QUANSER
INNOVATE. EDUCATE.

LABORATORY GUIDE

Rotary Double Inverted Pendulum Experiment for MATLAB®/Simulink® Users

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

Quanser educational solutions are powered by:
MATLAB
SIMULINK

CAPTIVATE. MOTIVATE. GRADUATE.

© 2012 Quanser Inc., All rights reserved.

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

This document and the software described in it are provided subject to a license agreement. Neither the software nor this document may be used or copied except as specified under the terms of that license agreement. All rights are reserved and no part may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Quanser Inc.

CONTENTS

1 INTRODUCTION

This laboratory manual describes how to design a state-feedback control system that balances a rotary double inverted pendulum and positions the rotary arm to a commanded angular position.

The plant has two main components: the Quanser SRV02 rotary motion plant and the Quanser Rotary Double Inverted Pendulum, i.e., DBPEN-ROT module. The double pendulum is composed of a rotary arm that attaches to the SRV02 system, a short 7-inch bottom blue rod, an encoder hinge, and the top 12-inch blue rod.

Topics Covered

- Obtain a state-space representation of the open-loop system.
- Design a the state-feedback gain for the closed-loop system using Linear-Quadratic Regulator (LQR) optimization.
- Simulate the system and ensure it is stabilized using the designed state-feedback control.
- Implement the state-feedback controller on the DBPEN-ROT system and evaluate its actual performance.

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

1. Hardware and software requirements given in Section 4.
2. Modeling and state-space representation.
3. State-feedback design using Linear-Quadratic Regular (LQR) optimization.
4. Basics of Simulink®.
5. QUARC Integration lab detailed in Appendix A in the SRV02 Workbook [5].

2 BACKGROUND

2.1 Modeling

2.1.1 Model Convention

The rotary pendulum model is shown in Figure 2.1. The rotary arm pivot is attached to the SRV02 system and is actuated. The arm has a length of L_r, a moment of inertia of J_r, and its angle, θ, increases positively when it rotates counter-clockwise (CCW). The servo (and thus the arm) should turn in the CCW direction when the control voltage is positive, i.e., $V_m > 0$.

The double-pendulum assembly is connected to the end of the rotary arm. The short-sized, bottom pendulum has a total length of L_{p1} and a center of mass of l_{p1}. The moment of inertia about its center of mass is J_{p1} and it has a mass of M_{p1}. The top medium-sized pendulum has a total length of L_{p2}, a center of mass of l_{p2}, a moment of inertia of J_{p2}, and a mass M_{p2}. The short bottom pendulum angle, α, and the medium top pendulum angle, ϕ, are both zero when it is perfectly upright in the vertical position and they increase positively when rotated CCW. The hinge between the two pendulum has a mass of M_h.

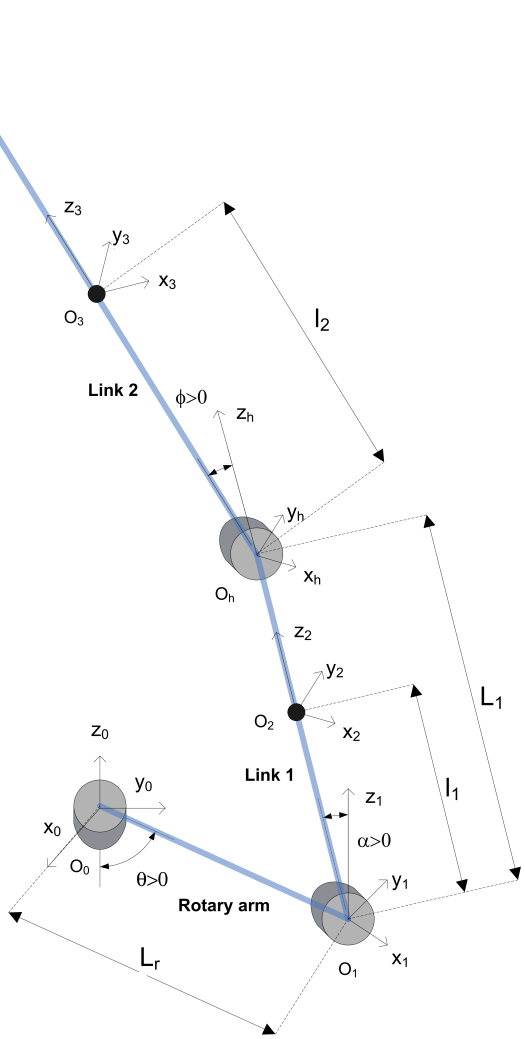


Figure 2.1: Rotary Double-Inverted Pendulum conventions

2.1.2 Nonlinear 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 rotary arm and the pendulum with respect to the servo motor voltage, i.e. the dynamics, will be obtained using the Euler-Lagrange equation:

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

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

$$q(t)^\top = \begin{bmatrix} \theta(t) & \alpha(t) & \phi(t) \end{bmatrix}$$

where, as shown in Figure 2.1, $\theta(t)$ is the rotary arm angle and $\alpha(t)$ is the inverted pendulum angle. The corresponding velocities are

$$\dot{q}(t)^\top == \begin{bmatrix} \frac{\partial \theta(t)}{\partial t} & \frac{\partial \alpha(t)}{\partial t} & \frac{\partial \phi(t)}{\partial t} \end{bmatrix}$$

