Middle East Technical University
Department of Electrical and Electronics Engineering

EE 406
Laboratory of Feedback Control Systems

Experiment #6:
Single Inverted Pendulum Control using Linear Quadratic Regulator (LQR)

Preliminary Work and Laboratory Manual

Initial Version: Afşar Saranlı, Emre Tuna

Group Members:

1 Objectives

The Single Inverted Pendulum (SIP) experiment is an example of a naturally unstable nonlinear system that needs to be stabilized by a control system. It is a “real-life example” in the sense that it can be seen as balancing a broomstick on the tip of one's finger, a common and familiar childhood activity. The difference being that the broomstick is in a three-dimensional space while our inverted pendulum setup is restricted on a plane. In this laboratory session, the inverted pendulum is mounted on a linear IP02 cart system. During the course of this experiment, you will again apply the design and tuning principles of a Linear Quadratic Regulator (LQR). The challenge of the present laboratory is to keep a single inverted pendulum balanced around its naturally unstable equilibrium point and at the same time, bring the linear cart to a commanded position. In practice, it is should be pointed out that similar dynamics and control problem applies to many applications including ship roll control, legged robot stabilization as well as vertical missile launch control.

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

i) How to mathematically model the SIP mounted on the IP02 linear servo plant, using, for example, Lagrangian mechanics or force analysis on free body diagrams.
ii) How to linearize the obtained non-linear equations of motion around the unstable equilibrium point of interest.
iii) How to obtain a state-space representation of the open-loop system.
iv) How to design and tune, in a simulation environment, an LQR-based state-feedback controller satisfying the closed-loop system's desired design specifications.
v) How to implement your LQR-based controller in real-time and evaluate its actual performance.
vi) How to tune on-line and in real-time your LQR-based controller so that the actual inverted-pendulum-linear-cart hardware system meets the controller design requirements.
vii) How to observe and investigate the disturbance response of the stabilized inverted-pendulum-linear-cart system, in response to a tap to the pendulum.

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 power amplifier (e.g., UPM), and your data acquisition card (e.g., Q8 or Q2), as described in References [1], [2], [3], and [4].
ii) 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.
iii) To be familiar with the complete wiring of your IP01 or IP02 servo plant, as per dictated in Reference [2] and carried out in pre-laboratory [1].
iv) To be familiar with LQRs' design theory and working principles.

3 References

[1] Experiment #1: Control Hardware and Software Setup, Signal Interfaces: Lab Sheet
[2] IP02 User Manual.
[3] IP02 - Single Inverted Pendulum User Manual.
[4] DAQ User Manual.
[5] Universal Power Module User Manual.
[6] QuaRC User Manual (type `doc quarc` in Matlab to access)
[7] Experiment #2: Proportional Derivative Position Control – Lab Sheet.
[8] QuaRC Installation Manual.

4 Experimental Setup

4.1 Main Components

To setup this experiment, the following hardware and software are required:

- **Power Module:** Quanser UPM 1503 or 2405, VoltPAQ-X1

[9] **Data Acquisition Board:** Quanser Q8 and Q2

- **Linear Motion Servo Plant:** Quanser IP02 cart system
- **Single Pendulum:** Quanser 24-inch Single Pendulum, seen in Figures 1 below.
- **Real-Time Control Software:** The QuaRC-Simulink configuration, as detailed in Reference [8].
- **A Computer to Run Matlab-Simulink and the QuaRC software**

The setup for this laboratory will be mostly prepared for you. Your desk setup will have the necessary components including all necessary software components pre-installed. You will follow the procedure of this laboratory session to setup and use the system.

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

4.2 Wiring

To wire up the system, please follow the default wiring procedure for your IP02 as fully described in Reference [2]. When you are confident with your connections, you can power up the UPM.

Figure 1 – *The IP02 Servo Plant fitted with the 24-inch Pendulum*

5 Controller Design Specifications

In the present laboratory (i.e., the Preliminary Work and the in-lab experiments), 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 keep your single inverted pendulum balanced around its naturally unstable equilibrium point. At the same time, your IP02 linear cart will be required to track a desired (square wave) position signal. The problem requires that the corresponding control effort should also be 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 optimal control is to allow for best trade-off between performance and cost of control.

We would like to tune the LQR controlling the inverted-pendulum-and-linear-cart system in order to satisfy the following design performance requirements:

1. Regulate the pendulum angle around its upright position and never exceed a ±1-degree-deflection from it, i.e.:

$$|\alpha| \leq 1.0 \text{ [deg]}$$

2. Have a maximum rise time t_r, on the cart position response less than 1.5 second, i.e.,

$$tr \leq 1.5 \text{ [secs]}$$

3. Minimize the control effort, represented by the motor input voltage V_m. The power amplifier (UPM) should not go into saturation in any case.

The previous specifications are given in response to a ±20-to-30 mm square wave cart position reference signal.

As a remark, it can be seen that the previous design requirements bear on the system's two outputs: x_c and α. Therefore, our inverted-pendulum-linear-cart system consists of two outputs, for one input.

You should print out preliminary work separately from other parts.

