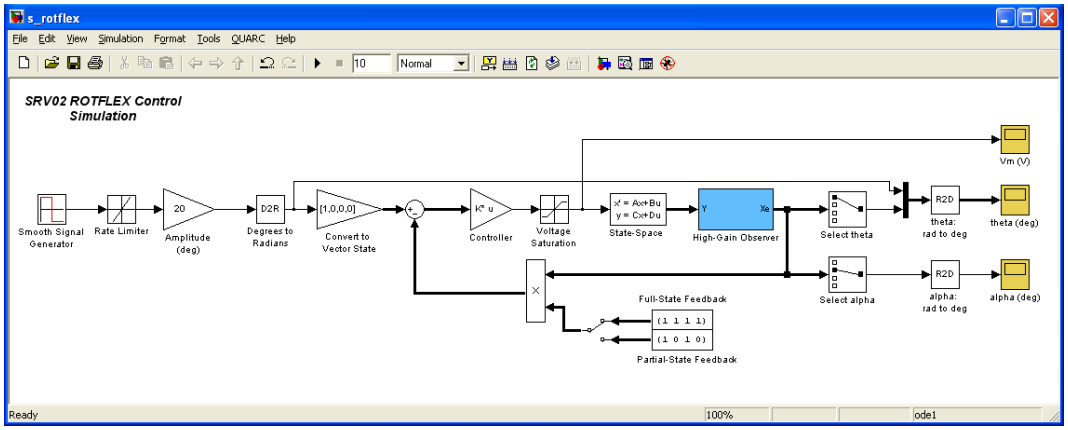



Figure 3.3: s_rotflex Simulink diagram used to simulate the state-feedback control

**IMPORTANT:** Before you can conduct this experiment, you need to make sure that the lab files are configured according to your system setup. If they have not been configured already, go to Section Section 4.4 to configure the lab files first. **Make sure the model you found in Section 2.3.2 is entered in ROTFLEX_ABCD_eqns_student.m and the d_pole_placement_student.m calculates the control gain.**

1. Run *setup_rotpen.m*. Ensure the gain $K$ you found in Section 3.4.1 is loaded.
2. In *s_rotflex*, make sure the *Manual Switch* is set to the *Full-State Feedback* (upward) position.
3. Run *s_rotflex* to simulate the closed-loop response with this gain. The response in the scopes shown in Figure 3.4 were generated using an arbitary gain. Plot the simulated response of the servo, link, and motor input voltage obtained using your obtained gain $K$ in a Matlab figure and attach it to your report.

   **Note:** When the simulation stops, the last 10 seconds of data is automatically saved in the Matlab workspace to the variables *data_theta*, *data_alpha*, and *data_Vm*. The time is stored in the *data_theta(:,1)* vector, the desired and measured rotary arm angles are saved in the *data_theta(;,2)* and *data_theta(;,3)* arrays, the pendulum angle is stored the *data_alpha(:,2)* vector, and the control input is in the *data_vm(:,2)* structure.
4. Measure the settling time and percentage overshoot of the simulated servo response, the maximum link deflection, and the voltage used. Does the response satisfy the specifications given in Section 3.1?

## 3.4.3 Control Implementation

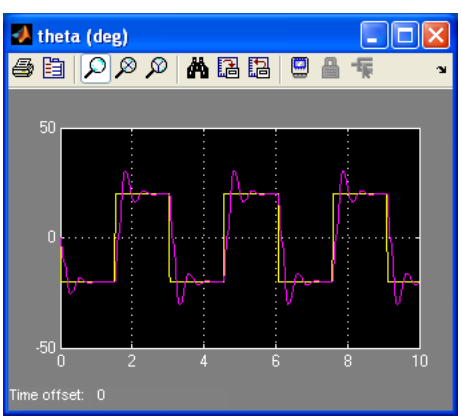
In this experiment, the servo position is controlled while minimizing the link deflection using the control found in Section 3.4.1. Measurements will then be taken to ensure that the specifications are satisfied.

### Experiment Setup

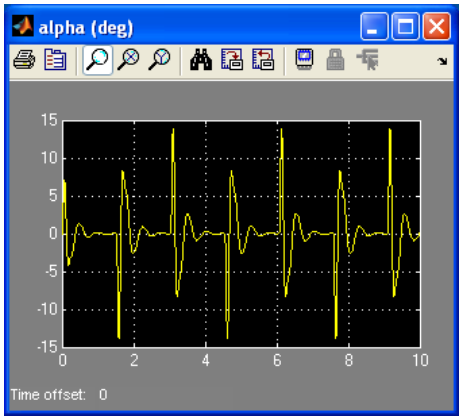
The *q_rotflex* Simulink diagram shown in Figure 3.5 is used to run the state-feedback control on the Quanser Flexible Joint system. The *SRV02 Flexible Joint* subsystem contains QUARC blocks that interface with the DC motor and sensors of the system. The feedback developed in Section 3.4.1 is implemented using a Simulink *Gain* block.