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

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

$$\begin{aligned} \frac{\partial^2 L}{\partial t \partial \dot{\theta}} - \frac{\partial L}{\partial \theta} &= Q_1 \\ \frac{\partial^2 L}{\partial t \partial \dot{\alpha}} - \frac{\partial L}{\partial \alpha} &= Q_2 \\ \frac{\partial^2 L}{\partial t \partial \dot{\phi}} - \frac{\partial L}{\partial \phi} &= Q_3 \end{aligned}$$

The Lagrangian of the system is described

$$L = T - V$$

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

$$Q_1 = \tau - D_r \dot{\theta},$$

and acting on the bottom and top pendulum are

$$Q_2 = -D_{p1} \dot{\alpha}$$

and

$$Q_3 = -D_{p2} \dot{\phi}.$$

See [3] for a description of the corresponding SRV02 parameters (e.g. such as the back-emf constant, k_m). Our control variable is the input servo motor voltage, V_m. Opposing the applied torque is the viscous friction torque, or viscous damping, corresponding to the term D_r. Since the pendulum is not actuated, the only force acting on the link is the damping. The viscous damping coefficient of the short (bottom) and medium (top) pendulums are denoted by D_{p1} and D_{p2}.

The Euler-Lagrange equations is a systematic method of finding the equations of motion, i.e., 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. After going through this process, the nonlinear equations of motion for the system can be obtained. **See the supplied Maple worksheet (or its equivalent HTML representation) for the complete derivation.**

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

$$\tau = \frac{\eta_g K_g \eta_m k_t (V_m - K_g k_m \dot{\theta})}{R_m}. \tag{2.1}$$

See [3] for a description of the corresponding SRV02 parameters (e.g. such as the back-emf constant, k_m).

Both the equations match the typical form of an EOM for a single body:

$$J\ddot{x} + b\dot{x} + g(x) = \tau_1$$

where x is an angular position, J is the moment of inertia, b is the damping, $g(x)$ is the gravitational function, and τ_1 is the applied torque (scalar value).

For a generalized coordinate vector q, this can be generalized into the matrix form

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \tag{2.2}$$

where D is the inertial matrix, C is the damping matrix, $g(q)$ is the gravitational vector, and τ is the applied torque vector.

2.1.3 Linearizing

Here is an example of how to linearize a two-variable nonlinear function called $f(z)$. Variable z is defined

$$z^\top = [z_1 \; z_2]$$

and $f(z)$ is to be linearized about the operating point

$$z_0{}^\top = [a \; b]$$

The linearized function is

$$f_{lin} = f(z_0) + \left(\frac{\partial f(z)}{\partial z_1}\right)\Big|_{z=z_0} (z_1 - a) + \left(\frac{\partial f(z)}{\partial z_2}\right)\Big|_{z=z_0} (z_2 - b)$$

2.1.4 Linear State-Space Model

The linear state-space equations are

$$\dot{x} = Ax + Bu \tag{2.3}$$

and

$$y = Cx + Du \tag{2.4}$$

where x is the state, u is the control input, A, B, C, and D are state-space matrices. For the rotary pendulum system, the state and output are defined

$$x^\top = \begin{bmatrix} \theta & \alpha & \phi & \dot{\theta} & \dot{\alpha} & \dot{\phi} \end{bmatrix}$$

and

$$y^\top = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}.$$

After linearizing the nonlinear equations of motion, solving for the acceleration terms, i.e., $\ddot{\theta}$, $\ddot{\alpha}$, and $\ddot{\phi}$, and substituting the state given in , we obtain the following state-space matrices:

$$A = \frac{1}{J_T}\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & a_{42} & -M_{p1}l_{p1}L_rM_{p2}g(-l_{p1}+L_{p1}) & 0 & 0 & 0 \\ 0 & a_{52} & -M_{p2}g(-L_r^2L_{p1}M_{p1}+L_{p1}L_r^2M_{p1}+J_rL_{p1}) & 0 & 0 & 0 \\ 0 & a_{62} & a_{63} & 0 & 0 & 0 \end{bmatrix}$$

