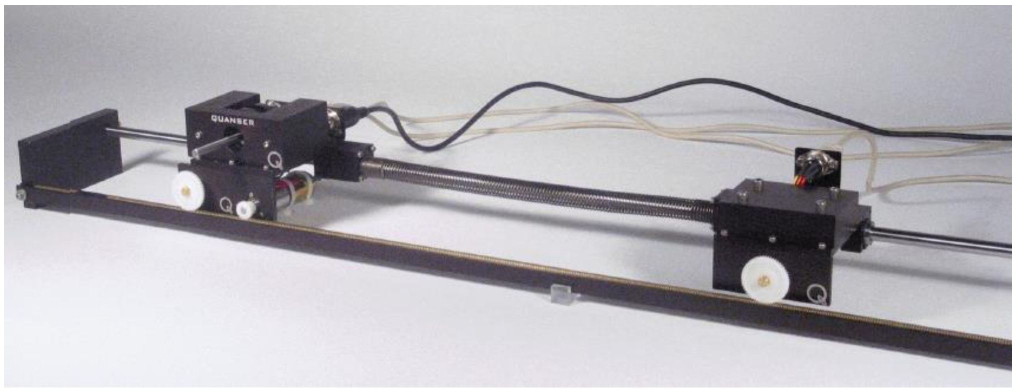
Middle East Technical University
Department of Electrical and Electronics Engineering

EE 406
Laboratory of Feedback Control Systems



Experiment #5:
Single Linear Flexible Joint Control using Linear Quadratic Regulator (LQR)

Preliminary Work and Laboratory Manual

Initial Version: Afşar Saranlı, Emre Tuna

Group Members:

____

____

1 Objectives

The Single Linear Flexible Joint (SLFJ) experiment consists of a system of two carts sliding on an IP02 track, as shown in Figure 1 below. While one of the two carts is motorized and drives the system (i.e., IP02), the second cart, an LFJC-E module, is passive and coupled to the first cart through a linear spring. The shafts of these elements are coupled to a rack and pinion mechanism in order to input the driving force to the system (IP02) and to measure the two cart positions (IP02 and LFJC-E). The challenge of the present laboratory is to design a control system to track the spring-driven cart (LFJC-E module) to a desired position as quickly as possible while minimizing the overshoot and residual vibration. Therefore, the desired outcome is to design a feedback controller such that the output cart (LFJC-E module) tracks a position setpoint while minimizing joint deflection and resonance in the system. This should only be achieved by controlling the input (i.e., motorized) cart position. There are many real-world applications of such a spring-mass problem. Some of which are found in the presence of elasticity in mechanical transmissions, like for example in gearboxes or long robotic arms such as the one mounted on the International Space Station (ISS). Systems with elastic linkages also include machine tools or other mechanical positioning servo-systems. Therefore, the design requirements for a system with elastic linkage usually bear on the final load (tip point) motion performance. Generally speaking, the load motion is assessed in terms of speed of response, minimum oscillation, and position accuracy. During the course of this experiment, you will become familiar with the design and implementation (e.g., tuning principles) of a full-state Linear Quadratic Regulator (LQR). Both time and frequency analyses will be carried out.

At the end of the session, you should know the following:

i) How to mathematically model the Linear Flexible Joint Cart (LFJC-E) coupled to an IP02 linear servo plant, using, for example, Lagrangian mechanics or force analysis on free body diagrams.
ii) How to obtain a state-space representation of the open-loop system.
iii) How to design, simulate, and tune an LQR-based state-feedback controller satisfying the closed-loop system's desired design specifications.
iv) How to use integral action to eliminate steady-state error.
v) How to implement your LQR in real-time and evaluate its actual performance.
vi) How to tune on-line and in real-time your LQR so that the actual linear-flexible-linkage-cart system meets the closed-loop design requirements.

2 Prerequisites

To successfully carry out this laboratory, the prerequisites are:

i) To be familiar with your IP02 main components (e.g., actuator, sensors), your data acquisition card (Q8 and Q2), and your power amplifier (UPM), as described in References [2], [4], and [5].
ii) To be familiar with your Linear Flexible Joint Cart (LFJC-E) module, as described in Reference [3].

iii) To have successfully completed the pre-laboratory described in Reference [1]. Students are therefore expected to be familiar in using QuaRC to control and monitor the plant in real-time and in designing their controller through Simulink.
iv) To be familiar with the complete wiring of your IP02 servo plant, as per dictated in Reference [2] and carried out in pre-laboratory [1].
v) To be familiar with the complete wiring of your LFJC-E module, as described in Reference [3].
vi) To be familiar with LQRs' design theory and working principles.
vii) <u>To be in possession of a basic 30cm ruler.</u>

3 References

[1] *Experiment #1: Control Harware and Software Setup, Signal Interfaces – Student Handout.*
[2] *IP02 User Manual.*
[3] *IP02 – Single Linear Flexible Joint (SLFJ) User Manual.*
[4] *DAQ User Manual.*
[5] *Universal Power Module (UPM) User Manual.*
[6] *QuaRC User Manual (type `doc quarc` in Matlab to access).*
[7] *QuaRC Installation Manual.*
[8] *Experiment #2: Proportional-Derivative Position Control – Student Handout.*

4 Experimental Setup

4.1. Main Components

This laboratory is composed of the following hardware and software components:

• **Power Module:**	Quanser UPM 1503 or 2405, VoltPAQ-X1
• **Data Acquisition Board:**	Quanser Q8 and Q2
• **Linear Motion Servo Plant:**	Quanser IP02,
• **LFJC-E:**	Quanser LFJC-E module,
• **Real-Time Control Software:**	The QuaRC-Simulink configuration, as detailed in Reference [6].

For a complete and detailed description of the main components comprising this setup, please refer to the manuals corresponding to your configuration or consult your lab assistants.

4.2. Wiring

To wire up the system, please follow the default wiring procedures for your IP02 setup, as well as for the LFJC-E module, as fully described in References [2] and [3], respectively. When you are confident with your connections, you can power up the UPM. (If you are uncertain of any parts of the hardware, seek help from your lab assistant)

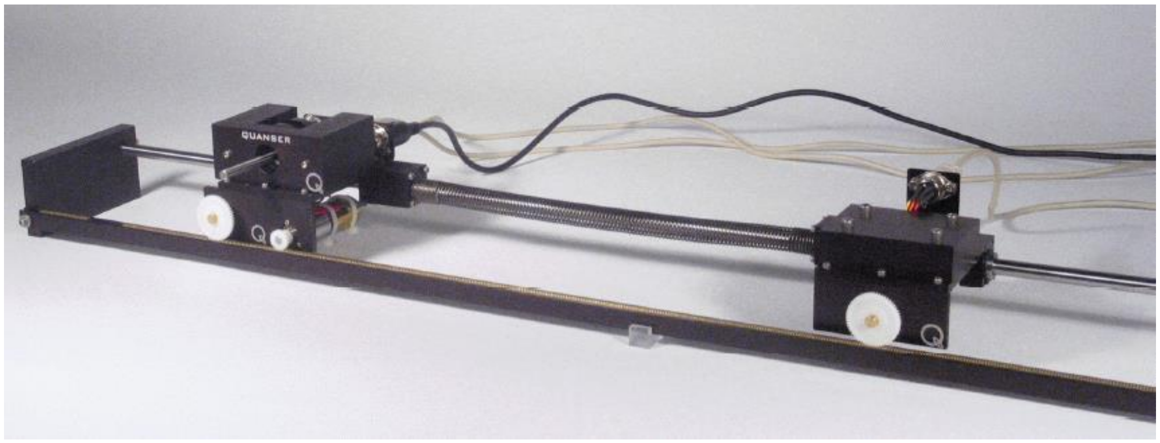


Figure 1 – The IP02 Servo Plant fitted with LFJC-E "Flexible Joint" module

5 Controller Design Specifications

In the preliminary work and in-lab session of the present laboratory, you will design and implement a control strategy based on the Linear Quadratic Regulator (LQR) scheme. As a primary objective, the obtained optimal feedback gain vector, K, should allow you to track the spring-driven load cart to a desired position as quickly as possible while minimizing its overshoot, residual oscillations, and steady-state error. The corresponding control effort should also be looked at and minimized.

Please refer to your in-class notes, as needed, regarding the LQR design theory and the corresponding implementation aspects of it. Generally speaking, the purpose of LQR based optimal control is to allow for best trade-off between performance and cost of control.

We would like to tune the LQR controlling the SLFJ system in order to satisfy the following design performance requirements:

1. The Percent Overshoot (PO) of the load-cart position x_2 along the x-axis from the commanded reference position should be

$$PO \leq 10\ \%$$

2. The 5% band settling-time for the load-cart position x_2 along the x-axis should be

$$t_s \leq 0.6\ secs$$

3. Zero steady-state position error on the load-cart position response x_2:

   $e_{ss} = 0$.

4. Have no saturation in the system. In other words, the commanded motor input voltage V_m (proportional to the control effort produced) should not make the power amplifier (e.g., UPM) go into saturation.

The previous specifications are given in response to a ±20 mm square wave load cart position setpoint.

You should print out preliminary work separately from other parts.

Student Name:
Student ID:

6 Preliminary Work

6.1 Assignment #1: Equations of Motion

A schematic of the Linear Flexible Joint Cart (LFJC-E) mounted on an IP02 linear-cart-and track system, excluding the motor and gearbox models is represented in Figure 2, below. The model of the motor and gearbox components has been derived in the preliminary work for Experiment #2. The LFJC-plus-IP02 system's nomenclature is provided in Appendix A. As illustrated in Figure 2, the positive direction of linear displacement is to the right when facing the cart.