Student Name: Özgür Gülsuna
Student ID: 2307668

6 Preliminary Work

6.1 Review the Non-Linear Equations of Motion

The system we will be studying in this laboratory is very similar to the "Crane System" of Experiment #4. Note that, the regular pendulum equilibrium position there is replaced in this experiment with the pendulum in an upright (inverted) equilibrium position. While the non-inverted equilibrium is a naturally stable one, the upright (inverted) equilibrium is naturally unstable. A schematic of our Single Inverted Pendulum (SIP) mounted on an IP02 cart is illustrated in Figure 2. The set of system parameters and variables relating to this system is given in Appendix A. As given in the figure, axis conventions are such that the positive sign of pendulum rotation is counterclockwise when facing the cart, the cart and pendulum tip positions as well as the input force are defined as positive towards the right. Finally, α=0 is defined such that the pendulum is perfectly perpendicular to the ground (pointing upwards, opposite to the direction of gravity).

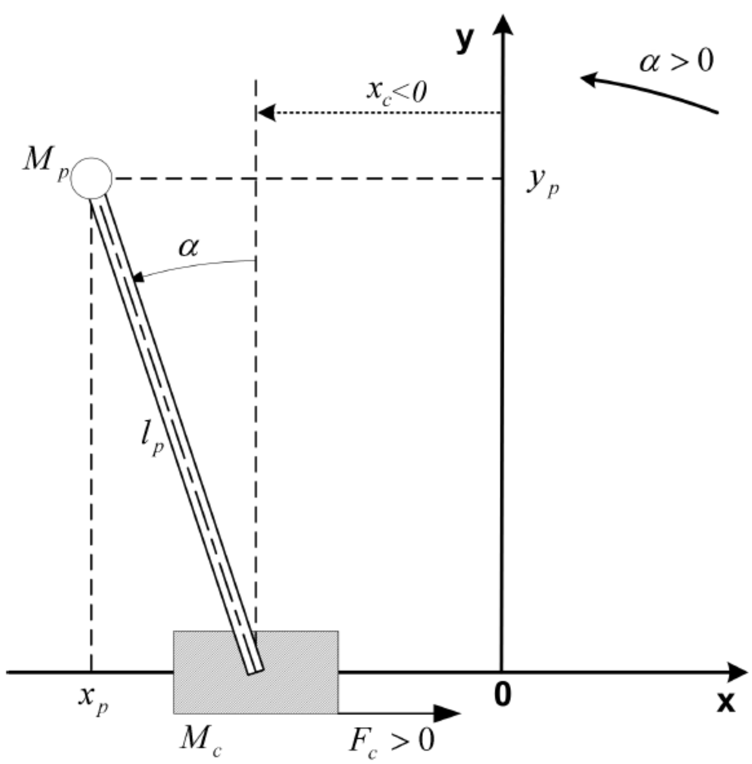


Figure 2 – Schematic of the SIP mounted on the IP02 servo plant

Due to the similarity between the inverted and non-inverted pendulum systems, most of the discussion below follows closely the discussion in Experiment #4. Similar to Experiment #4, this system also has two *degrees-of-freedom* (DOF) indicating the minimum number of variables (*generalized coordinates*) to describe the system configuration. The system state vector is composed of these variables and their derivatives and completely captures the dynamics of the system. It is given as $\mathbf{z} = [z_1\ z_2\ z_3\ z_4]^T = [x_c\ \alpha\ \dot{x}_c\ \dot{\alpha}\ ]^T$ and the task of deriving the system equations of motion amounts to determining the expressions for the two variables $\ddot{x}_c$ and $\ddot{\alpha}$ in the following form

$$\ddot{x}_c = g_3(\mathbf{z}, V_m) \tag{1a}$$

$$\ddot{\alpha} = g_4(\mathbf{z}, V_m) \tag{1b}$$

where g_3 and g_4 are two nonlinear functions of the state and the input. Here, the mechanical system input is again the force F_c to the IP02 cart, which in turn is a function of the overall plant input, i.e., the motor voltage V_m. The relation between F_c and V_m is established through the DC motor model. This means that our overall nonlinear plant has the DC motor voltage V_m as its input, i.e., u. In this experiment, for assessing the control performance, we are interested two of the state variables: cart position x_c and the pendulum angle α. Remember that the controller performance specifications are expressed in terms of these two variables. Note that since we will be using an LQR controller, all of the states will be used in a state-feedback configuration.

The nonlinear state-space representation of the plant is again given by

$$\dot{z}_1 = g_1(\mathbf{z}, V_m) = z_3 \tag{2a}$$

$$\dot{z}_2 = g_2(\mathbf{z}, V_m) = z_4 \tag{2c}$$

$$\dot{z}_3 = g_3(\mathbf{z}, V_m) \tag{2b}$$

$$\dot{z}_4 = g_4(\mathbf{z}, V_m) \tag{2d}$$

while we no longer need to define a plant output (since it is nothing but the selected two states).