where

$$\begin{aligned}
a_{42} &= L_r g(M_{p1}^2 l_{p1}^2 + 2M_{p1}l_{p1}M_hL_{p1}M_hL_{p1}^2M_{p2}M_{p1}l_{p1}^2M_{p2}M_h^2L_{p1}^2) \\
a_{52} &= g(L_r^2L_{p1}M_{p1}M_{p2} + L_{p1}L_r^2M_hM_{p2} + L_{p1}L_r^2M_h^2 + L_r^2L_{p1}M_{p1}^2 + J_rL_{p1}M_h + J_rL_{p1}M_{p1} \\
&\quad + L_{p1}L_r^2M_hM_{p1} + L_r^2L_{p1}M_hM_{p1}) \\
a_{62} &= -\frac{g}{L_{p2}}\Big(L_{p1}L_r^2L_{p1}M_{p1}^2 + L_{p1}L_r^2L_{p2}M_hM_{p1} + L_r^2L_{p1}L_{p2}M_hM_{p1} + L_{p1}L_r^2L_{p2}M_h^2 \\
&\quad + L_r^2L_{p1}L_{p2}M_{p1}^2 - M_{p1}L_{p1}^2L_r^2M_{p2} + L_r^2L_{p1}L_{p2}M_{p1}M_{p2} + L_{p1}L_r^2L_{p2}M_hM_{p2} + J_rL_{p1}L_{p2}M_h + J_rL_{p1}L_{p2}M_{p1} \\
&\quad - M_{p1}L_{p1}^2J_r + L_{p1}J_rL_{p1}M_{p1} - M_{p1}L_{p1}^2M_hL_r^2 + L_r^2L_{p1}M_{p2}M_{p1}L_{p1} + M_{p1}L_{p1}L_r^2M_hL_{p1} - L_r^2M_{p1}^2L_{p1}^2\Big) \\
a_{63} &= \frac{g}{L_{p2}}\Big(J_rL_{p1}L_{p2}M_{p2} + J_rL_{p1}^2M_{p2} + L_{p1}^2L_r^2M_{p1}M_{p2} + M_{p1}L_{p1}^2L_r^2M_{p2} + L_{p1}L_r^2L_{p2}M_{p1}M_{p2} \\
&\quad - L_r^2L_{p1}L_{p2}M_{p1}M_{p2} + M_{p1}L_{p1}^2J_r + M_hL_{p1}^2J_r + M_hL_{p1}^2M_{p1}L_r^2 + M_{p1}L_{p1}^2M_hL_r^2 \\
&\quad - 2L_r^2L_{p1}M_{p2}M_{p1}L_{p1} - 2M_{p1}L_{p1}L_r^2M_hL_{p1}\Big)
\end{aligned}$$

and

$$B = \frac{1}{J_T}\begin{bmatrix} 0 \\ 0 \\ 0 \\ M_hL_{p1}^2 + M_{p1}L_{p1}^2 \\ L_r(M_hL_{p1} + M_{p1}L_{p1}) \\ -\frac{L_r}{L_{p2}}(-M_{p1}L_{p1}^2 + M_{p1}L_{p1}L_{p2} + M_hL_{p1}L_{p2} + M_{p1}L_{p1}L_{p1}) \end{bmatrix}.$$

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

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.5}$$

and

$$D = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{2.6}$$

Note: In the above model, we ignore the viscous damping paramters D_r, D_{p1}, and D_{p2}. For the full state-space system, see the Maple worksheet.

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

2.2 Control

In Section 2.1, we found a linear state-state space model that represents the Rotary Double Inverted Pendulum system. This model is used to investigate the stability properties of the system in Section 2.2.1. In Section 2.2.2,

the notion of controllability is introduced. Using the Linear Quadratic Regular algorithm, or LQR, is a common way to find the control gain and is discussed in Section 2.2.3. Lastly, Section 2.2.4 describes the state-feedback control used to control the servo position while minimizing link deflection.

2.2.1 Stability

The stability of a system can be determined from its poles ([7]):

- 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

$$\det(sI - A) = 0 \tag{2.7}$$

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.

2.2.2 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 ([7]).

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

$$T = \begin{bmatrix} B & AB & A^2B \dots A^nB \end{bmatrix} \tag{2.8}$$

equals the number of states in the system,

$$\text{rank}(T) = n. \tag{2.9}$$

2.2.3 Linear Quadratic Regular (LQR)

If (A,B) are controllable, then the Linear Quadratic Regular optimization method can be used to find a feedback control gain. Given the plant model in Equation 2.3, find a control input u that minimizes the cost function

$$J = \int_0^\infty x(t)'Qx(t) + u(t)'Ru(t)\,dt, \tag{2.10}$$

where Q and R are the weighting matrices. The weighting matrices affect how LQR minimizes the function and are, essentially, tuning variables.

Given the control law $u = -Kx$, the state-space in Equation 2.3 becomes

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

2.2.4 Feedback Control

The feedback control loop that in Figure 2.2 is designed to stabilize the servo to a desired position, θ_d, while minimizing the deflection of the pendulum.

The reference state is defined

$$x_d = \begin{bmatrix} \theta_d & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the controller is

$$u = K(x_d - x). \tag{2.11}$$

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

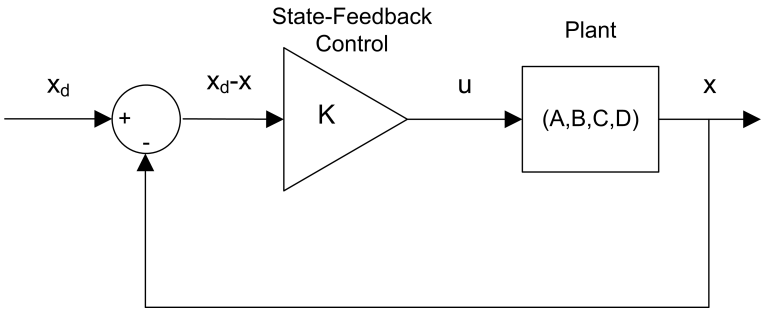


Figure 2.2: State-feedback control loop

To eliminate servo tracking error, we can augment the system to include an integrator such that

$$\dot{\eta} = \begin{bmatrix} A & 0 \\ 1 & 0 \end{bmatrix} \eta + \begin{bmatrix} B \\ 0 \end{bmatrix} u$$

where A and B are the state-space matrices defined in Section 2.1.4 and the states are

$$\eta^\top = \begin{bmatrix} \theta & \alpha & \phi & \dot{\theta} & \dot{\alpha} & \dot{\phi} & \int \theta \, \mathrm{d}t \end{bmatrix}$$

and

$$\eta_d{}^\top = \begin{bmatrix} \theta_d & 0 & 0 & 0 & 0 & 0 & \int \theta_d \, \mathrm{d}t \end{bmatrix}.$$