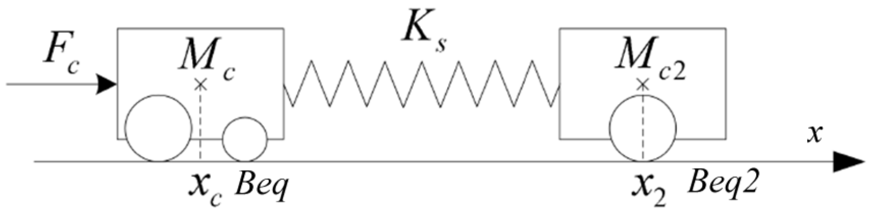


Figure 2 – Schematic of the SLFJ System

Using the terminology given in the appendix and Figure 2, derive the equations of motion for the system. The resulting EOM should have the following format:

$$\frac{\partial^2}{\partial t^2}x_c = \left(\frac{\partial^2}{\partial t^2}x_c\right)(x_c, x_2, F_c) \quad \text{and} \quad \frac{\partial^2}{\partial t^2}x_2 = \left(\frac{\partial^2}{\partial t^2}x_2\right)(x_c, x_2, F_c) \qquad [1]$$

Hint #1:
By neglecting the Coulomb (a.k.a. static) friction of the SLFJ system, the two EOM should be linear. They represent a pure spring-mass system with viscous friction causing the damping.

Hint #2:
You can use the method of your choice to model the system's dynamics. In EE302, we mostly used the free-body diagrams to model mechanical systems. An alternative is the energy based Lagrangian approach. In this case, since the system has two Degrees-Of-Freedom (DOF), there should be two Lagrangian coordinates (a.k.a. generalized coordinates). The chosen two coordinates are namely: x_c and x_2. Also, the input to the system is defined to be F_c, the linear force applied by the motorized cart.

Hint #3:
Ignore the rotational inertia of motor and simply consider the system as seen in Figure 2. The rotational inertia of motor will be considered while calculating the mass of the cart.

6.2 Assignment #2: Linear State-Space Representation

In order to design and implement a LQ Regulator for our system, a linear state-space representation of that system needs to be derived. It is reminded that state-space matrices, by definition, represent a set of linear differential equations that describe the system's dynamics. Since the two EOM of the SLFJ (a pure spring-mass-damper system), as found in Assignment #1, should already be linear, they can be transformed into matrix notation.

The state vector of the mechanical system X is often chosen to include the generalized coordinates as well as their first-order time derivatives. Taking the state vector X as

$$X^T = \left[x_c(t), x_2(t), \frac{d}{dt} x_c(t), \frac{d}{dt} x_2(t) \right] \qquad [2]$$

1. Determine the linear state-space representation of the system. That is, from your EOM determined above, determine the **A** and **B** matrices in

$$\frac{\partial}{\partial t} X = AX + BU \qquad [3]$$

where we have the input to the mechanical system U initially given by $U = F_c$.

A = **B** =

2. From the system's state-space representation previously found, transform the matrices *A* and *B* for the case where the system's input *U* is equal to the IP02 cart's DC motor voltage V_m, instead of the linear force F_c. The system's input *U* can now be expressed by $U = V_m$.

   Hint: In order to convert the previously found force equation state-space representation for the motor voltage input, it is reminded that the driving force, F_c, generated by the DC motor and acting on the cart through the motor pinion has already been determined in previous laboratories. You should be able to verify that F_c can be expressed by:

$$F_c(t) = -\frac{\eta_g K_g^2 \eta_m K_t K_m \left(\frac{d}{dt} x_c(t)\right)}{R_m r_{mp}^2} + \frac{\eta_g K_g \eta_m K_t V_m(t)}{R_m r_{mp}} \qquad [4]$$

A = **B** =

3. Now, numerically evaluate the matrices *A* and *B* as found in Question 2, that is to say in case the system's input *U* is $U = V_m$.

   Hint1:
   Evaluate matrices *A* and *B* by using the model parameter values given in References [2] and [3]. The system configuration you are going to use in your in-lab session is composed of a LFJC-E module connected to an IP02 motorized cart.

Hint2:

Calculate masses of carts as follows:

$$M_c = M_{cart\_IP02} + M_{extra\_weight\_IP02} + \frac{\eta_g K_g^2 J_m}{r_{mp}^2} + M_{spring\_connecting\_piece} + \left(M_{spring}/2\right)$$

$$M_{c2} = M_{cart\_LFJ} + 2 * M_{extra\_weight\_LFJ} + M_{spring\_connecting\_piece} + \left(M_{spring}/2\right)$$

Hint3:

Beq of a IP02 cart with extra mass is not given in the user manual of IP02. Take Beq with extra load as 5.4.

M_c =

M_{c2} =

A = **B =**

4. Calculate the open-loop poles from the system's state-space representation, as previously evaluated in Question 3. Is the system stable? Justify your answer.

What is the type of the system? Justify your answer.

Considering the output matrices $C_1 = [1\ 0\ 0\ 0]$ and $C_2 = [0\ 0\ 1\ 0]$, comment on the system's observability.

What can you infer regarding the system's dynamic behavior? Do you see the need for a closed-loop controller? Explain.

6.3 Assignment #3: Free Oscillation Differential Equation

The theoretical results derived in this assignment will be used in your in-lab session to finely estimate your system parameter values K_s and B_{eq2} from your own experimental data. This process of estimating parameters of a model from experimentation is known as "System Identification". Answer the following questions:

1. Let us consider a linear (i.e., one-dimensional) spring-damper-mass system, of parameters K_s, B_{eq2}, and M_{c2}, respectively. Also let us call $x_2(t)$ the output displacement of the mass centre of gravity as a function of time. Determine the ordinary differential equation (ODE) describing the free-oscillatory motion of the mass.

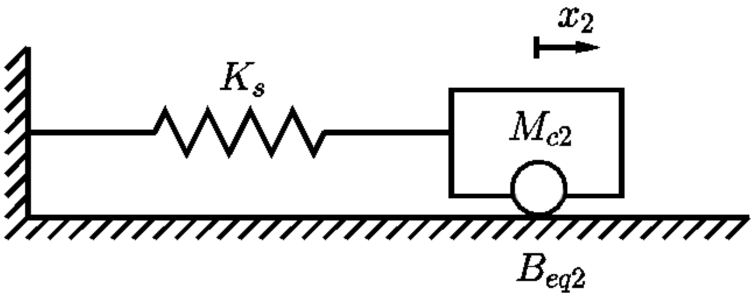


Figure 3 – Mass-Spring-Damper System corresponding to load-cart

2. As you probably found in Question 1, the ordinary differential equation (ODE) describing the free-oscillatory motion of the mass is a second-order differential equation. Show (by determining ζ and ω_n <u>as a function of</u> the given three system parameters K_s, B_{eq2}, and M_{c2}) that this ODE, leads to the following characteristic equation:

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \qquad [5]$$

Together with the final functions, also provide any relevant details in the box below.

ω_n =

ζ =

Note:
Remember from your EE302 lectures that for an underdamped system ($\zeta < 1$), the roots of the quadratic Equation [4] are a complex conjugate pair given as

$$s_1 = -\zeta\,\omega_n + j\,\omega_n\sqrt{1-\zeta^2} \quad \text{and} \quad s_2 = -\zeta\,\omega_n - j\,\omega_n\sqrt{1-\zeta^2} \qquad [6]$$

3. Let us assume now that ω_n and ζ can be determined from experimental measurements. M_{c2} can also be easily measured experimentally (but will be given for this experiment). Therefore, determine from your findings in question 2 the two relationships giving K_s and B_{eq2} as functions of the experimentally measured system parameters ζ, ω_n, and M_{c2}. Also provide your steps in the box provided below.

K_s =
B_{eq2} =

4. Now, we will proceed by defining how to determine ω_n and ζ from experimental measurements. Suppose now that the spring-mass-damper system depicted in Figure 3 is subjected to a force impulse input. This input can be emulated in the form of starting the system with a compressed spring (corresponding to a non-zero energy stored in the system). The effect of an impulse on such a system is similar, namely it instantaneously sets the system to non-zero initial conditions. Figure 4 illustrates the emulation of this

force impulse response by starting the system from a compressed spring state. The x-axis position of the load-cart is zero when the spring is at rest. This is also the steady-state value of the load-cart position response.