**IMPORTANT:** Before you can conduct this experiment, you need to make sure that the lab files are configured according to your system setup. If they have not been configured already, then go to Section 4.5 to configure the lab files first.

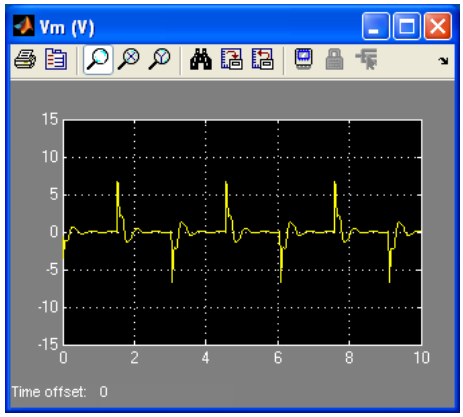
1. Run the *setup_rotflex.m*.




(a) Servo Angle




(b) Flexible Link Angle




(c) Voltage

Figure 3.4: Default Simulated Closed-Loop Response

2. Ensure the controller you found in Section 3.4.1 is loaded, i.e., gain $K$.
3. Open the q_rotflex Simulink diagram. Make sure the *Manual Switch* is set to the *Full-State Feedback* (upward) position.
4. Go to QUARC | Build to build the controller.
5. Go to QUARC | Start to run the controller. The flexible link should be tracking the default $\pm$ 20 degree signal.
6. Stop the controller once you have obtained a representative response.
7. Plot the responses from the *theta (deg)*, *alpha (deg)*, and *Vm (V)* scopes in a Matlab figure. Similarly as described in Section 3.4.2, the response data is saved in variables *data_theta*, *data_alpha*, and *data_vm*.
8. Measure the settling time and percentage overshoot of the measured servo response and the maximum link deflection. Does the response satisfy the specifications given in Section 3.1?

### 3.4.4 Implementing Partial-State Feedback Control

In this section, the partial-state feedback response of the system is assessed and compared with the full-state feedback control in Section 3.4.3.

1. Run the *setup_rotflex.m*.
2. Ensure the control gain you settled on in Section 3.4.3 is loaded, i.e., gain $K$.

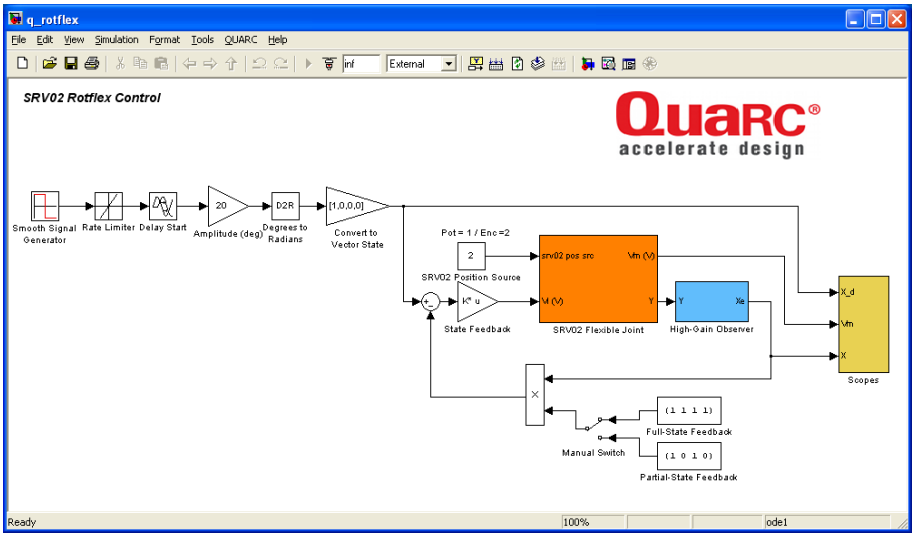



Figure 3.5: q_rotflex Simulink diagram used the model

3. Open the q_rotflex Simulink diagram. Make sure the *Manual Switch* is set to the *Partial-State Feedback* (downward) position.

4. Go to QUARC | Start to run the controller. The link should be tracking the default $\pm 20$ degree square wave.

5. Stop the controller once you have obtained a representative response.

6. As in Section 3.4.3, attach a Matlab figure representing the SRV02 angle and flexible link angle response, as well as the input voltage.

