

Lab 2: Quanser Hardware and Proportional Control

I. Objective

The goal of this lab is:

- Help students become familiar with the MATLAB Real Time Workshop (RTW), Quanser's QuaRC tool, and the Q4 data acquisition board.
- Derive and understand a model for the dynamics of the cart (minus the pendulum).
- Use proportional control to generate a step response on the actual hardware.

II. Software and equipment

- Computer with MATLAB, Simulink, RTW, and QuaRC installed (204-04, 204-06, 204-12).
- ee128 student account

III. Theory

1. Real Time Workshop, QuaRC, and the Q4 DAQ board

Real Time Workshop is a MATLAB toolbox that enables the user to generate and execute stand-alone C code for developing and testing algorithms modeled in Simulink and Embedded MATLAB code [3].

QuaRC is Quanser's new, state-of-the-art rapid prototyping and production system for real-time control. QuaRC integrates seamlessly with Simulink to allow Simulink models to be run in real-time on Windows. It uses a **host** and **target** relationship that allows code generation and execution to occur on separate machines. However, we will be using what is known as "Single User Mode" or "Local Configuration," where we will be generating and executing code on the same computer, as shown in Figure 1 below.

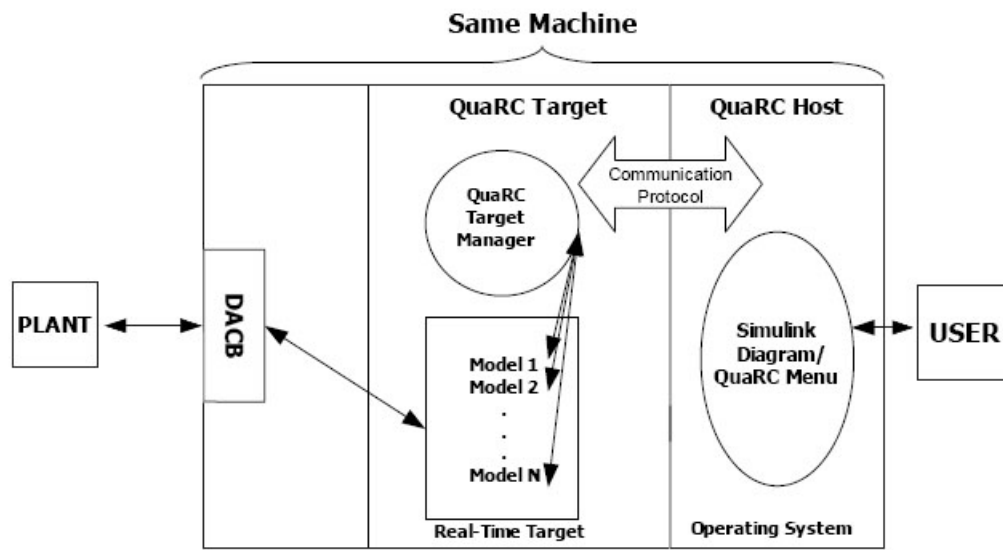


Figure 1. Local Configuration: QuaRC Host and Target on the same PC.

The **QuaRC Simulink Development Environment (SDE)** is used to generate/build code to be later run on a real-time target from MATLAB/Simulink models. The **QuaRC Windows Target** feature is required to run the generated code from MATLAB/Simulink models on a real-time Windows target (local or remote). QuaRC Windows Target needs to be open to run any QuaRC-generated code.

The functionality of QuaRC is transparent to the user, your task is to just design the controller based on either classic or state-space techniques. Then you implement the controller in Simulink via Real-Time Workshop. This is then “downloaded” to the QuaRC target.

But, how do you interface your controller with the plant? The answer is the **Q4 Data Acquisition (DAQ) board** [4]. This board supports 4 A/D converters, 4 D/A converters, 16 Digital I/Os, 2 Realtime clocks, and up to 4 Quadrature input decoders/counters [4].

The Q4 board’s functionality has also been abstracted from the user. The board has been set up to work with the cart and pendulum for all the station. **DO NOT CHANGE ANY OF THE HARDWARE AT THE QUANSER STATIONS WITHOUT FIRST CONSULTING THE TA!!!**

2. Dynamics of the Cart

Figure 2 below shows the cart's free body diagram. For simplicity, we will ignore the effects of friction.