This introduces the integration terms $\eta_7(t) = \int \theta \, \mathrm{d}t$ to the feedback controller

$$u = K(\eta_d - \eta).$$

3 LAB EXPERIMENTS

3.1 Simulation

In this section we will use the Simulink diagram shown in Figure 3.1 to simulate the closed-loop control of the Rotary Double Inverted Pendulum system. The system is simulated using the linear model summarized in Section 2.1. The Simulink model uses the state-feedback control described in Section 2.2.4. The feedback gain K is found using the Matlab LQR command (LQR is described briefly in Section 2.2.3). The goal is to make sure the gain used successfully stabilizes the system (i.e., keeps it inverted), tracks the reference servo position, and does not saturate the dc motor.

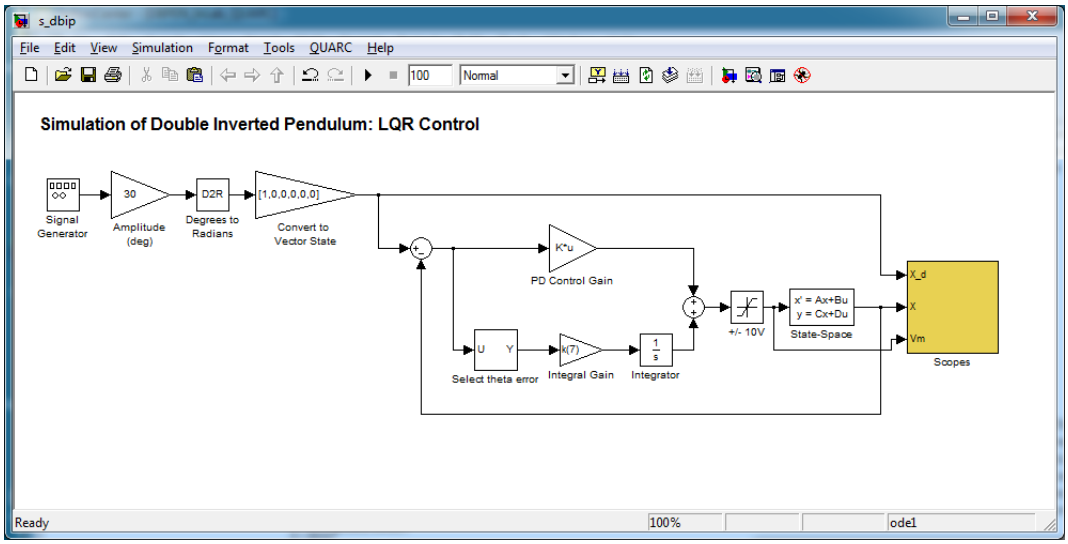


Figure 3.1: Simulink model used to simulate Rotary Double Inverted Pendulum.

The state-feedback controller has proportional-derivative (PD) action and the integral (I) action. The *PD Control Gain* block shown in Figure 3.1 multiplies the position and velocity states of the model by the first six elements of the vector gain k computed in earlier in, i.e., $k(1:6) = [k1, k2, k3, k4, k5, k6]$. The integral action is implemented in the *Integral Control* subsystem.

IMPORTANT: Before you can conduct these experiments, you need to make sure that the lab files are configured according to your setup. If they have not been configured already, then you need to go to Section 4 to configure the lab files first.

3.1.1 Procedure

Follow these steps to simulate the system:

1. Make sure the LQR weighting matrices in setup_dbip.m are to

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 \end{bmatrix}$$

and

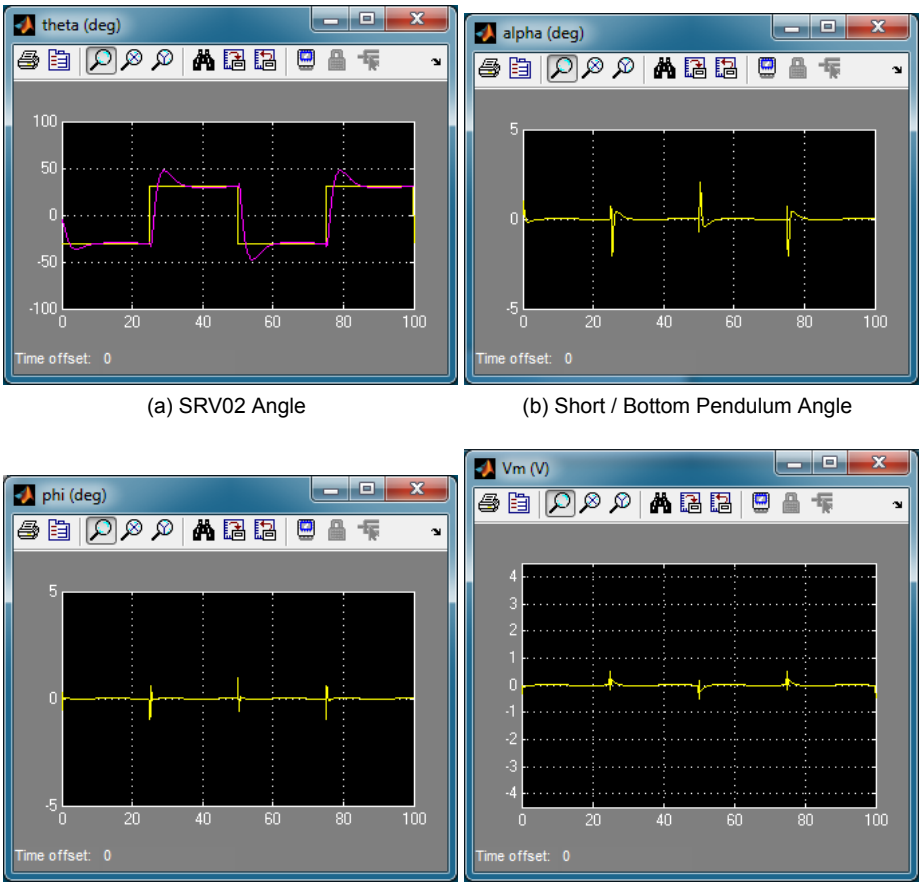
$$R = 30.$$

2. Run the script to generate the gain

$$K = \begin{bmatrix} 0.4777 & -14.4195 & -43.1956 & 0.5365 & -5.5311 & -4.3321 & 0.1291 \end{bmatrix}.$$

LQR Tuning: When tuning the LQR, we start with the identity matrix. To put more emphasis on the top / medium pendulum angle ϕ, we set $Q(4,4) = 5$. The last diagonal element, $Q(7,7)$ is set to 0.5 to generate a small integral gain for the servo tracking.

3. To generate a 0.02 Hz square wave reference, ensure the *Signal Generator* is set to the following:
   - Signal type = *square*
   - Amplitude = 1
   - Frequency = 0.02 Hz
4. Set the *Amplitude (deg)* gain blocks to 30 to generate a step with an amplitude of 30 degrees.
5. Open the servo gear position scope, *theta_l (rad)*, the bottom pendulum angle scope, *alpha (deg)*, the top pendulum angle scope, *phi (deg)*, and the motor input voltage scope, *Vm (V)*.
6. Start the simulation. By default, the simulation runs for 100 seconds. The scopes should be displaying responses similar to Figure 3.2. Note that in the *theta_l* (rad) scopes, the yellow trace is the setpoint position while the purple trace is the simulated position.



(a) SRV02 Angle

(b) Short / Bottom Pendulum Angle

(c) Medium / Top Pendulum Angle

(d) Voltage

Figure 3.2: Simulated closed-loop response.

3.1.2 Analysis

The feedback response is shown in Figure 3.3. You can generate this figure by running the plot_response_dbip.m script.

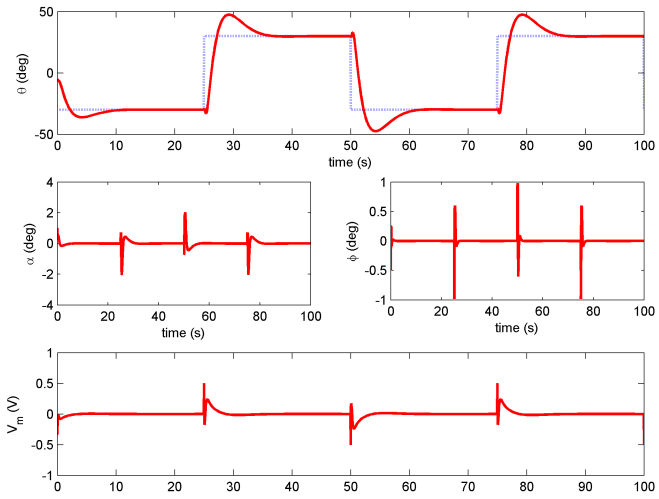


Figure 3.3: Simulated Rotary Double Inverted Pendulum feedback response.

As shown by the response in Figure 3.3, the pendulum maintains its balance about the upright vertical position while tracking the $\pm$ 30 degree servo angle.

Generating the Matlab figure: After each simulation run, each scope automatically saves their response to a variable in the Matlab® workspace. The *theta (deg)* scope saves its response to the variable called *data_theta*, the *alpha (deg)* scope saves its data to the *data_alpha* variable, the *phi (deg)* scope saves its data to the *data_phi* variable, and the *Vm (V)* scope saves its plot to the *data_vm* variable. See the *plot_response_dbip.m* script to see how they are used for plotting.

3.2 Implementation

The q_2bip Simulink diagram shown in Figure 3.4 is used to perform the balance control on the DBPEN-ROT. The *DBPEN-ROT* subsystem contains QUARC® blocks that interface with the DC motor and sensors of the DBPEN-ROT system.