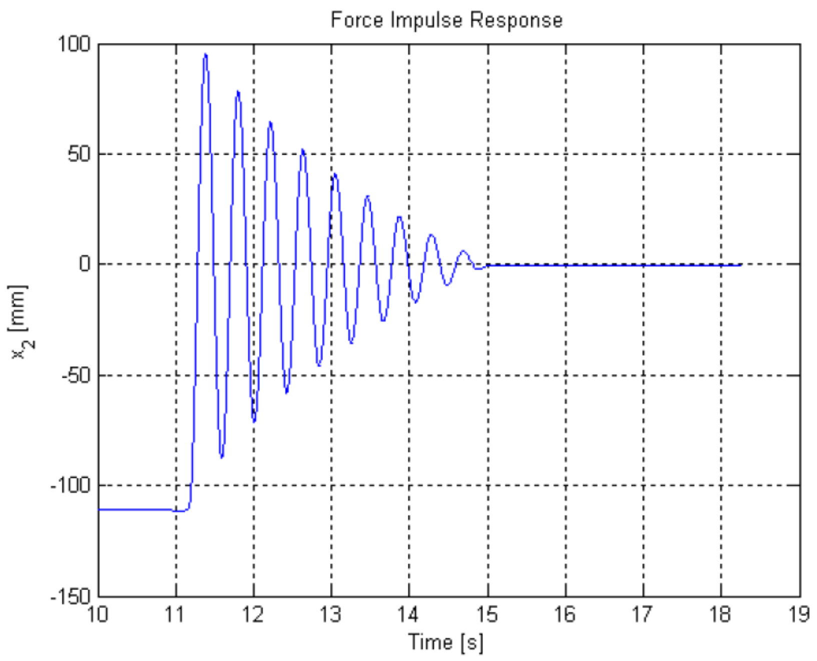


Figure 4 – Typical force impulse response of the system given in Figure 3

5. Let us start by estimating ζ. Based on measurements from the response curve given in Figure 4. In your EE302 lectures, review the section where you have computed the "peak-time" for a second order, under-damped system. There, the approach was to compute the derivative of the time response of the system and equate to zero. This would give you and expression for all the positive and negative peaks of the decaying sinusoidal response. In EE302, you have found that successive peaks (both positive and negative) occur at

$$\omega_d t = k\pi, \qquad [7]$$

where, $k = 1,2,\ldots$ Also remember that the "envelope" of the decaying sinusoid is given by an exponential function of the form

$$|x_2(t)| \leq Aexp(-\zeta\omega_n t). \qquad [8]$$

From these results, together with the relation between ω_n and ω_d, compute, in closed form, the ratio of the amplitudes of the first two positive peaks (R_p) in the damped sinusoidal response for the system as a function of ζ. You will use this expression to compute ζ once you measure the magnitudes of these two peaks from the experimental response.

$R_p = f_1(\zeta) =$

Now, from your above result, derive the expression for ζ as a function of the ratio of the first two positive peaks R_p.

$\zeta = f_2(R_p) =$

7. At this point, you are ready to determine the damped natural frequency w_d when you can experimentally measure the oscillation period of the damped sinusoidal response. Additionally, knowing ζ, you can also determine the undamped natural frequency ω_n.

8. Once you have the values for ζ and ω_n, you can proceed to determine the system parameters K_s and B_{eq2} hence completing the system identification problem. As part of this step, consolidate all your above results in the form of a Matlab function: `[Ks, Beq2] = Identify(p1, p2, Period)` where `p1` and `p2` are the amplitudes of the first two positive peaks in the experimental response and `Period` is the period of the damped sinusoidal signal. You will use this Matlab function during the actual experimental procedure. Save it into a USB drive and keep it with you at all times.

7 In-Lab Experimental Procedure

7.1 Experimental Setup

Even if you don't configure the experimental setup entirely yourself, you should be at least completely familiar with it and understand it. If in doubt, refer to References [1], [2], [3], [4], [5], and/or [6].

The first task upon entering the lab is to ensure that the complete system is wired as fully described in References [2] and [3]. You should have become familiar with the complete wiring and connections of your IP02 system during the preparatory session described in Reference [1]. If you are still unsure of the wiring, please ask for assistance from the Teaching Assistant. Note that in this experiment, different from the previous experiment, there is a second cart, with only an encoder sensor on it. This encoder should be connected to Encoder Channel 2 of the Q2(Q8) break-out board. When you are confident with your connections, you can power up the UPM. You are now ready to begin the lab.

7.2 System Free Response to a Force Impulse Test

7.2.1 Objectives

- To perform a force impulse test on your LFJC-E system, with the IP02 cart being clamped.
- To record and analyze the resulting LFJC free-oscillatory position in response to the initial force impulse.
- To estimate, from the obtained force impulse experimental data, the system's parameters K_s and B_{eq2} that correspond to your particular plant setup.

7.2.2 Experimental Procedure

Please follow the steps described below:

Step 1. Clamp your IP02 cart to one of the rack end plates so that it cannot move. (You can also do this with manually holding the cart against the end plate just while releasing the second cart as described below.)

Step 2. In order to acquire your LFJC position data during the force impulse test, create a Simulink diagram containing a scope and interfacing to the encoder channel for your second cart position measurement. An example of such a diagram is depicted in Figure 7, below. Also, configure your scope to save data. See Reference [6] for more information on how to stream data to Matlab. To determine the K_EC_LFJC gain factor of the encoder, you may need to design a small experiment with your ruler.

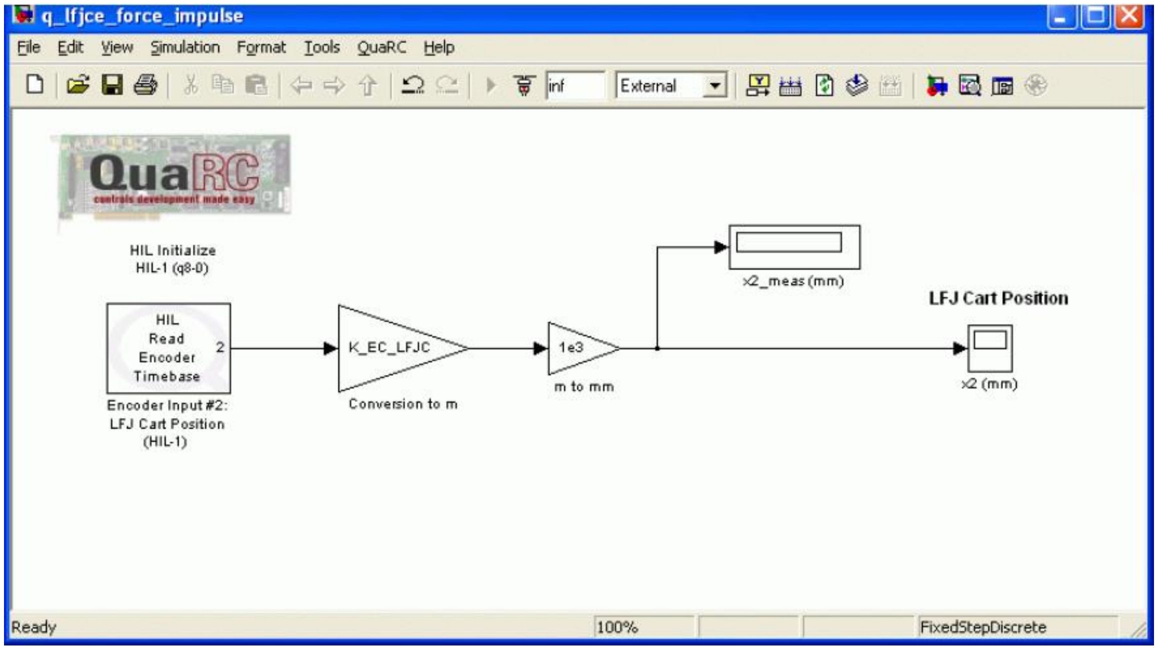


Figure 5 – Diagram containing a Scope to acquire the LFJC-E Second Cart position data

Step 3. You can now compile your diagram for QuaRC. Open the `x2_(mm)` scope and adjust the time scale to a minimum of 8 seconds.

Step 4. To start QuaRC, click on the `QuaRC | Start` from the Simulink model menu bar. At this point in time, your linear spring should be at rest and your LFJC position initialized to zero.

Step 5. The IP02 cart being clamped (or firmly held against the face plate by hand), you can now fully compress the linear spring by pushing on the LFJC Second Cart. Such an initial displacement for the second cart preloads the linear spring. The spring restoring force can then create the initial impulse force needed to start the free oscillations of your LFJC spring-damper-mass system.

Step 6. Wait for the beginning of a trace on the scope before suddenly letting your second cart go. This sudden release of the compressed linear spring generates an impulse force. The scope should now be recording the second cart oscillatory motion. Lastly when the oscillations have significantly decayed and before the trace reaches the end of the scope buffer, stop the controller by clicking on the `Stop` button in the Simulink model tool bar. The data will stop being recorded and your test data will be saved in Matlab. Your scope should now show a trace similar to the one shown in Figure 4 in your preliminary work.

Step 7. To save the Matlab variable to disk use, for example, the command:

```
i. save 'force_impulse_data.mat' 'data_x2'
```

This would save the variable `data_x2` (saved from the scope) to a MAT file called `force_impulse_data.mat` and have your experimental data available for further analysis. This is in fact not necessary at this step since you will just make some measurements from the plot by using the zoom feature of the plot.

Step 8. Now, measure the first two positive peaks p1 and p2 as well as the damped oscillation period from the Matlab plot. Call the `identify(.)` function you have implemented in the preliminary to obtain the K_s and B_{eq2} parameters. Indicate them in the space provided.

Step 9. Compare your experimentally determined K_s and B_{eq2} values with the ones provided in the specification table of Reference [3]. Comment on any possible discrepancies.

Step 10. Assuming you have obtained reasonable values for these two parameters, during the experiment, use these experimentally obtained parameter values to reevaluate your state-space model matrices **A** and **B**.

7.3 System Free Response to a Force Impulse Test

7.3.1 Objectives

- To use the obtained LFJC-plusIP02 state-space representation to design Linear Quadratic Regulator (LQR) type controller.
- To tune on-the-fly and iteratively the LQ Regulator by simulating the closed-loop system in real-time.
- To infer and comprehend the basic principles at work during LQR tuning.

Note:
Please refer to your in-class notes regarding the LQR design theory and associated working principles.

7.3.2 Experimental Procedure

Please follow the steps described below:

Step 1. If you have not done so yet, you can start-up Matlab now and open the Simulink diagram titled `s_lqr_slfj.mdl`. You should obtain a diagram similar to the one shown in Figure 8, below.