The derivation of the system equations of motion corresponding to the mechanical part can be based either on free-body diagrams combined with Newton's Second Law or an energy-based approach using the Lagrangian formulation. A derivation using the latter approach is given in Appendix B. The relation between F_c and V_m is given by the armature-controlled DC motor model. This is also summarized in Appendix B.

The resulting system equations of motion are given by the following equations. The only difference of these equations with the ones in Experiment #4, is sign changes of the $\sin(\alpha)$ and $\cos(\alpha)$ due to the fact that we now have α=0 when the pendulum is pointing to the top, i.e., $\alpha = \alpha_{old} - \pi$. Here, α_{old} is the pendulum angle for Experiment #4.

$$\frac{d^2}{dt^2}x_c(t) = \frac{\left(\begin{array}{l} -\left(I_p + M_p\, l_p^2\right)B_{eq}\left(\frac{d}{dt}x_c(t)\right) - \left(M_p^2\, l_p^3 + I_p\, M_p\, l_p\right)\sin(\alpha(t))\left(\frac{d}{dt}\alpha(t)\right)^2 \ldots \\ + \left(I_p + M_p\, l_p^2\right)\left(-\frac{\eta_g\, K_g^2\, \eta_m\, K_t\, K_m\left(\frac{d}{dt}x_c(t)\right)}{R_m\, r_{mp}^2} + \frac{\eta_g\, K_g\, \eta_m\, K_t\, V_m(t)}{R_m\, r_{mp}}\right)\ldots \\ - M_p\, l_p \cos(\alpha(t))B_p\left(\frac{d}{dt}\alpha(t)\right) + M_p^2\, l_p^2\, g\cos(\alpha(t))\sin(\alpha(t)) \end{array}\right)}{\left(\left(M_c + M_p\right)I_p + M_c\, M_p\, l_p^2 + M_p^2\, l_p^2 \sin(\alpha(t))^2\right)} \tag{4}$$

$$\frac{d^2}{dt^2}\alpha(t) = \frac{\left(\begin{array}{l} \left(M_c + M_p\right)M_p\, g\, l_p \sin(\alpha(t)) - \left(M_c + M_p\right)B_p\left(\frac{d}{dt}\alpha(t)\right)\ldots \\ + \left(-\frac{\eta_g\, K_g^2\, \eta_m\, K_t\, K_m\left(\frac{d}{dt}x_c(t)\right)}{R_m\, r_{mp}^2} + \frac{\eta_g\, K_g\, \eta_m\, K_t\, V_m(t)}{R_m\, r_{mp}}\right)M_p\, l_p \cos(\alpha(t))\ldots \\ - M_p^2\, l_p^2 \sin(\alpha(t))\cos(\alpha(t))\left(\frac{d}{dt}\alpha(t)\right)^2 - M_p\, l_p \cos(\alpha(t))B_{eq}\left(\frac{d}{dt}x_c(t)\right) \end{array}\right)}{\left(\left(M_c + M_p\right)I_p + M_c\, M_p\, l_p^2 + M_p^2\, l_p^2 \sin(\alpha(t))^2\right)} \tag{5}$$

In this formulation, the mass of the pendulum was assumed to be concentrated at the center-of-mass of the pendulum rod. Details of their derivation can be found in Appendix B.

6.2 Linearization and Linear State-Space Approximation

Remember that the LQR based optimal controller design requires a linear state-space representation for the system given by the standard form

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}u \tag{6a}$$
$$y = \mathbf{C}\mathbf{z} \tag{6b}$$

In this experiment, we are not interested in a single output. But if we consider the two states of interest for the system performance as our two outputs (x_c and α), these are already given by a linear relationship in Eq. (6b) and the **C** matrix given by

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

However, since our pendulum system is non-linear in nature, Eq. (6a) needs to be obtained through a linearization procedure.

Finding a linear approximation to the system equations (i.e., *linearization*) is covered in your EE302 lectures and was previously carried out in Experiment #4. The **A** matrix is the Jacobian matrix **J** (essentially composed of the derivatives of the non-linear state functions $g_i(.)$ w.r.t all state variables) evaluated at the equilibrium point **z**=0. This point, under the new α definition, now corresponds to the pendulum-up equilibrium position of the system. Since we have a single input, **B** is a vector also composed of the derivatives of the same functions with respect to this input, evaluated again at **z**=0.

Substituting the numerical values of all system parameters and evaluating the derivatives at the equilibrium point of the system, find the linear approximation of our system.

A =

B =

C =

D =

6.3 Assignment #1: Analysis of Open-Loop System

As a first step, we would like to analyze the behavior of the open-loop system using its linear approximation. The first step is to determine the pole-zero configuration of the system using Matlab. Notice that, we have 2 different outputs in this experiment. Hence, we have 2 different transfer functions. Denominator of these functions will be the same since poles are eigenvalues of A matrix. But numerators, i.e., location of zeros, will be different for these transfer functions. Using `ss2tf` command of Matlab, calculate those transfer functions. Using C matrix in `ss2tf` will yield 2 sets of numerator coefficients. Instead of 2x4 C matrix, first, you can give the 1st row of C matrix (1x4, [1,0,0,0]) and obtain the transfer function, then, give the 2nd row for the other one. Then find open loop pole and zero locations of both transfer functions. You can use `ss2zp` function of Matlab. Finally sketch the open loop pole and zero locations, indicating necessary details.