7. Measure the settling time and percentage overshoot of the measured servo response and the maximum link deflection. Does the response satisfy the specifications given in Section 3.1?

8. Examine the difference between the partial-state feedback (PSF) response and the full-state feedback (FSF) response. Explain why PSF control behaves this way by looking at the q_rotflex Simulink diagram.

## 3.5 Results

Fill out Table 2 with your answers from your control lab results - both simulation and implementation.

| Description | Symbol | Value | Units |
|---|---|---|---|
| **Pre Lab Questions** | | | |
| Desired poles | DP | | |
| Companion Gain | $\bar{K}$ | | |
| **Simulation: Control Design** | | | |
| Transformation Matrix | $W$ | | |
| Control Gain | $K$ | | |
| Closed-loop poles | CLP | | |
| **Simulation: Closed-Loop System** | | | |
| Maximum deflection | $\|\alpha\|_{max}$ | | deg |
| Maximum voltage | $\|V_m\|_{max}$ | | V |
| **Implementation** | | | |
| Control Gain | $K$ | | |
| Maximum deflection | $\|\alpha\|_{max}$ | | deg |
| Maximum voltage | $\|V_m\|_{max}$ | | V |

Table 2: Results

# 4 SYSTEM REQUIREMENTS

**Required Software**

- Microsoft Visual Studio
- Matlab®with Simulink®, Real-Time Workshop, and the Control System Toolbox.
- QUARC®2.1, or later.

See the QUARC®software compatibility chart at [6] to see what versions of MS VS and Matlab are compatible with your version of QUARC and for what OS.

**Required Hardware**

- Data-acquisition (DAQ) card that is compatible with QUARC. This includes Quanser Hardware-in-the-loop (HIL) boards such as:
  - Q2-USB
  - Q8-USB
  - QPID
  - QPIDe

  and some National Instruments DAQ devices (e.g., NI USB-6251, NI PCIe-6259). For a full listing of compliant DAQ cards, see Reference [2].
- Quanser SRV02-ET rotary servo.
- Quanser Rotary Flexible Joint (attached to SRV02).
- Quanser VoltPAQ-X1 power amplifier, or equivalent.

**Before Starting Lab**

Before you begin this laboratory make sure:

- QUARC®is installed on your PC, as described in [4].
- The *QUARC Analog Loopback Demo* has been ran successfully.
- SRV02 Rotary Flexible Joint and amplifier are connected to your DAQ board as described Reference [8].

## 4.1 Overview of Files

| File Name | Description |
|---|---|
| Flexible Joint User Manual.pdf | This manual describes the hardware of the Rotary Flexible Joint system and explains how to setup and wire the system for the experiments. |
| Flexible Joint Workbook (Student).pdf | This laboratory guide contains pre-lab questions and lab experiments demonstrating how to design and implement a position controller on the Quanser SRV02 Flexible Joint plant using QUARC® . |
| setup_rotflex.m | The main Matlab script that sets the SRV02 motor and sensor parameters, the SRV02 configuration-dependent model parameters, and the Flexible Joint (i.e., rotflex) 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. |
| config_rotflex.m | Returns the Flexible Joint model inertia, $J_l$, and viscous damping, $B_l$. |
| ROTFLEX_ABCD_eqns_student.m | Contains the incomplete state-space A, B, C, and D matrices. These are used to represent the Flexible Joint system. |
| calc_conversion_constants.m | Returns various conversions factors. |
| d_pole_placement_student.m | Use this script to find the feedback control gain K. |
| s_rotflex.mdl | Simulink file that simulates the Flexible Joint system when using a full or partial state-feedback control. |
| q_rotflex_id.mdl | When ran with QUARC®, this Simulink model feed a Sine Sweep signal to the servo and measures the corresponding Flexible Joint angle. The measured response can then be used to find the natural frequency of the link. |
| q_rotflex_val.mdl | This Simulink model is used with QUARC®to compare the Flexible Joint state-space model with the measured response from the actual system. |
| q_rotflex.mdl | Simulink file that implements a closed-loop state-feedback controller on the actual ROTFLEX system using QUARC® . |

Table 3: Files supplied with the SRV02 Flexible Joint Control Laboratory.

## 4.2 Setup for Finding Stiffness

Before beginning in-lab procedure outlined in Section 2.3.1, the q_rotflex_id Simulink diagram must be properly configured.