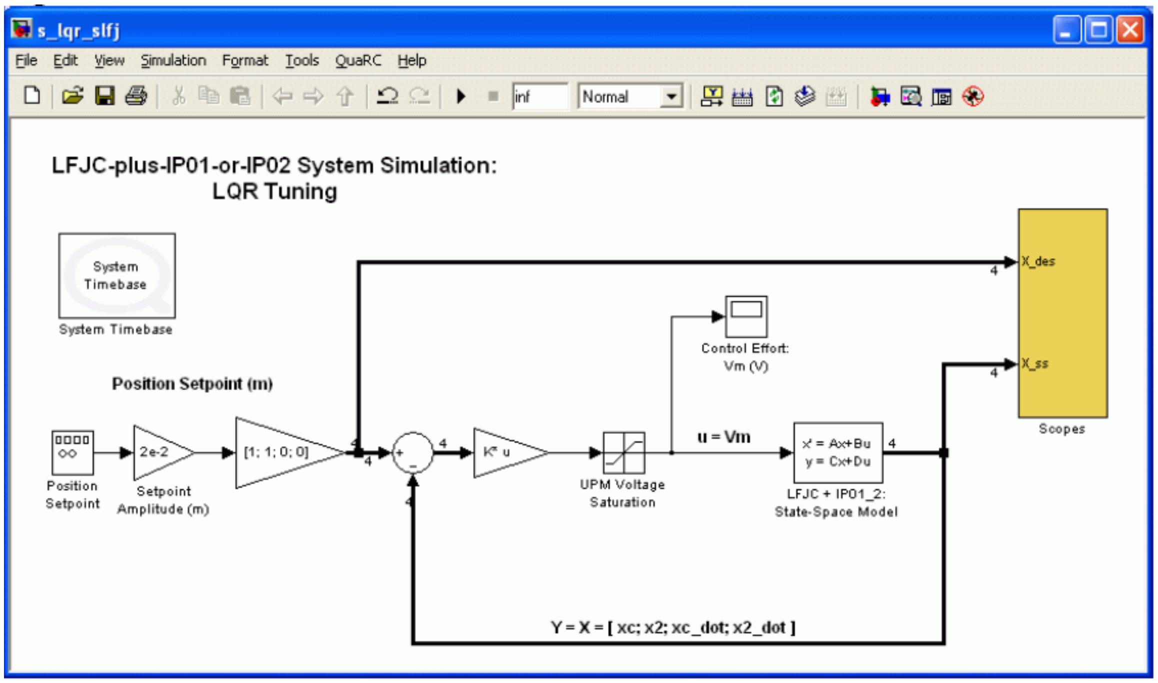


Figure 6 – LQR Tuning Simulation Diagram

Take some time to familiarize yourself with this model: it will be used for the LQR tuning simulation. You should check that the signal generator block properties are properly set to output a square wave signal, of amplitude 1 and of frequency 0.33 Hz. Set the *Stop Time:* in the Simulink *Simulation parameters...* (Ctrl+E) to 5 seconds. By definition, it is reminded that the LQR feedback vector, named *K*, has four elements, corresponding to the four system states defined in Equation [2].

Step 2. Before beginning the real-time simulation, you must run the Matlab script called `setup_lab_ip01_2_slfj.m`. However, ensure beforehand that the *CONTROLLER_TYPE* flag is set to *'MANUAL'*. This mode initializes, before starting on-line the tuning procedure, the optimal gain vector *K* to zero, i.e. [0,0,0,0]. The script also initializes all the LFJC and IP02 model parameters and user-defined configuration variables needed and used by the Simulink diagram. Lastly, it also calculates the state-space matrices, **A** and **B**, corresponding to the LFJC-plus-IP02 system configuration that you defined. Check that the **A** and **B** matrices thus set in the Matlab workspace correspond to the ones that you evaluated in Assignment #2. You may wish fine-tune **A** and **B** to account for your parameter estimation of K_s and B_{eq2}, as previously carried out. It is reminded that the LQR design is based on the plant's linear state-space model. (Therefore, a linearization of the plant model would be needed if the system is originally nonlinear)

Step 3. You can now start your LQR closed-loop simulation in real-time.

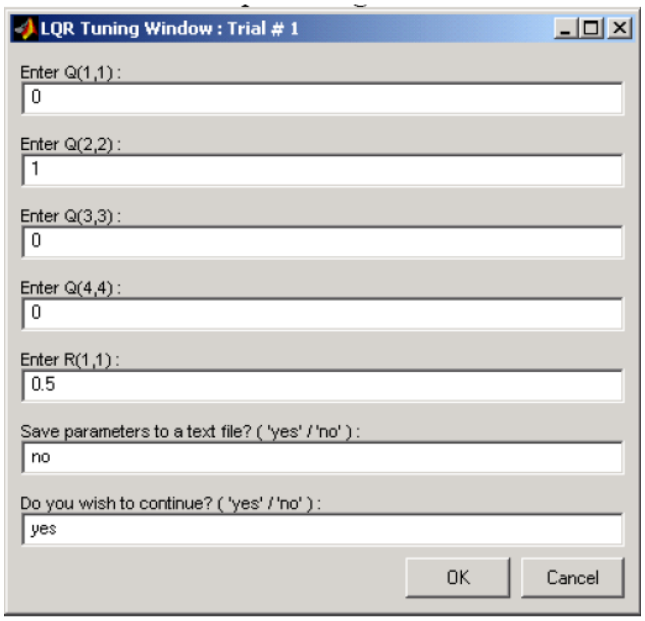
Step 4. In order to observe the system's responses from the real-time simulation, open the three following scopes: `Scopes/xc (mm)`, `Scopes/x2 (mm)`, and `Control Effort: Vm (V)`. You should now be able to monitor on-the-fly the simulated position of the two carts as it tracks your pre-defined reference input, as well as the corresponding control effort. However, at this stage, the responses obtained should all be zero since the feedback gain vector has been initialized to zero.

Step 5. To gain even more insight on the system's behavior you can also open the Scope named `Scopes/dx (mm)`. This scope displays the variation in the linear spring length, as defined below:

$$dx(t) = x_2(t) - x_c(t) \qquad [9]$$

Equation [9] shows that an extension of the linear spring corresponds to $dx > 0$, while a spring compression is equivalent to $dx < 0$.

Step 6. You are now ready to start the on-line tuning of your LQR type controller. First, the Matlab script `setup_lab_ip01_2_slfj.m` should be edited in order to set the *CONTROLLER_TYPE* flag to *'LQR_GUI_TUNING'*. Press F5 to execute the modified file. In this mode, the `setup_lab_ip01_2_slfj.m` script calls the custom function `d_gui_lqr_tuning.m` that creates a Matlab input dialog box similar to the one shown in Figure 9, below.


LQR Tuning Window : Trial # 1

Enter Q(1,1) :
0

Enter Q(2,2) :
1

Enter Q(3,3) :
0

Enter Q(4,4) :
0

Enter R(1,1) :
0.5

Save parameters to a text file? ('yes' / 'no') :
no

Do you wish to continue? ('yes' / 'no') :
yes

Figure 7 – On-Line LQR Tuning Dialog Box

The dialog box pictured in Figure 9 allows the user to enter the diagonal elements of the LQR weighting matrices **Q** and **R**. At this stage, the user, i.e., you, are involved in an iterative trial and error simulation loop. Once the tuning values of **Q** and **R** from the dialog box have been validated by clicking on the OK button, the tuning procedure then automatically calculates the corresponding LQR feedback vector **K**, by calling the Matlab function `'lqr'`, from the Matlab's Control System Toolbox. The effect of the newly calculated feedback vector can be seen right away in the real-time simulation run. At this point, the tuning dialog box of Figure 9 re-appears on the screen for a new trial, if desired/needed. The user can click on the Cancel button (or write *'no'* in the last text input field) at any time to stop the tuning iterations. Also, if the user is satisfied by the simulated response of a particular **Q** and **R**, those tuning values can be saved to a text file, called `lqr_tuning_logfile.txt`, for future reference. To do so, the user just needs to write *'yes'* in the `Save parameters to a text file ('yes' / 'no'):` input field.

Step 7. Keep the initial, un-tuned, values for **Q** and **R** and click OK. This will compute the corresponding LQR feedback vector **K**. Observe the effect of the newly determined **K** on the system simulated responses displayed in the three scopes previously opened.

Step 8. In brief, the LQR tuning principle is as follows: by modifying the elements in **Q** and **R**, one can change the relative gain/weight of the error in each particular state component, and the amount of control effort supplied by the input (i.e., the energy spent). The objective of this simulation is to make you infer, feel, and comprehend the basic principles at work during the LQR tuning for the LFJC-plus-IP02 system.

Now, you should decrease and/or increase the values of *Q(1,1)*, *Q(2,2)*, *Q(3,3)*, *Q(4,4)*, and *R(1,1)*, individually, or in conjunction, and observe the resulting effect on the three system responses simulated in real-time and being plotted on the scopes. Also, to keep a linear behavior, ensure that the motor voltage V_m never goes into saturation.

Try to figure out trends on how the five tuning parameters *Q(1,1)*, *Q(2,2)*, *Q(3,3)*, *Q(4,4)*, and *R(1,1)*, affect the simulated two cart positions, and control effort spent. Provide plots as needed to support your conjunctures.

Hint: You can also refer to the definition of the LQR cost function to be minimized. (MATLAB prompt => "help lqr")

Step 9. Now, finalize the tuning procedure in order to meet the desired closed-loop specifications as stated at the beginning of the preliminary work. Remember that to avoid system saturation, V_m should always be within ±10V.

a) What are your final performance variables?

b) What are your final feedback gain vector, *K*, satisfying the design requirements, and the corresponding weighting matrices **Q** and **R**?

c) Calculate the location of the corresponding closed-loop poles. Compare them to the location of the open-loop poles found in Assignment #2.

d) Sketch the resulting response plots from the Simulink simulation of the system.