TF for xc output:

TF for alpha output:

Now you will simulate the behavior of system for initial conditions on Matlab. First define your system on Matlab with following command (Assigning state variables and input-output signals' names is not necessary, but it will make your simulation clearer).

```
sys = ss (A, B, C, D, 'statename', {'xc' 'a' 'xc_dot' 'a_dot'}, …
                              'inputname', 'Vm', …
                              'outputname', {'xc' 'a'});
```

Then, you will obtain the initial condition response of your linearized system when input is set to 0 by using `initial` function of Matlab. Your plots should be until 5 seconds. Try 2 different initial conditions; the 1st one should be zero for all states (xo= [0 0 0 0]), the 2nd one should have a very little non-zero initial condition on alpha (xo= [0 10^{-7} 0 0]). Observe and plot both outputs (xc and alpha) for both of the initial conditions.

What can you comment on the dynamic behavior of the linear approximation to the system? Is the linear approximation stable? What is the effect of a small disturbance around the $\mathbf{z}$=0 equilibrium point? What would you expect to see for alpha if you were using the original non-linear system equations? Also try to sketch an expected alpha vs. time plot for non-zero initial conditions. (You can try to imagine the real setup)

Can a controller be used in order to balance (stabilize) the system around the **z**=0 equilibrium point? Give necessary details and comment.

6.4 Assignment #2: Reading Assignment

From this manual, read the following parts:

7.2. Assessment of the State Space Model Operating Range. (Until 7.2.4)
7.2.1. Simulation Diagram
7.2.2. State-Space Model Implementation
7.2.3. Simulink Representation of the 2 EOMs.

7 In-Lab Experimental Procedure

In the beginning of the lab session, have your Teaching Assistant check your results and sign below.

Teaching Assistant Review Notes:	
Assistant Name:	Signature:

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].

7.1.1 Check Wiring and Connections

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. When you are confident with your connections, you can power up the UPM. You are now ready to begin the lab. The pendulum that we will use in the lab should NOT initially be mounted on the setup. To ensure there is no damage to the pendulum and/or the encoder connection, it should only be mounted when the actual hardware experiment steps are conducted.

7.1.2 IP02 Configuration

This experiment is designed for an IP02 cart without the extra weight on it. However, once a working controller has been tested, the additional mass can be mounted on top the cart in order to see its effect on the response of the system. For the time being, please carefully remove the extra weight if it is mounted on top of the cart.

7.1.3 Single Inverted Pendulum Configuration

This laboratory uses the long single pendulum as a default configuration. However, the same pre-lab and in-lab sessions could be applied, as long as they are consistent, to a different configuration using, for example, the medium single pendulum.
Likewise, once a working controller has been developed and tested for one pendulum, another type of pendulum can be mounted on top the cart in order to observe the resulting system response with regard to, for example, the controller robustness to modeling errors. As an extension to the present lab, the previously designed controller could then be modified in order to account for the new pendulum, of different characteristics.

7.2 Assessment of the State Space Model Operating Range

It is obvious that linearized models, such as the inverted pendulum system state-space model, are only approximate models. Therefore, they should be treated as such and used with appropriate caution, i.e., within the valid operating range and/or conditions. However, they also give valuable insights on the system's dynamics and usually allow for the development of fairly straightforward control strategies.

This section attempts to assess the operating range of the system's state-space representation derived in the preliminary work. It does so by simulating and comparing the inverted pendulum response to an initial non-zero angle (i.e., free fall and free swing) obtained from both state-space model and original nonlinear equations of motion.

7.2.1 Simulation Diagram

Opening the file `s_sip_freefall_ss_vs_eom.mdl` should display a diagram similar to the one shown in Figure 3, below. It implements, in parallel, the two inverted pendulum models previously discussed. Using this Simulink model allows us to compare the two inverted pendulum responses, to a small non-zero initial angle α_0, obtained from the two models.