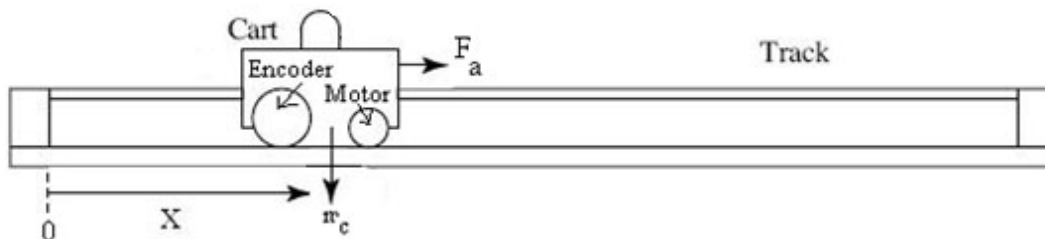


Figure 2. Pendulum system free body diagram.

Parameters in figure 2:

F_a is the input force exerted on the cart by the voltage applied to the motor.

m_c is the mass of the cart.

Encoder is used to keep track of the position of the cart on the track.

Based on figure 2 and basic Newtonian dynamics you can derive the equations governing the system.

3. Motor Dynamics [5]

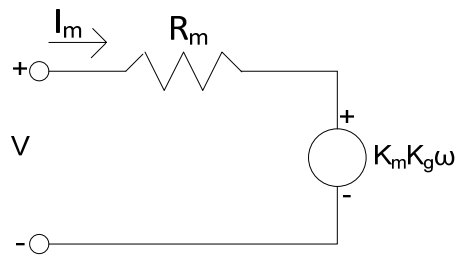


Figure 3. Circuit diagram of motor.

The input to your system is actually a voltage to the drive motor. Thus, you need to derive the dynamics of the system that converts the input voltage to force. This is precisely the dynamics of the drive motor.

The torque generated by the motor is proportional to the current flowing through the motor windings:

$$\tau = K_m K_g I_m \quad (1)$$

K_m is the back EMF constant $\left(\frac{V}{\text{rad/sec}}\right)$

K_g is the motorbox gear ratio

I_m is the current flowing through the coil

Now, the current flowing through the motor can be related to the motor voltage input by:

$$V = I_m R_m + K_m K_g \omega \quad (2)$$

R_m is resistance of the motor windings

ω is angular velocity of the motor

The angular velocity is related to the linear velocity and the torque is also related to the applied force.

$$\dot{x} = \omega \cdot r \quad (3)$$

$$\tau = F_a \cdot r \quad (4)$$

r is the radius of the motor gear

4. Step Response of a System [6]

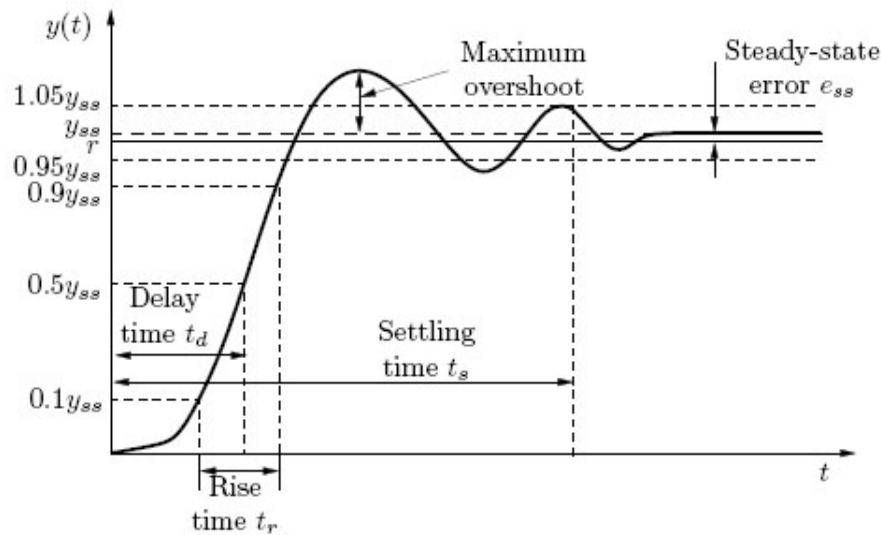


Figure 4. Typical step response of a control system.