Step 10. Once you found acceptable values for **Q** and **R** satisfying the design requirements, save them for the in-lab session as well as the corresponding value of the feedback gain vector *K*. Have your T.A. check your values and simulation plots.

Once you feel comfortable regarding the working principles of LQR tuning and you found acceptable values for **Q** and **R** satisfying the design requirements, you can proceed to the next section. The following section deals with the implementation in real-time of your LQR closed-loop on the actual LFJC-plus- IP02 system.

7.4 Real-Time Implementation of the LQR

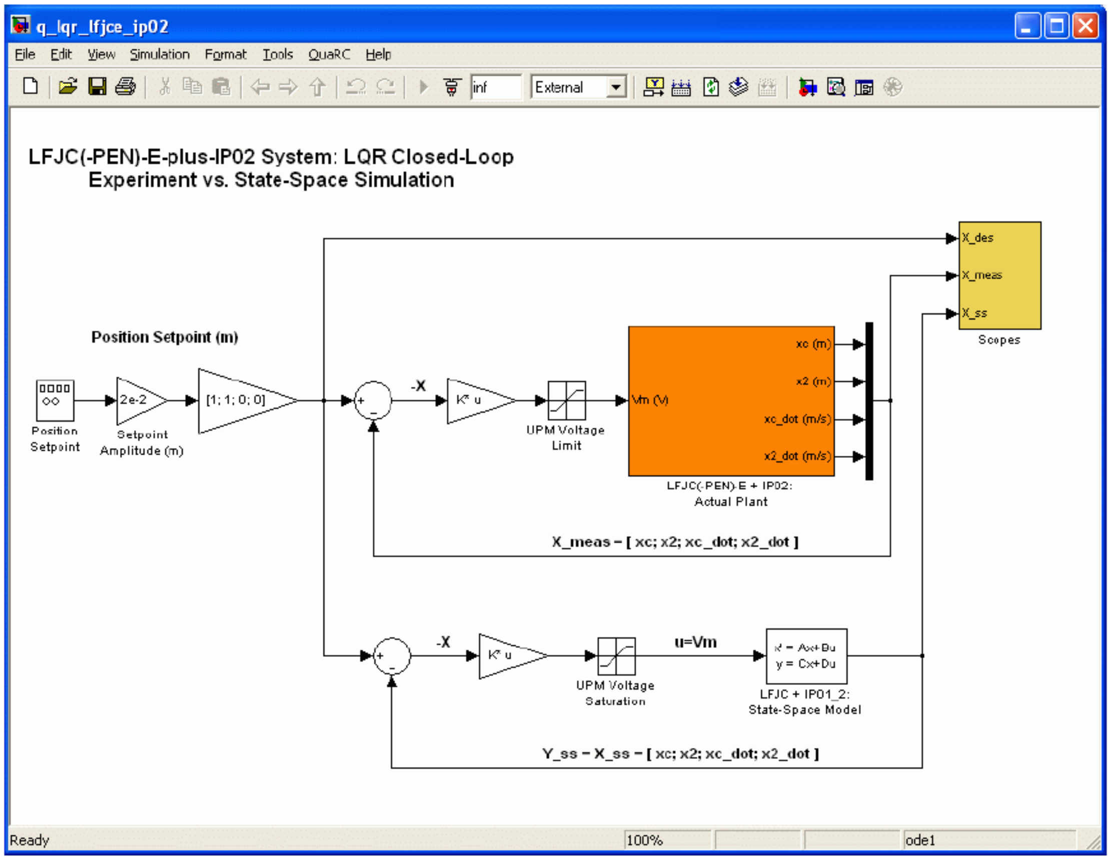
7.4.1 Objectives

- To implement with QuaRC a real-time LQR for your actual LFJC-plus-IP02 plant.
- To tune, on-the-fly and iteratively the LQR observing your actual system response.
- To run the LQR closed-loop system simulation in parallel and simultaneously, in order to compare the actual and simulated responses.

7.4.2 Experimental Procedure

After having gained insights, through the previous closed-loop simulation, on the LQR tuning procedure for your LFJC-plus-IP02 plant and checked the type of responses obtained from the system (e.g., x_c, x_2, V_m), you are now ready to implement your designed controller in real-time and observe its effect on your actual SLFJ-E system. To achieve this, please follow the steps described below:

Step 1. Open the Simulink model file of type `q_lqr_lfjce_ip02.mdl`. Ask your lab Assistant if you are unsure which Simulink model is to be used in the lab. You should obtain a diagram similar to the one shown in Figure 8 10.



q_lqr_lfjce_ip02
File Edit View Simulation Format Tools QuaRC Help
inf
External
LFJC(-PEN)-E-plus-IP02 System: LQR Closed-Loop
Experiment vs. State-Space Simulation
Position Setpoint (m)
Position Setpoint
2e-2
Setpoint Amplitude (m)
[1; 1; 0; 0]
-X
K* u
UPM Voltage Limit
Vm (V)
xc (m)
x2 (m)
xc_dot (m/s)
x2_dot (m/s)
LFJC(-PEN)-E + IP02: Actual Plant
X_meas = [xc; x2; xc_dot; x2_dot]
X_des
X_meas
X_ss
Scopes
-X
K* u
UPM Voltage Saturation
u=Vm
x' = Ax+Bu
y = Cx+Du
LFJC + IP01_2: State-Space Model
Y_ss = X_ss = [xc; x2; xc_dot; x2_dot]
Ready
100%
ode1

Figure 8 – Actual Implementation of the LQR Closed-Loop System (i.e., LFJC-E-plus-IP02 with Q2 (Q8)

The model has two parallel and independent control loops: one runs a pure simulation of the LQR-plus-LFJC-plus-IP02 system, using the plant's state-space representation. Since full-state feedback is used, ensure that the **C** state-space matrix is a 4-by-4 identity matrix by entering `'C=eye (4)'` at the Matlab prompt if necessary. The other loop directly interfaces with your hardware and runs your actual LFJC-E connected to your IP02 linear servo plant. To familiarize yourself with the diagram, it is suggested that you open both subsystems to get a better idea of their composing blocks as well as taking note of the I/O connections.

Step 2. Check that the position setpoint generated for the cart position to follow is a square wave of amplitude 20 mm and frequency 0.33 Hz. Lastly, your model sampling time should be set to 1 ms, i.e., $T_s = 10^{-3}$ s.

Step 3. Ensure that your LQR feedback gain vector *K* satisfying the system specifications, as determined in the previous section's simulations, is still set in the Matlab workspace. Otherwise, re-initialize it to the vector you found. ***Hint:*** *K* can be re-calculated in the Matlab workspace using the following command line:

```
>>K=lqr(A,B,diag([Q(1,1),Q(2,2),Q(3,3),Q(4,4)]),R(1,1)).
```

Step 4. You are now ready to build the real-time code corresponding to your diagram, by using the `QuaRC | Build` option from the Simulink menu bar.

Step 5. After successful compilation and download, you should be able to run in real-time your actual system. However, before doing so, **manually move your LFJC-plus-IP02 system to the middle of the track (i.e., around the mid-stroke position) and make sure that it is free to move on both sides.**

Step 6. The real-time code can now be started. To do so, click on `QuaRC | Start` from the Simulink model menu bar. Your load cart position should now be tracking the desired setpoint.

Step 7. From the `Scopes` subsystem block, open the two sinks `x2_(mm)` and `xc_(mm)`. You should also check the system's control effort and saturation, as mentioned in the specifications, by opening the `V Command (V)` scope located in the following subsystem path: `LFJC(-PEN)-E + IP02: Actual Plant/IP02 Plant/`. On the `x2 (mm)` scope, you should now be able to monitor on-line, as the load cart moves, the actual cart position as it tracks your pre-defined reference input and compare it to the simulation result produced by the LFJC-plus-IP02 state-space model. On the `xc (mm)` scope, you should also be able to monitor on-line the actual motorized cart position as it drives your LFJC. You can also compare these actual responses, in their respective scopes, to the simulated (i.e., theoretical) ones resulting from the state-space plant model located in a parallel LQR closed-loop.

Step 8. What are your observations at this point? Does your actual (hardware) LQR closed-loop implementation meet the desired design specifications? If it does not, then you should fine tune the LQR weighting matrices, **Q** and **R**, in order for the actual LFJC-plus-IP02 system to meet (or be as close as possible to) the design requirements. You can do so on-the-fly and in real-time by means of the previously used on-line LQR tuning GUI. First, ensure that the Matlab script `setup_lab_ip01_2_slfj.m` still has the *CONTROLLER_TYPE* flag set to

'LQR_GUI_TUNING'. Press F5 to execute the Matlab script. In this mode, `setup_lab_ip01_2_slfj.m` calls the function `d_gui_lqr_tuning.m`, which creates a Matlab input dialog box similar to the one shown in Figure 9.

Step 9. Try to achieve the design specifications as closely as possible by iterating and refining your manual LQR tuning as many times as necessary. If you are still unable to even come close to the required performance level, ask your T.A. for advice. Also remember to avoid system saturation by monitoring the corresponding control effort spent, by means of the `V Command (V)` scope. Write down your modifications and their effects.

Step 10. Acceptable results at this point are displayed in Figure 11, below, where the solid red trace is the measured load cart position response, and the blue dotted trace is the simulated one.