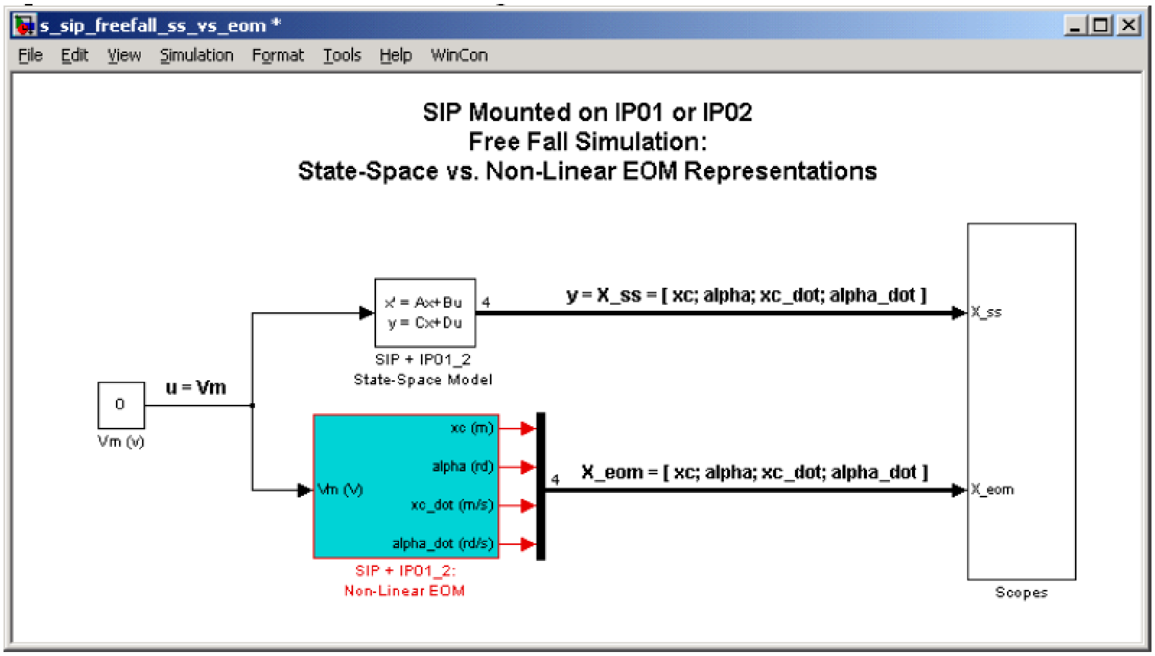


Figure 3 – Simulink diagram for the Free-Fall Comparative Simulation

7.2.2 State-Space Model Implementation

Opening the state-space block named `SIP + IP01_2 State-Space Model` in the `s_sip_freefall_ss_vs_eom.mdl` file should show a dialog box similar to that displayed in Figure 4. The model state-space matrices **A** and **B** should correspond to the matrices evaluated in the Preliminary Work. It should also be noted that the initial condition on α, α_0, is setup inside the *Initial conditions:* input field. That initial angle is initialized in the Matlab workspace as the variable *IC_ALPHA0*.

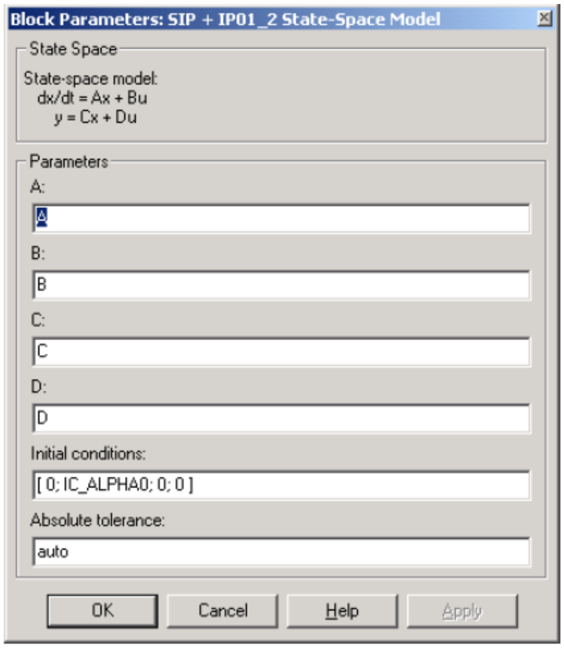


Figure 4 –Linear State-Space model parameters

7.2.3 Simulink Representation of the Two EOM

After obtaining of the system's two EOM in the Preliminary Work, they can be represented by a series of block diagrams, as illustrated in Figure 5, Figure 6 and Figure 7, below. Note that, although not a trivial task, the Simulink block diagram can be transformed into the Equations of Motion given in Eqs. (4) and (5) and vice-versa. Please spend some time to see the relationship between the equations and the three block diagrams given below.

Opening the subsystem block named `SIP + IP01_2: Non-Linear EOM` in the file `s_sip_freefall_ss_vs_eom.mdl` should show something similar to Figure 5. This mostly corresponds to the familiar IP02 open-loop transfer function representation, as previously discussed in, for example, Reference [7].