The interior of the *DBPEN-ROT* subsystem is shown in Figure 3.5. The *Enable Control* subsystem outputs 1 when the pendulum angles are within the value in the *Desired Initial Angle (deg)* source block. By default, the controller is enabled when they are within 1.5 degrees. In the *Integral Control* subsystem (in the main q_dbip diagram), the integrator is reset to 0 when the controller is enabled.

IMPORTANT: Before you can conduct these experiments, you need to make sure that the lab files are configured according to your setup. If they have not been configured already, then you need to go to Section 4 to configure the lab files first.

3.2.1 Procedure

Follow this procedure:

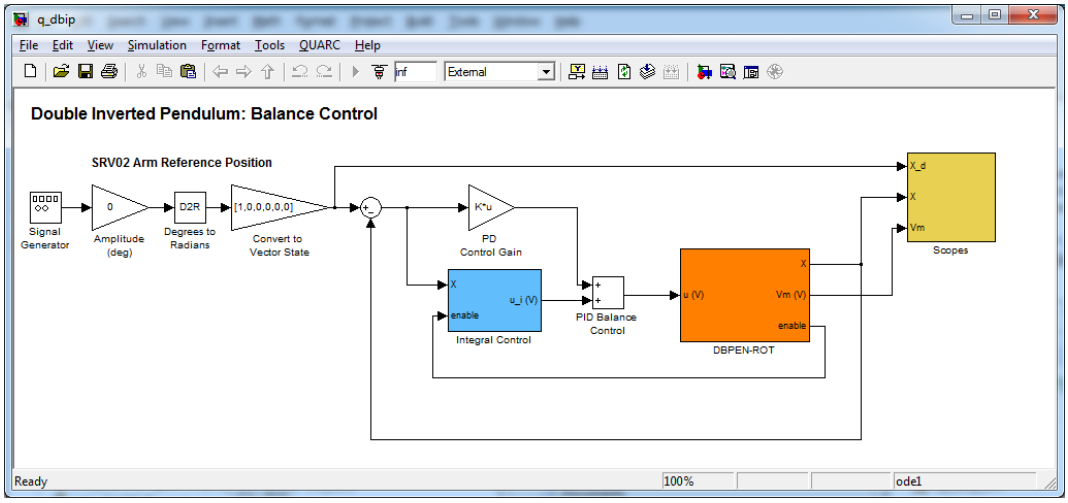


Figure 3.4: Simulink model used with QUARC® to run controller on the DBPEN-ROT.

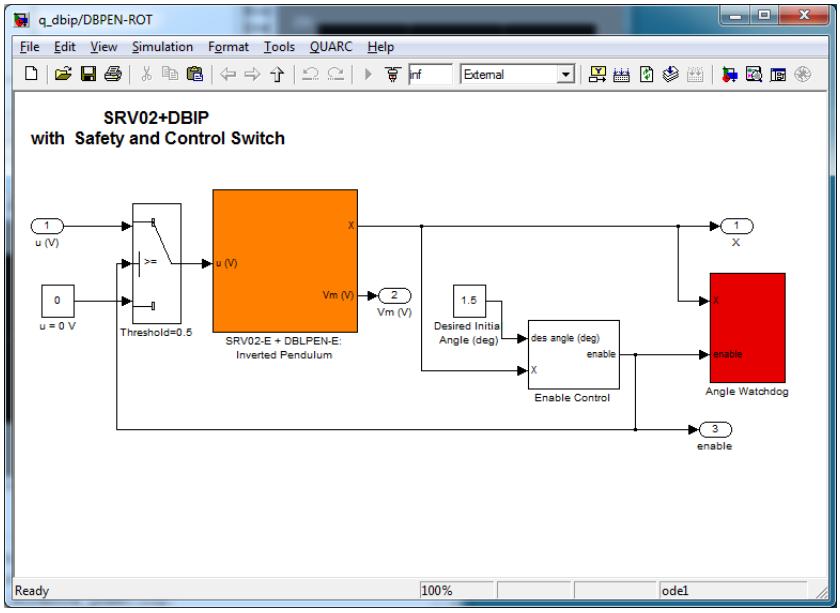
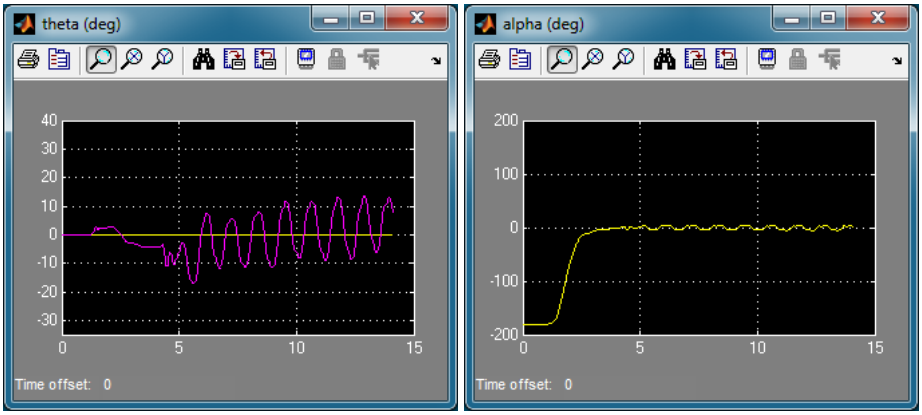