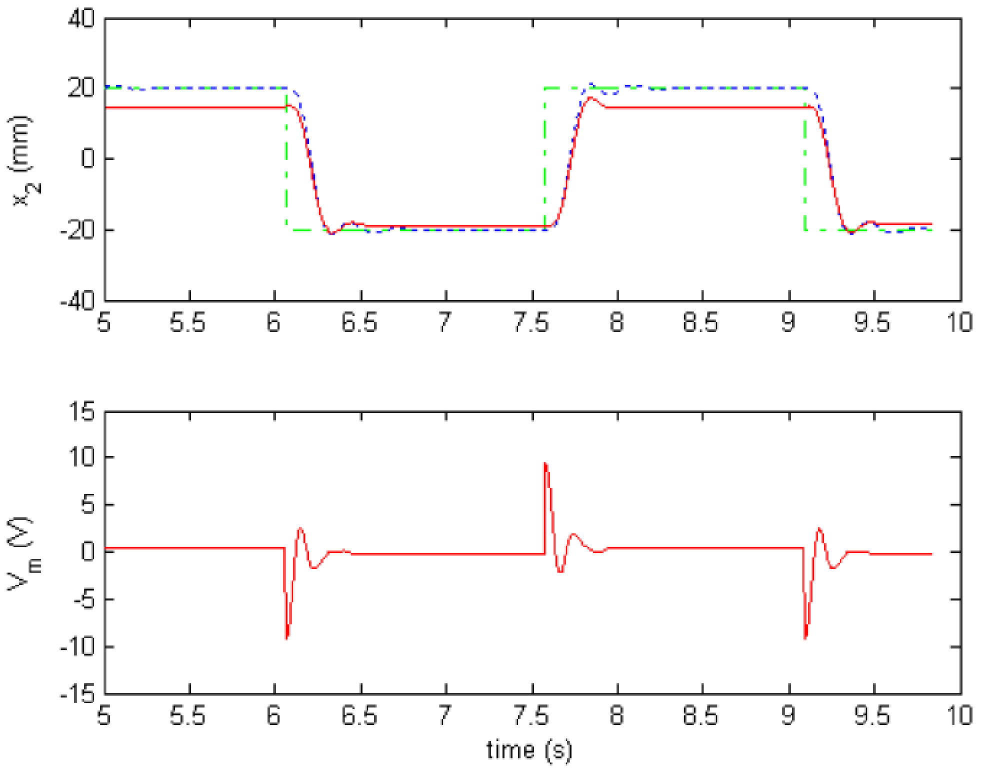


Figure 9 – LQR measured and simulated response. Top plot: desired (dash-dot green), measured (solid red), and simulated (dotted blue) LFJ cart position. Bottom plot: motor input voltage.

Do you notice a steady-state error on your measured load cart position response? Comment on some of the possible reasons for the presence of steady-state error. Are you able to alter your LQR tuning in a way that would eliminate it? If not, can you think of any improvement on the closed-loop scheme that would further reduce, or eliminate, that steady-state error?

Step 11. As you might have already figured out, to meet the zero steady-state error requirement in our SLFJ system, we could introduce integral action on the load cart's position state, x_2. Our goal is to eliminate the steady-state error seen in Figure 11, above. Such an integrator on x_2 has already been implemented for you in a new controller diagram. Open the Simulink model file `q_lqi_lfjce_ip02.mdl`. Ask your Lab Assistant if you are unsure which Simulink model is to be used in the lab. You should obtain a diagram similar to the one shown in Figure 12, below. Apart from the added integrator loop on x_2, this model should be the same in every aspect, including the I/O connections, as the one you previously used (i.e., `q_lqr_lfjce_ip02.mdl`).

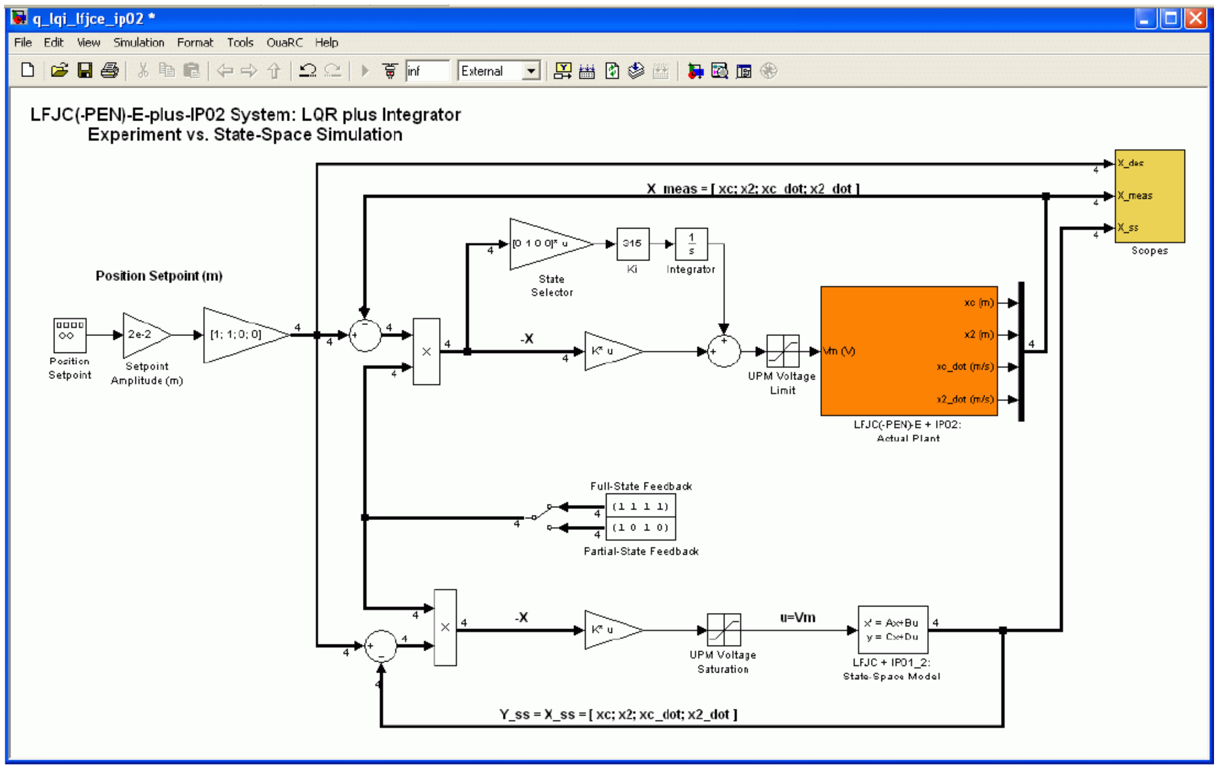


Figure 10 – Integral Action Added to The LQR Closed-Loop for The Actual Plant

Step 12. Compile and run your state-feedback controller with integral action on x_2. Re-open the previous Scopes of interest.

Step 13. Start running the new QuaRC real-time controller. The integral gain is actually a Simulink slider gain block named K_i. Double-click on it to open the slider gain window. You can now use the slider gain window to tune K_i on-the-fly and try to completely eliminate the steady-state error on x_2. Indicate your obtained integral gain K_i in the box provided.

Step 14. Monitor your load cart actual position response as plotted in the `Scopes/x2 (mm)` Scope. Do you achieve a zero-steady-state error? Refine your tuning if you do not. In the space provided, sketch by indicating all critical values, the plots of the *simulated* and *actual* plant responses for the load cart position x_2. Plot the two traces by different line styles to emphasize the differences between the two plots.

Step 15. You should again ensure that the actual commanded motor input voltage V_m never goes into saturation for the system to remain close to linear. (No sign of saturation should be seen on the `V Command (V)` scope inside of the actual plant Simulink block) In the space provided, sketch the motor input voltage plot, indicating all critical values including the positive and negative maximum values.

Step 16. Now, also refine your LQR tuning (i.e., the values for **Q** and **R**) as necessary so that your actual SLFJ closed-loop system performances finally meet the desired design specifications. Iterate your manual tuning as many times as necessary. If you are still unable to achieve the required performance level, ask your Lab Assistant for advice.

Step 17. What are the values that you obtain for *PO*, t_s, e_{ss}, and V_m (minimum and maximum)?

Step 18. What are the final **Q** and **R** matrices, the resulting LQR gain vector *K*, as well as the value for K_i? Ensure to properly document all your results and observations before moving on to the next section.

Step 19. Outline the most prominent differences between the theoretical (i.e., simulated) and actual response. Remember that there is no such thing as a perfect model. Specifically discuss in your lab report the following points:

- First and foremost, how does your actual load cart position compare to the simulated one?
- How does your actual IP02 cart position compare to the simulated one?
- Is there any discrepancy in the results? If so, find some of the possible reasons.

Step 20. In this step, we would like you to observe the disturbance rejection behavior of your designed LQR controller in response to an external force applied to the system around its zero-state. For this you will open the `Position Setpoint` block and set the amplitude of the square wave reference signal to zero. This will cause your controller to track the zero reference, i.e., try to keep both carts around their initial (start-up) positions. The controller will react to any change in the state as a result of an external disturbance.

Step 21. Now open the scopes to observe the load cart position x_2(mm) and the commanded motor voltage V_m(V). apply a small amount of force to the load cart to disturb it from its zero-state. You should apply enough force to change the location of the load cart. Then release your finger to observe the response of the controller. Observe the traces in the two scopes. What happens? Sketch, for a reasonable trial, the two plots in the space provided below. Comment on your observations.