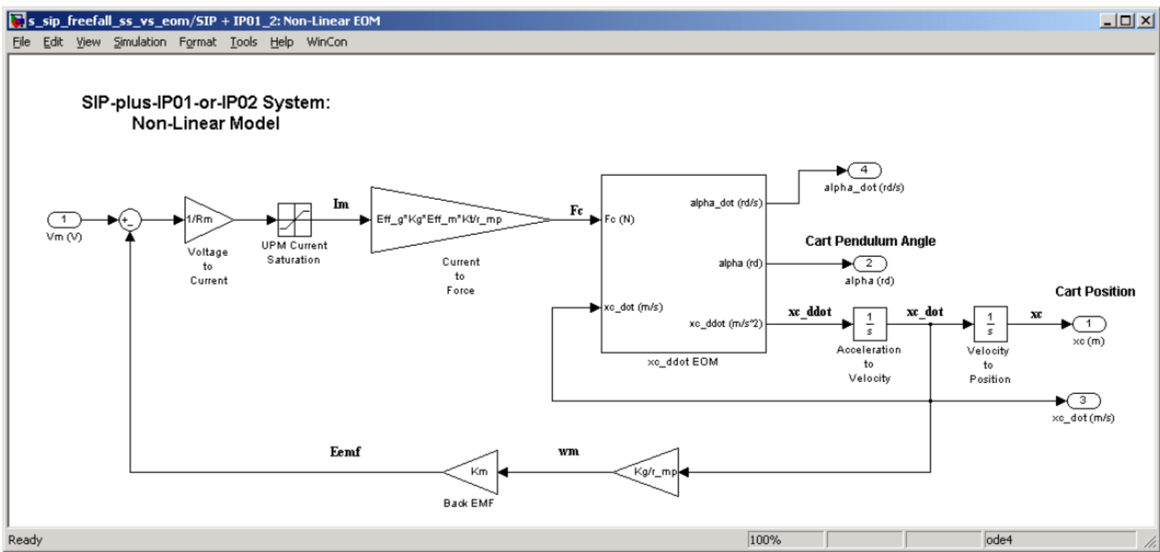


Figure 5 –Simulink sub-system diagram representing the IP02 model

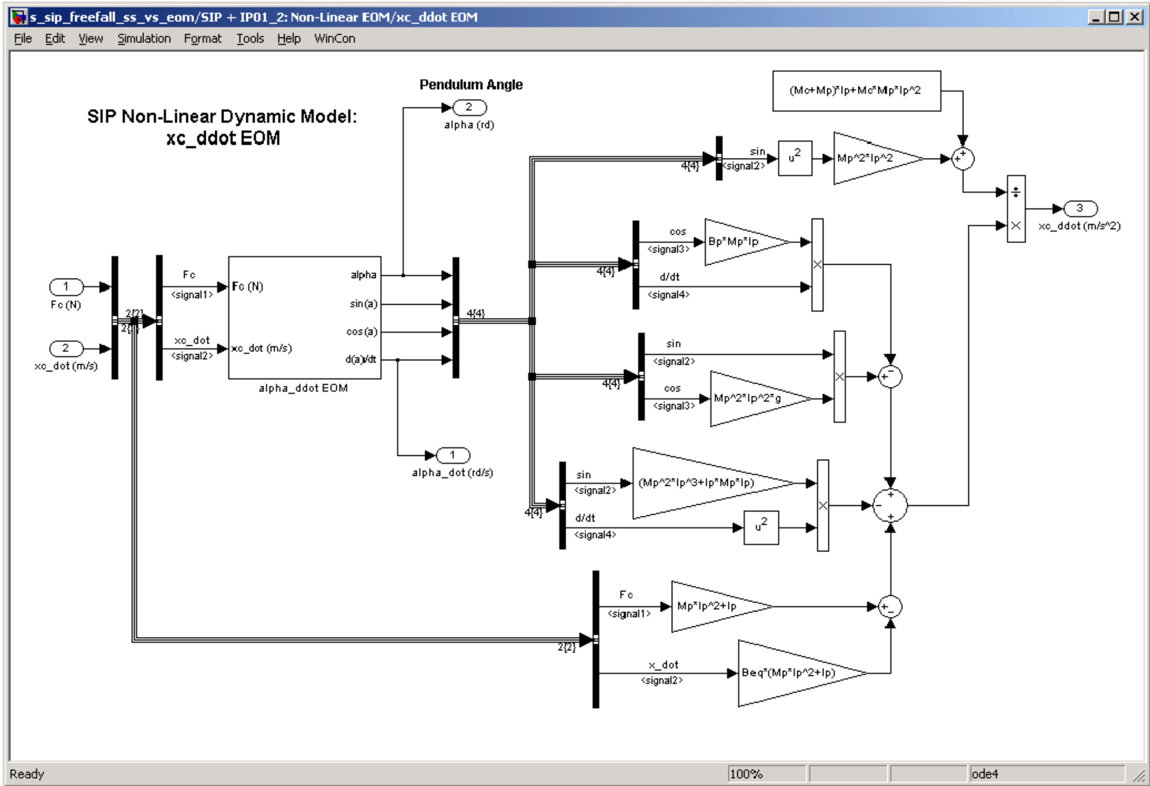


Figure 6 –Simulink sub-system diagram representing the first EOM for $d^2(x_c(t))/dt^2$