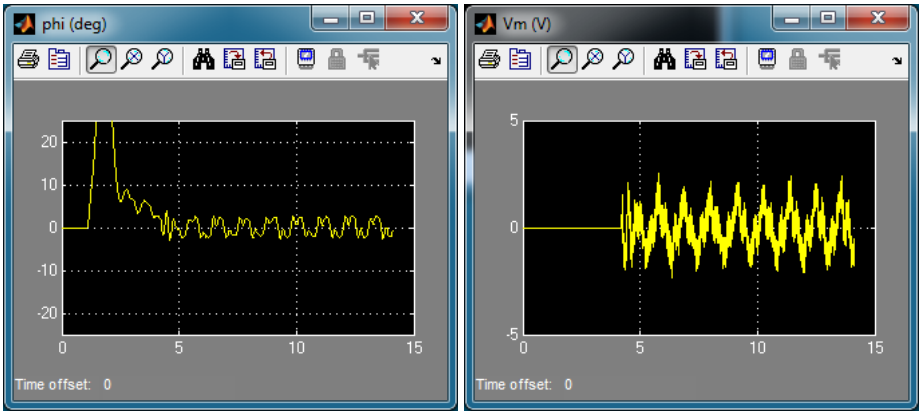
Figure 3.5: DBPEN-ROT subsystem in q_dbip Simulink diagram

1. Run the setup_dbip.m script using the LQR weighting matrices that you used in the simulation in Section 3.1.
2. Set the *Amplitude* gain block to 0 to turn off the servo setpoint.
3. Open the servo gear position scope, *theta_l (rad)*, the bottom pendulum angle scope, *alpha (deg)*, the top pendulum angle scope, *phi (deg)*, and the motor input voltage scope, *Vm (V)*.
4. In the Simulink diagram, go to QUARC | Build.
5. **Make sure the pendulum assembly is in the hanging down position and motionless.** The Simulink diagram will measures the short pendulum angle, α, as -180 degrees.
6. Click on QUARC | Start to run the controller. The scopes should all be reading 0 except for *alpha (deg)* - it should read -180 degrees.
7. Manually rotate the pendulum assembly to the upright vertical position. Do this slowly and keep the bottom and top pendulums straight and in-line with each other. Once the control engages, immediately release the pendlum and let the servo balance the links. The scopes should be displaying responses similar to Figure

3.6. Note that in the *theta_x (deg)* and *theta_y (deg)* scopes, the yellow trace is the setpoint position while the purple trace is the measured position.



(a) SRV02 Angle (b) Short / Bottom Pendulum Angle



(c) Medium / Top Pendulum Angle (d) Voltage

Figure 3.6: Typical response when running control on DBPEN-ROT system

8. To stop the experiment, click on the QUARC | Stop button but make sure you catch the pendulum before it swings down. This will prevent the assembly from hitting the table surface.

3.2.2 Analysis

The closed-loop position response is shown in Figure 3.7. You can generate this using the plot_response_dbip.m script after running the q_dbip QUARC controller.

Due to the friction in the system, the servo oscillates back-and-forth approximately ± 10 degrees to balance the double pendulum. Because of the integrator, the servo eventually stabilizes about the 0 degree setpoint. In the *alpha (deg)* scope, the pendulum angle goes up from -180 degrees to 0 at which point the balance control is engaged (around the 2.5 second mark).

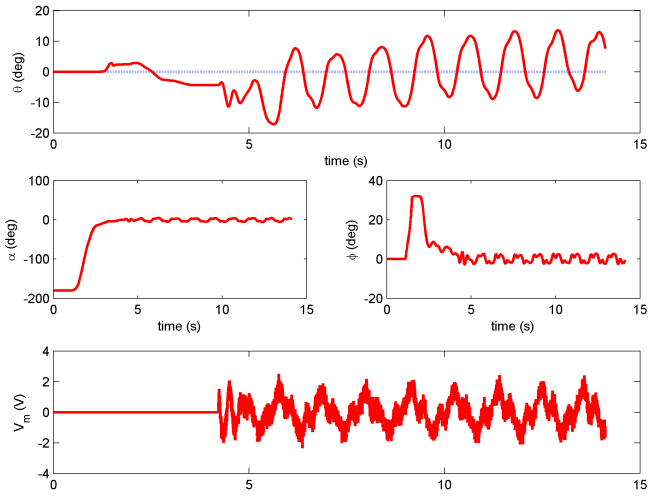


Figure 3.7: DBPEN-ROT balance control response

4 SYSTEM REQUIREMENTS

Required Software

- Microsoft Visual Studio (MS VS)
- Matlab® with Simulink®, Real-Time Workshop, and the Control System Toolbox
- QUARC®

See the QUARC® software compatibility chart in [4] 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) device **with 3x encoder inputs** and that is compatible with QUARC®. This includes Quanser DAQ boards such as Q8-USB, QPID, and QPIDe and some National Instruments DAQ devices. For a full listing of compliant DAQ cards, see Reference [1].
- Quanser SRV02-ET rotary servo.
- Quanser Rotary Double Inverted Pendulum (attached to the load gear of the 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 [2].
- DAQ device has been successfully tested (e.g., using the test software in the Quick Start Guide or the *QUARC Analog Loopback Demo*).
- Rotary Double Inverted Pendulum and amplifier are connected to your DAQ board as described its User Manual [6].