Appendix A. Nomenclature

Table A.1, below, provides a complete listing of the symbols and notations used in the IP01 and IP02 mathematical modeling and controller design presented in this laboratory. The numerical values of the system parameters can be found in the IP02 User Manual.

Symbol	***Description***	***Matlab / Simulink Notation***
V_m	Motor Armature Voltage	Vm
I_m	Motor Armature Current	Im
R_m	Motor Armature Resistance	Rm
K_t	Motor Torque Constant	Kt
η_m	Motor Efficiency	Eff_m
K_m	Back-Electro Motive-Force (EMF) Constant	Km
E_{emf}	Back-EMF Voltage	Eemf
J_m	Rotor Moment of Inertia	Jm
K_g	Planetary Gearbox Gear Ratio	Kg
η_g	Planetary Gearbox Efficiency	Eff_g
M	Total Mass of the IP02 Cart System	M
M_c	Lumped Mass of the Cart System, including the Rotor Inertia	Mc
r_{mp}	Motor Pinion Radius	r_mp
B_{eq}	Equivalent Viscous Damping Coefficient as seen at the Motor Pinion	Beq
F_c	Cart Driving Force Produced by the Motor	
x_c	Cart Linear Position	xc
$\frac{\partial}{\partial t} x_c$	Cart Linear Velocity	xc_dot

Table A.1 IP02 Model Nomenclature

Table A.2, below, provides a complete listing of the symbols and notations used in the mathematical modeling of the Single Linear Flexible Joint (SLFJ) system, as presented in this laboratory. The numerical values of the system parameters can be found in SLFJ User Manual.

Symbol	***Description***	***Matlab / Simulink Notation***
M_{c2}	Mass of the Load Cart System	Mc2
K_s	Linear Spring Stiffness Constant	Ks
B_{eq2}	Equivalent Viscous Damping Coefficient as seen at the Load Cart Position Pinion	Beq2
x_2	Load Cart Linear Position	X3
$\frac{\partial}{\partial t} x_c$	Load Cart Linear Velocity	x2_dot
V_T	Total Potential Energy of the SLFJ System	
Tt_c	IP02 Cart's Translational Kinetic Energy	
Tr_c	IP02 Cart Rotor's Rotational Kinetic Energy	
Tt_2	Load Cart's Translational Kinetic Energy	
T_T	Total Kinetic Energy of the SLFJ System	
Q_{xc}	Generalized Force Applied on the Generalized Coordinate x_c	
Q_{x2}	Generalized Force Applied on the Generalized Coordinate x_2	

Table A.2 SLFJ System Model Nomenclature

Table A.3, below, provides a complete listing of the symbols and notations used in the LQR controller design, as presented in this laboratory.

Symbol	*Description*	*Matlab / Simulink Notation*
A, B, C, D	State-Space Matrices of the LHJC-plus-IP02 System	A, B, C, D
X	State Vector	X
Y	System Output Vector	
PO	Percent Overshoot of the Load Cart Position Response	
t_s	Settling Time of the Load Cart Position Response	
ζ	LFJC Damping Ratio	zeta
ω_n	LFJC Undamped (a.k.a. Natural) Angular Frequency	Wn
ω_d	Damped Angular Frequency of the LFJC Free Oscillations	
T_{osc}	Period of the LFJC Free Oscillations	
O_k	K^{th} overshoot of the LFJC Free Oscillations	
K	Optimal Feedback Gain Vector	K
U	Control Signal (a.k.a. System Input)	
Q	Non-Negative Definite Hermitian Matrix	Q
R	Positive-Definite Hermitian Matrix	R
t	Continuous Time	
dx	Variation in the Linear Spring Length	dx

Table A.3 LQR Controller Nomenclature

8 Report

Names, Student IDs and Signatures of Group Students:

1) ____

2) ____

Date of Experiment:

Name of Lab Assistant:

7.2 System Free Response to a force Impulse test

Step 8:

K_s = B_{eq2} =

Step 9:

Step 10:

A =	**B =**

7.3 System Free Response to a Force Impulse Test

Step 8:

Q(1,1)	Q(2,2)	Q(3,3)	Q(4,4)	R(1,1)	Comment
0	1	0	0	0.5	

Further comments and observations.

Step 9:

a)

PO(%) =

t_s =

V_m (min) =

V_m (max) =

e_{ss} =

b)

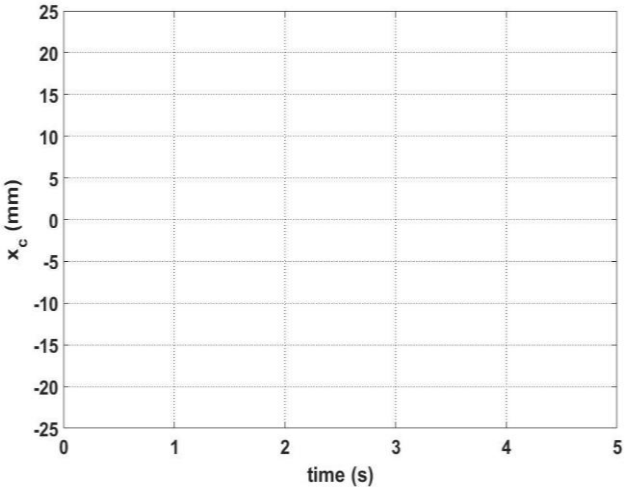
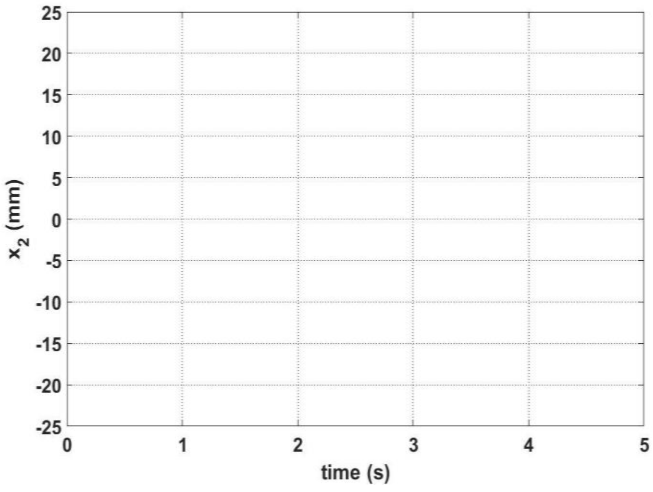
K = [ ]

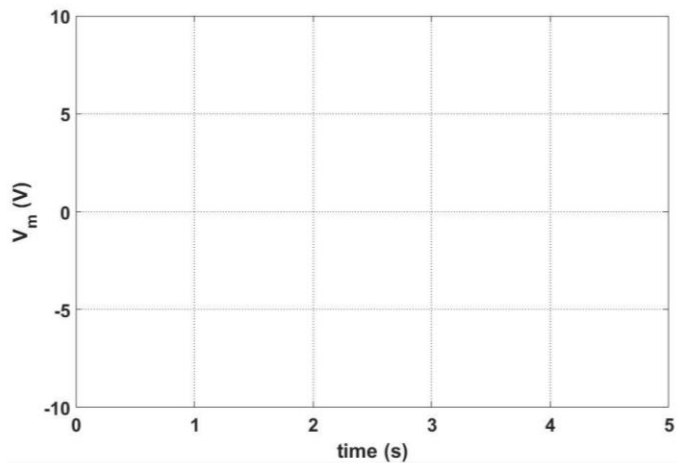
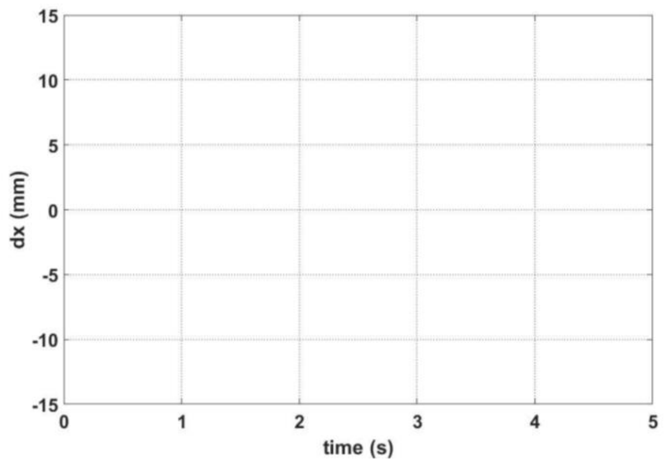
Q = diag([ ])

R = [ ]

c)

d)

First cart position	*Second cart position*
	

Motor voltage	*Spring dx*
	

7.4 Real-Time Implementation of the LQR

Step 9:

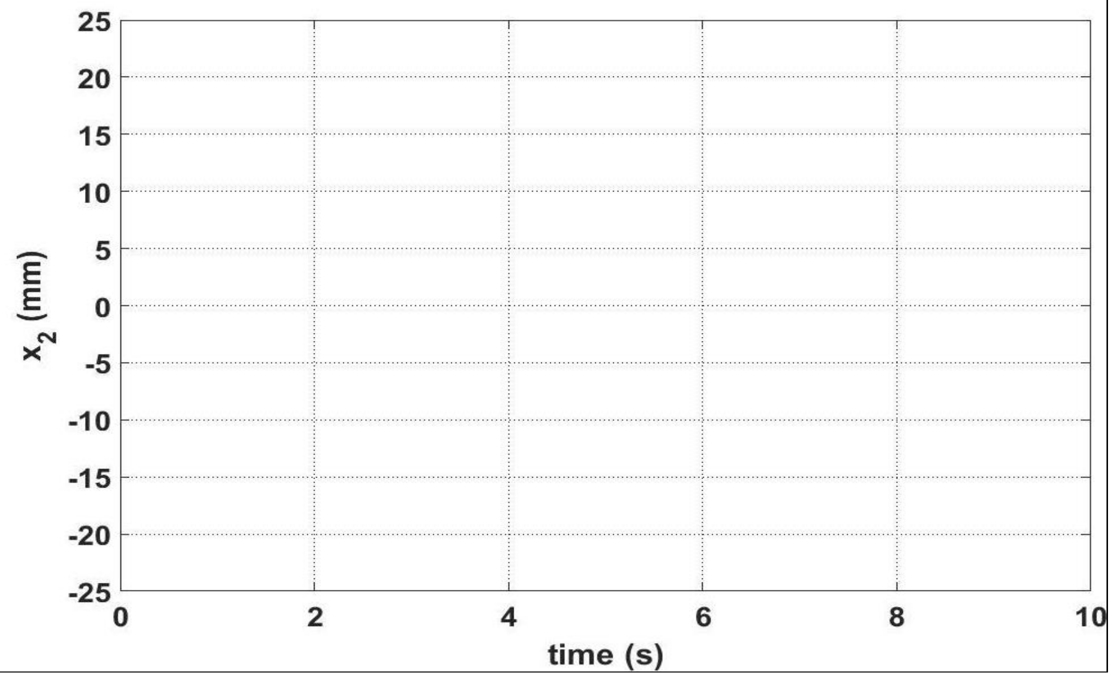
Step 10:

Step 13:

K_i =

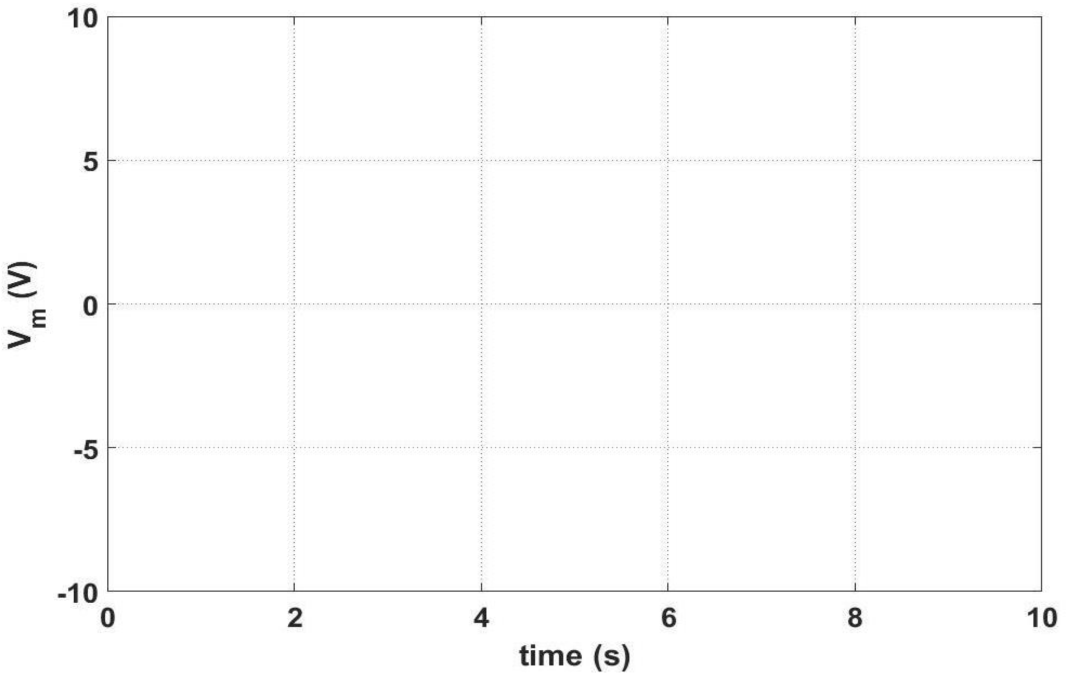
Step 14:

Sketch of the output x_2 for simulated and actual plant



Step 15:

Sketch of the motor input voltage V_m(t)



Step 17:

$PO\%$ =

t_s =

e_{ss} =

V_m(min) =

V_m(max) =

Step 18:

Q = diag([ ])

R = [ ]

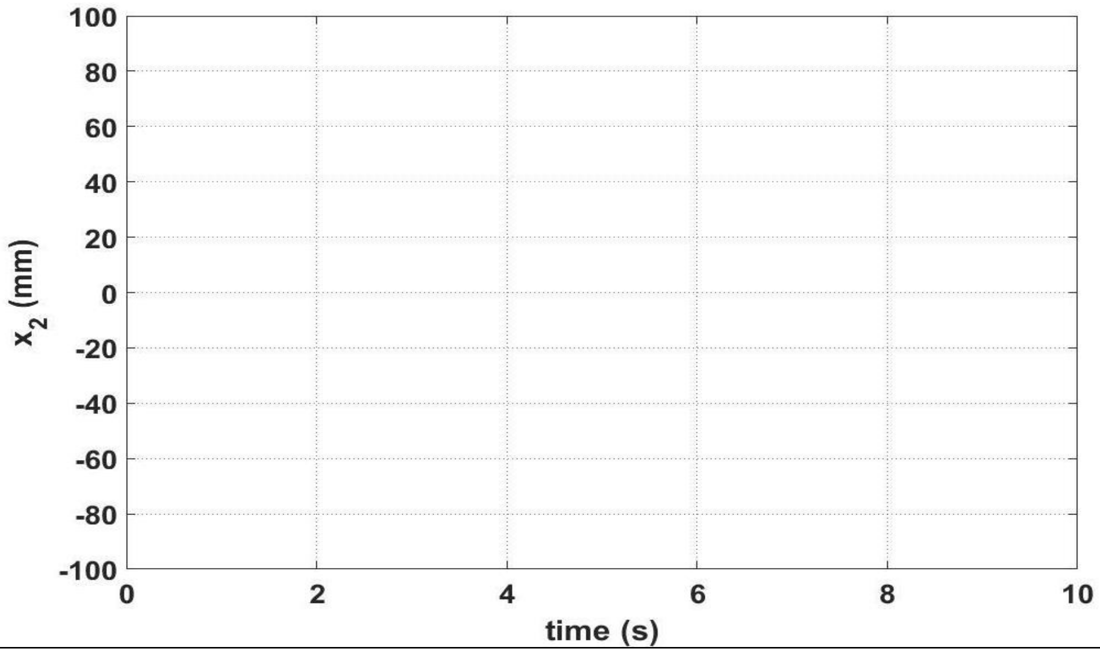
K = [ ]

K_i =

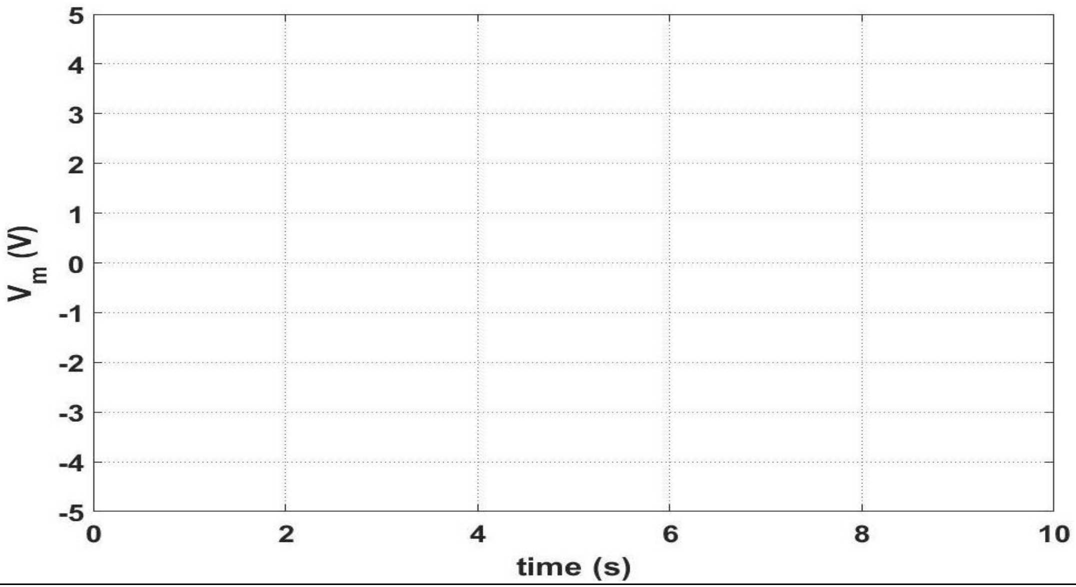
Step 19:

Step 21:

Sketch of the output x_2 in response to external disturbance



Sketch of the motor voltage Vm in response to external disturbance



Comments:

9 Knowledge Test

1. What is the difference between obtaining ABCD matrices of linear and nonlinear systems?

2. What is “System Identification”? What part of this experiment was related with system identification? Give more examples where system identification can be used.

3. Explain your procedure to determine Ks and Beq2.

4. What is the cost function, J, of the LQR you have designed in this experiment?

5. What is the meaning of increasing value of R matrix components in LQR design?

6. What is the meaning of increasing an individual diagonal element of Q matrix relative to the other elements in the same matrix in LQR design?

7. What would be the size of an R matrix for a system where number of states is 4, number of outputs is 3 and number of inputs is 2.

8. What would be the size of Q matrix for the system in question 6.

9. What is the aim of LQR design? Explain the procedure (steps) of LQR design. Also give a basic formulation.

10. What is LQI control? Why is it used?