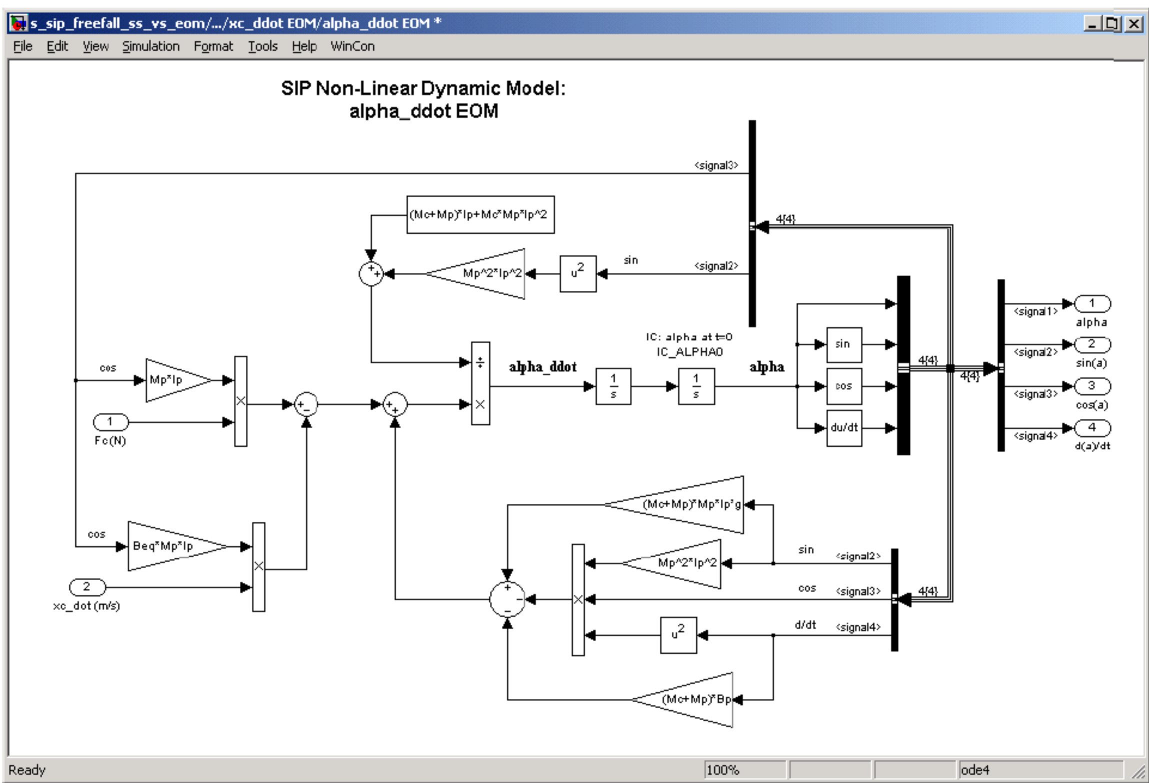


Figure 7 – Simulink sub-system diagram representing the second EOM for $d^2(\alpha(t))/dt^2$

Figure 6 above represents the first EOM for the cart position, given in Eq. (4) and has the form:

$$\frac{\partial^2}{\partial t^2} x_c = \left(\frac{\partial^2}{\partial t^2} x_c \right) (x_c, \alpha, F_c)$$

while Figure 7 represents the second EOM for the pendulum angle, given in Eq. (5) and having the form

$$\frac{\partial^2}{\partial t^2} \alpha = \left(\frac{\partial^2}{\partial t^2} \alpha \right) (x_c, \alpha, F_c)$$

Of particular interest in Figure 7, it should be noted that the initial condition on α, α_0, is contained inside the Simulink integration block located between the time derivative of α and α. That initial angle is referred in the Matlab workspace as the variable *IC_ALPHA0*.

By carrying out block diagram reduction, you can check that the two EOM that were discussed in the Preliminary Work correspond to the Simulink representations shown above.

Even though you are not obliged to do block reduction to re-obtain EOMs for your preliminary work, becoming familiar with realization of complicated non-linear equations in Simulink, such as EOMs in this experiment, will most likely be beneficial in your future studies.

7.2.4 Simulation of the Inverted Pendulum Free Fall

7.2.4.1 Objectives

- To simulate with Simulink the two inverted pendulum open-loop models previously discussed, namely the state-space linearized model and the two original EOM as expressed in Equations (4) and (5).
- To assess the operating range of the linearized, i.e., state-space, model. Therefore, assert the validity conditions of the linear approximation.

7.2.4.2 Experimental Procedure

Please follow the steps described below:

Step1. If you have not done so yet, you can start-up Matlab now and open the Simulink diagram titled `s_sip_freefall_ss_vs_eom.mdl`.

Step2. Before beginning the simulation, you must run the Matlab script called `setup_lab_ip01_2_sip.m`. This file initializes all the SIP and IP02 model parameters and user-defined configuration variables needed and used by the Simulink diagram.

Step3. Initialize at the Matlab prompt *IC_ALPHA0* to +0.1radian. Set the *Stop Time:* in the Simulink `Simulation parameters`... (Ctrl+E) to 3 seconds. You can now `Start` (Ctrl+T) the simulation of your diagram.

Step4. After the simulation run, open the two Scopes titled `Cart Position (mm)` and `Pend Angle (deg)`. What do you observe? Sketch the responses of the two models on these two scopes in the space provided. Use separate plot line-styles to separate the two traces from each other.

Is there a good match between the state-space and the EOM responses? Over the first half second of the simulation? Over the 3-second simulation range?
Hint: In fact, the pendulum actual behavior would first be like a free fall about its pivot then followed by a free swing around the "hanging down" equilibrium position.