4.1 Overview of Files

File Name	Description
Rotary Double Inverted Pendulum User Manual.pdf	This manual describes the hardware of the DBPEN-ROT system and explains how to setup and wire the system for the experiments.
Rotary Double Inverted Pendulum Laboratory Guide.pdf	This document demonstrates how to obtain the linear state-space model of the system, simulate the closed-loop system, and implement controllers on the DBPEN-ROT plant using QUARC®.
setup_dbip.m	The main Matlab script that sets the SRV02 motor and sensor parameters, the SRV02 configuration-dependent model parameters, and the DBPEN-ROT 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_sp.m	Returns the pendulum model parameters.
s_dbip.mdl	Simulink file that simulates the closed-loop control of a Rotary Double Inverted Pendulum system using state-feedback control.
q_dbip.mdl	Simulink file that implements the state-feedback control on the Rotary Double Inverted Pendulum system using QUARC®.
dbpen-rot.mws	Maple worksheet used to develop the model for the DBPEN-ROT experiment. Waterloo Maple 9, or a later release, is required to open, modify, and execute this file.
dbpen-rot.html	HTML presentation of the Maple Worksheet. It allows users to view the content of the Maple file without having Maple 9 installed. No modifications to the equations can be performed when in this format.
plot_response_dbip.m	Plots the response found in the variables data_theta, data_alpha, data_phi, and data_vm in the Matlab workspace. These variables are saved from the Simulink diagrams.
meas_step_rsp_specs.m	Function that measures the peak time, settling time, steady-state error, and percent overshoot of a given step response.

Table 4.1: Files supplied with the DBPEN-ROT

4.2 Setup for Simulation

Before beginning the in-lab procedure outlined in Section 3.1, the s_dbip Simulink diagram and the setup_dbip.m script must be configured.

Follow these steps:

1. Load the Matlab software.
2. Browse through the *Current Directory* window in Matlab and find the folder that contains the file *setup_dbip.m*.
3. Open the *setup_dbip.m* script.

4. **Configure setup_dbip.m script**: When used with the DBPEN-ROT, 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'
    - K_AMP to 1 (unless your amplifier gain is different)
    - AMP_TYPE to your amplifier type (e.g., VoltPAQ).
    - Ensure other parameters such as ENCODER_TYPE, TACH_OPTION, and VMAX_DAC match your system configuration.
5. Run setup_dbip.m to setup the Matlab workspace.
6. Open the *s_dbip.mdl* Simulink diagram, shown in Figure 3.1.

4.3 Setup for Running on DBPEN-ROT

Before performing the in-lab exercises in Section 3.2, the q_dbip Simulink diagram and the setup_dbip.m script must be configured.

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

1. Setup the SRV02 with the DBPEN-ROT module as detailed in the DBPEN-ROT User Manual [6].
2. **Make sure the pendulum is in the hanging, downward position**. For more information, go to the DBPEN-ROT User Manual [6].
3. Configure and run setup_dbip.m as explained in Section 4.2.
4. Open the *q_dbip.mdl* Simulink diagram, shown in Figure 3.4.
5. **Configure DAQ:** Ensure the HIL Initialize block in the *DBPEN-ROT* subsystem is configured for the DAQ device that is installed in your system. See Reference [1] for more information on configuring the HIL Initialize block.

REFERENCES

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

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

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

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

[5] Quanser Inc. SRV02 lab manual. 2011.

[6] Quanser Inc. *Rotary Double Inverted Pendulum User Manual*, 2012.

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

Over ten rotary experiments for teaching fundamental and advanced controls concepts

Rotary Servo Base Unit

Double Inverted Pendulum

Inverted Pendulum

Flexible Link

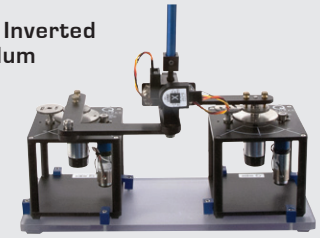
Flexible Joint

Ball and Beam

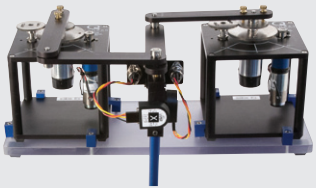
Gyro/Stable Platform

2 DOF Robot

2 DOF Inverted Pendulum



2 DOF Gantry



Multi-DOF Torsion

Quanser's rotary collection allows you to create experiments of varying complexity – from basic to advanced. Your lab starts with the Rotary Servo Base Unit and is designed to help engineering educators reach a new level of efficiency and effectiveness in teaching controls in virtually every engineering discipline including electrical, computer, mechanical, aerospace, civil, robotics and mechatronics. For more information please contact info@quanser.com

©2013 Quanser Inc. All rights reserved.

QUANSER

INNOVATE. EDUCATE.

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

Solutions for teaching and research. Made in Canada.