Follow these steps:

1. Setup the SRV02 with the Flexible Joint module as detailed in the Flexible Joint User Manual ([7]).
2. Load the Matlab software.
3. Browse through the *Current Directory* window in Matlab and find the folder that contains the file *q_rotflex_id.mdl*.
4. Open the *q_rotflex_id.mdl* Simulink diagram, shown in Figure 2.4.

5. **Configure DAQ:** Ensure the HIL Initialize block subsystem is configured for the DAQ device that is installed in your system. By default, the block is setup for the Quanser Q8 hardware-in-the-loop board. See Reference [2] for more information on configuring the HIL Initialize block.

## 4.3 Setup for Model Validation

Before performing the in-lab exercises in Section 2.3.2, the q_rotflex_val Simulink diagram and the setup_rotflex.m script must be configured.

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

1. Setup the SRV02 with the Flexible Joint module as detailed in [7].
2. Load the Matlab software.
3. Browse through the *Current Directory* window in Matlab and find the folder that contains the QUARC ROTFLEX file *q_rotflex_val.mdl*.
4. Open the *q_rotflex_val.mdl* Simulink diagram, shown in Figure 2.6.
5. **Configure DAQ:** Ensure the HIL Initialize block in the *SRV02 Flexible Joint* subsystem is configured for the DAQ device that is installed in your system. By default, the block is setup for the Quanser Q8 hardware-in-the-loop board. See Reference [2] for more information on configuring the HIL Initialize block.
6. **Configure Sensor:** The position of the SRV02 load shaft can be measured using either the potentiometer or the encoder. Set the *Pos Src* Source block in *q_rotflex_val*, as shown in Figure 2.6, 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 Input:** Set the *Manual Switch* to the DOWN position for a step input or the UP position for a square signal.
8. Open the setup_rotflex.m file. This is the setup script used for the ROTFLEX Simulink models.
9. **Configure setup script**: When used with the Flexible Joint, the SRV02 has no load (i.e., no disc or bar) and has to be in the high-gear configuration. Make sure the script is setup to match this setup:
   - EXT_GEAR_CONFIG to 'HIGH'
   - LOAD_TYPE to 'NONE'
   - Ensure 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.
   - CONTROL_TYPE to 'MANUAL'.

## 4.4 Setup for Flexible Joint Control Simulation

Before going through the control simulation in Section 3.4.2, the s_rotflex Simulink diagram and the setup_rotflex.m script must be configured.

Follow these steps to configure the lab properly:

1. Load the Matlab software.
2. **IMPORTANT:** Make sure the model you found in Section 2.3.2 is entered in ROTFLEX_ABCD _eqns _student.m.

3. Browse through the *Current Directory* window in Matlab and find the folder that contains the s_rotflex.mdl file.
4. Open s_rotflex.mdl Simulink diagram shown in Figure 3.3.
5. Configure the setup_rotflex.m script according to your hardware. See Section 4.3 for more information.
6. Run the setup_rotflex.m script.
7. Enter the stiffness (*Ks*) you found in Section 2.3.1.

## 4.5 Setup for Flexible Joint Control Implementation

Before beginning the in-lab exercises given in Section 3.4.3 (or Section 3.4.4), the q_rotflex Simulink diagram and the setup_rotflex.m script must be setup.

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

1. Setup the SRV02 with the Flexible Joint module as detailed in [7] .
2. Load the Matlab software.
3. Browse through the *Current Directory* window in Matlab and find the folder that contains the q_rotflex.mdl file.
4. Open the *q_rotflex.mdl* Simulink diagram shown in Figure 3.5.
5. **Configure DAQ:** Ensure the HIL Initialize block in the *SRV02 Flexible Joint* subsystem is configured for the DAQ device that is installed in your system. By default, the block is setup for the Quanser Q8 hardware-in-the-loop board. See Reference [2] for more information on configuring the HIL Initialize block.
6. **Configure Sensor:** The position of the SRV02 load shaft can be measured using the potentiometer or the encoder. Set the *Pos Src* Source block in *q_rotflex*, as shown in Figure 3.5, 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_rotflex.m* script according to your system setup. See Section 4.3 for more details.
8. Run the setup_rotflex.m script.

# 5 LAB REPORT

This laboratory contains two groups of experiments, namely,

1. Modeling the Quanser Rotary Flexible Joint system, and
2. State-feedback control.

For each experiment, follow the outline corresponding to that experiment to build the *content* of your report. Also, in Section 5.3 you can find some basic tips for the *format* of your report.

## 5.1 Template for Content (Modeling)

**I. PROCEDURE**

1. *Finding Stiffness*
   - Briefly describe the main goal of the experiment.
   - Briefly describe the experiment procedure in Step 3 in Section 2.3.1.
2. *Model Validation*
   - Briefly describe the main goal of the experiment.
   - Briefly describe loading the model in Step 6 in Section 2.3.2.
   - Briefly describe the model validation experiment in Step 13 in Section 2.3.2.