Steady-state error: The *steady-state value* of the step response $y(t)$ is defined as

$$y_{ss} := \lim_{t \rightarrow \infty} y(t)$$

For a control system, we want the output, $y(t)$, to follow a desired *reference* signal, $r(t)$. Thus we can define the error as

$$e(t) := r(t) - y(t)$$

and consequently, the *steady-state error* is given by

$$e_{ss} := \lim_{t \rightarrow \infty} e(t)$$

Maximum overshoot: Let y_{max} denote the maximum value of $y(t)$. The *maximum overshoot* of the step response $y(t)$ is defined as

$$\text{maximum overshoot} := y_{max} - y_{ss}$$

The maximum overshoot is often represented as a percentage of the steady-state value, i.e.

$$\text{percent maximum overshoot} = \frac{y_{max} - y_{ss}}{y_{ss}} \times 100\%$$

The maximum overshoot is often used to measure the relative stability of a system. A system with a large overshoot is usually undesirable.

Delay time: The *delay time* t_d is defined as the time required for the step response to reach 50% of its steady-state value.

Rise time: The *rise time* t_r is defined as the time required for the step response to rise from 10% to 90% of its steady-state value.

Settling time: The *settling time* t_s is defined as the time required for the step response to stay within 5% of its steady-state value.

IV. Prelab

1. Equations governing the cart dynamics

Derive the following equation of motion for the cart system shown in figure 2.

$$m_c r^2 R_m \ddot{x} + K_m^2 K_g^2 \dot{x} = r K_m K_g V \quad (5)$$

In the equation above:

V is the input voltage (volts)

m_c is the mass of the car (kilograms)

r is the radius of the motor gear (meters)

R_m is resistance of the motor windings (ohms)

K_m is the back EMF constant $\left(\frac{V}{\text{rad/sec}}\right)$

K_g is the motorbox gear ratio (no units)

In order to derive the equation (5), use the steps below:

Step 1. Applying Newton's second law to cart gives:

$$\sum F = m_c \ddot{x} \quad (6)$$

Using the free body diagram of the cart from figure 2 as a guide, replace F with the horizontal forces acting on the cart (Hint: Don't over think it).

Step 2. Substitute the motor dynamics:

Combine all the motor dynamics equations, equations 1-4, to obtain the relationship between the input voltage V and the applied force F_a . Substitute this relationship into your equation from Step 1. This is the final model of your plant.

Step 3. Is this system linear?

If not, linearize the system. If so, leave as is.

2. Derive system models

Transfer Function. Apply the Laplace transform to your linear system the transfer function $X(s)/V(s)$.

State Space. Derive the state space matrices A, B, C, and D for your linear system.

SS to TF. Using the following equations, derive a transfer function from your state space matrices and verify that it matches the transfer function you got directly from taking the Laplace transform of the equation of motion.

$$G(s) = \frac{Y(s)}{U(s)} = C(sI - A)^{-1}B + D \quad (7)$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \quad (8)$$

3. MATLAB Step Response

Use the following values to create a Simulink block diagram of the cart system in a simple negative feedback loop with a gain K as the controller. It is your choice whether you want to use the state space or transfer function representation of the system.

$$m_c = 0.57 + 0.37 = 0.94 \text{ kg}$$

$$r = 0.00635 \text{ m}$$

$$R_m = 2.6 \text{ } \Omega$$

$$K_m = 0.00767 \text{ V*s/rad}$$

$$K_g = 3.71$$

Vary the value of K until you can achieve a percent maximum overshoot < 5% and $t_r < 0.4s$. Include your block diagram as well as your final value of K and plots verifying these design conditions are met.

You will find the MATLAB function **find(cond,N)** to be very useful for this. This returns at most the first N *indices* that match the condition cond. Type "help find" to read about the other various uses for this function if you wish. For example, for an array of output values **out** and time values **time**, you can use the following code to find the time of the first value of **out** that exceeds 0.1:

```
>> find(out>0.1,1)
ans =
    110
>> time(110)
ans =
    1.0900
```

To get more precise time and output values, it is suggested that you set Simulink to a small fixed-step interval. You are also welcome, but not required, to use linear interpolation between points.