**II. RESULTS**

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

1. Frequency-response plot from step 3 in Section 2.3.1.
2. Model validation plot from step 13 in Section 2.3.2.
3. Provide applicable data collected in this laboratory (from Table 1).

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

1. Measured link stiffness in step 6 in Section 2.3.1.
2. Model discrepancies given in step 15 in Section 2.3.2.

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

1. How does the model compare with the actual system in step 14 of Section 2.3.2, *State-space model validation*.

## 5.2 Template for Content (Control)

**I. PROCEDURE**

1. *Control Design*
   - Briefly describe the main goal of the control design.
   - Briefly describe the control design procedure in Step 3 in Section 3.4.1.
2. *Simulation*
   - Briefly describe the main goal of the simulation.
   - Briefly describe the simulation procedure in Step 3 in Section 3.4.2.
3. *Full-State Feedback Implementation*
   - Briefly describe the main goal of this experiment.
   - Briefly describe the experimental procedure in Step 7 in Section 3.4.3.
4. *Partial-State Feedback Implementation*
   - Briefly describe the main goal of this experiment.
   - Briefly describe the experimental procedure in Step 6 in Section 3.4.4.

**II. RESULTS**

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

1. Matrices from Step 3 in Section 3.4.1, *Find transformation matrix*.
2. Response plot from step 3 in Section 3.4.2, *Full-state feedback controller simulation*.
3. Response plot from step 7 in Section 3.4.3, for *Full-state feedback controller implementation*.
4. Response plot from step 6 in Section 3.4.4, for *Partial-state feedback controller implementation*.
5. Provide applicable data collected in this laboratory (from Table 2).

**III. ANALYSIS**

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

1. Controllability of system in Step 2 in Section 3.4.1.
2. Closed-loop poles in Step 5 in Section 3.4.1.
3. Matlab commands used to generate the control gain in Step 6 in Section 3.4.1.
4. Settling time, percent overshoot, link deflection, and input voltage Step 4 in Section 3.4.2, *Full-state feedback controller simulation*.
5. Settling time, percent overshoot, link deflection, and input voltage in Step 8 in Section 3.4.3, *Full-state feedback controller implementation*.
6. Settling time, percent overshoot, link deflection, and input voltage in Step 7 in Section 3.4.3, *Partial-state feedback controller implementation*.
7. Comparison between partial-state and full-state feedback in step 8 in Section 3.4.4.

**IV. CONCLUSIONS**

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

1. Whether the closed-loop poles are in the required location in Step 5 in Section 3.4.1.
2. Whether the controller meets the specifications in Step 4 in Section 3.4.2, *Full-state feedback controller simulation*.
3. Whether the controller meets the specifications in Step 8 in Section 3.4.3, for *Full-state feedback controller implementation*.
4. Whether the controller meets the specifications in Step 7 in Section 3.4.4, for *Partial-state feedback controller implementation*.

## 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.

# REFERENCES


[1] Bruce Francis. Ece1619 linear systems course notes (university of toronto), 2001.

[2] Quanser Inc. *QUARC User Manual*.

[3] Quanser Inc. *SRV02 QUARC Integration*, 2008.

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

[5] Quanser Inc. *SRV02 User Manual*, 2009.

[6] Quanser Inc. *QUARC Compatibility Table*, 2010.

[7] Quanser Inc. *SRV02 Rotary Flexible Joint User Manual*, 2011.

[8] Quanser Inc. *SRV02 Rotary Flexible Link User Manual*, 2011.

[9] B. P. Lathi. *Modern Digital and Analog Communication Systems*. Oxford University Press, Inc., 3rd edition, 1998.

[10] Norman S. Nise. *Control Systems Engineering*. John Wiley & Sons, Inc., 2008.

## Ten modules to teach controls from the basic to advanced level

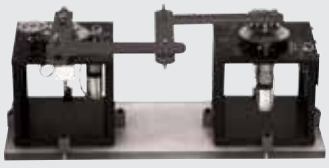
**SRV02 Base Unit**

**Flexible Link**

**Inverted Pendulum**

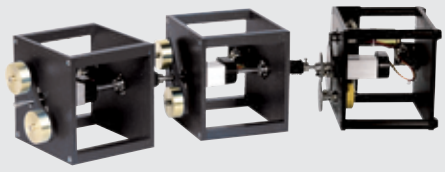
**Ball and Beam**

**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.