V. Lab

1. Cart Dynamics

Confer with your group to come up with a single equation of motion and system representation (either state space or transfer function) and get them checked off by the TA.

2. The Quanser Hardware

The TA will give a demonstration of how to properly use the Quanser system. Make sure you understand the system functionality so you can implement your controller easily.

MAKE SURE TO OBSERVE SAFETY PRECAUTIONS AT ALL TIMES. Any group that leaves the amplifier on after they are finished using Quanser hardware will **automatically lose 5 points from their lab report**. This penalty will stack for every offense.

We have a limited number of working systems, so expect delays if you have to wait for a system to open up. Some parts of this lab do NOT require the hardware. Please work on those sections while waiting a station. Be respectful of the sharing situation. Any group that leaves one of the computers at the designated stations (204-04, 204-06, and 204-12) LOCKED, will **automatically lose 5 points from their lab report**. This penalty will stack for every offense.

3. Using the Actual Hardware – Find Encoder-Distance Conversion

The TA will cover how to interface with the actual cart hardware by building a Simulink subsystem. Encoder values for the position of the cart will be read in encoder counts. Our input will be in inches. In feedback, the two values you compare MUST be in the same units, so we need a conversion factor.

Build a simple Simulink file that does nothing but read the position encoder count. If you use a Scope, it will update in real time. Attached at the end of this lab is a paper ruler that you can print and cut out. Feel free to bring an actual ruler to lab if you so desire. Once you start running the QuaRC program, you can manually move the cart along the track and watch the encoder values update. Using a ruler, move the cart manually an inch and let it sit for a while. Repeat this a couple of times. You can set the final time in Simulink to “inf” for infinity to run the program indefinitely. You can use QuaRC => Stop to stop the program. Using the plateaus in your plot (from letting the cart sit at a position for a while), calculate the change in encoder count from moving the cart an inch. Average your values over all the inch-length moves for better accuracy. Include the plot of the cart encoder values in your lab report and document your calculation of the encoder counts/inch.

The Quanser manual gives the position encoder resolution to be 4096 counts/revolution. Given that the radius of the position pinion is $r_{pp} = 0.01482975$ m, what is the “actual” encoder resolution in count/inch? How close was your estimated encoder resolution?

4. Using the Actual Hardware – Cart Step Response

Go back to your Simulink model of the cart system from the prelab. Now change the step function to be of height 4, corresponding to the cart moving 4 inches (watching it move 1 inch isn't very exciting).

Again, try to find a value of K so that percent maximum overshoot $< 5\%$ and $t_r < 0.4s$. How different is the value you found here from the value of K you found in the prelab for a step size of 1?

Once the simulated step response looks fine, you can move over to the actual hardware. Replace your system block (ss or tf) in the feedback loop with the hardware subsystem, with the counts/inch conversion included. Run the QuaRC program. Plot the initial hardware response and compare with the plot from the Simulink model. How close was the actual to the predicted? What might have caused any discrepancies? Now change your value of K until you achieve a percent maximum overshoot of $< 5\%$. Report your new K value and plot the hardware response.

You will be asked to run your modified hardware step response when you turn in your lab report.

VI. Revision History

Semester and Revision	Author(s)	Comments
Fall 2008 Rev. 2.00	Justin Hsia	Completely revamped lab. Kept parts of theory section, but updated for the new hardware and software.
Summer 2008 Rev. 1.00	Bharathwaj Muthuswamy	1. Formatted writeup into different sections. 2. Typed up solutions

VII. References

1. Franklin, Gene F., Powell, David J. and Emami-Naeini Abbas. Feedback Control of Dynamic Systems. 5th Edition. 2006, Prentice-Hall Inc.
2. Quanser Consulting Inc. QuaRC 1.1 Installation Guide, 2008.
3. The Mathworks Inc. Real-Time Workshop 7 Datasheet. Available online: https://tagteambdserver.mathworks.com/ttserverroot/Download/43808_9400v06_RTW.pdf June 27th 2008.
4. Quanser Consulting Inc. Q4 Data Acquisition System User's Guide Version 1.0, 2003.
5. Quanser Consulting Inc. Self Erecting IP User's Manual, 1996.
6. Ergueta, Edgar. PID Control Gain Tuning of a Drive Position Servo System. ME107B, Fall 2006.