Step5. Based on your observations, what would you suggest as a valid operating range for the linearized model? Explain your answer.

7.3 Simulation and Design of a Linear Quadratic Regulator (LQR)

7.3.1 Objectives

- To use the obtained SIP-plus-IP02 state-space representation to design the Linear Quadratic Regulator (LQR) type of optimal state-feedback controller.
- To tune on-the-fly and iteratively the LQR by *simulating* the closed-loop system in real-time with QuaRC.
- 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. Before you begin, you must run the Matlab script called `setup_lab_ip01_2_sip.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 `setup_lab_ip01_2_sip.m` file (re-)initializes all the SIP and IP02 model parameters and user-defined configuration variables needed and used by the Simulink diagrams. Lastly, it also calculates the state-space matrices, **A** and **B**, corresponding to the SIP-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 reviewed in the Preliminary Work. It is reminded that the LQR design is based on the plant's linearized state-space model. Finally, you should also check that the variable *IC_ALPHA0* has been set back to zero in the Matlab workspace (from its value during the previous simulation).

Step 2. Open the Simulink diagram titled `s_sip_lqr.mdl`. You should obtain a diagram similar to the one shown in Figure 8, below. Take some time to familiarize yourself with this model: it will be used for the LQR tuning simulation. For a plant simulation valid over the full angular range, the SIP-plus-IP02 model is implemented without linear approximation through its equations of motion given in Eqs. (4) and (5) as previously discussed. This model is represented in `s_sip_lqr.mdl` by the subsystem block titled `SIP + IP01_2: Non-Linear EOM`. You should also check that the signal generator block properties are properly set to output a square wave signal, of amplitude 1 and of frequency 0.1 Hz. As a remark, it should be noted that the reference input should be small enough in magnitude so that our system remains in the region where our linearization is valid. It can also be noticed in `s_sip_lqr.mdl` that the reference signal needs to be scaled in order to accommodate for the feedback vector, located in the feedback loop. By definition, it is reminded that the LQR feedback vector, named *K*, has four elements, corresponding to the four system's states defined in the Preliminary Work of this experiment.

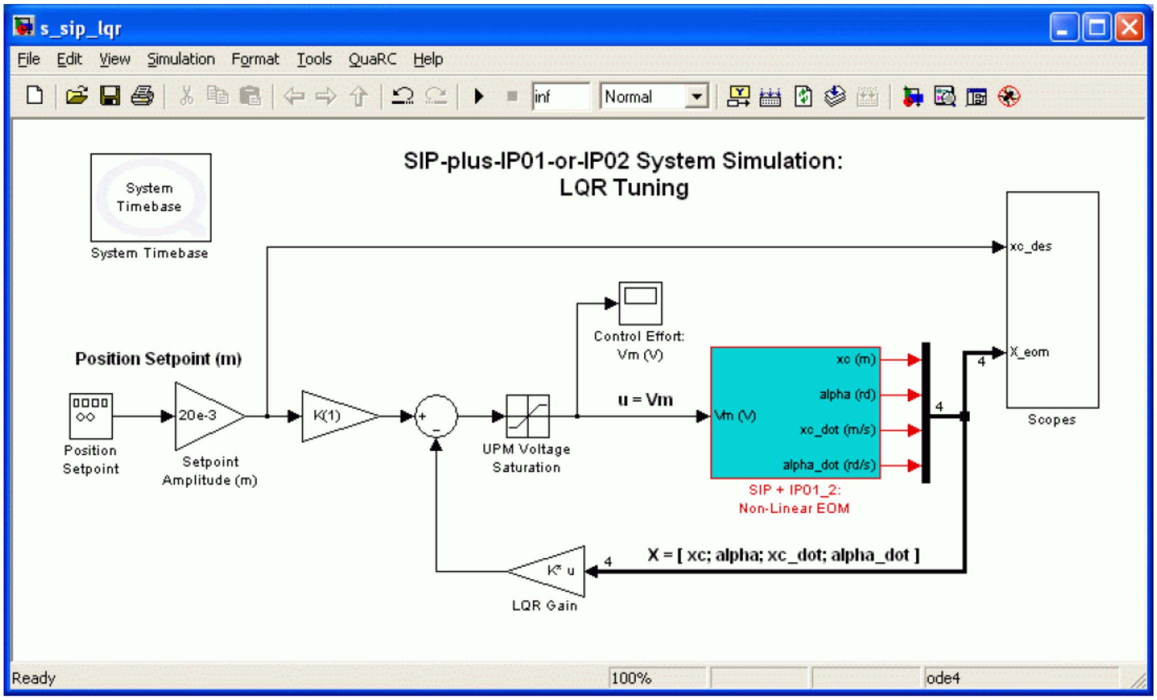


Figure 8 – Simulink model for the LQR Tuning simulation

Step 3. The `System Timebase` block is from the QuaRC library, and it enables simulations to be ran in real-time. In the Simulink model menu bar, click on `QuaRC | Start` to begin running the simulation.

Step 4. In order to observe the system's responses from the real-time simulation, open the three following scopes: `xc (mm)`, `alpha (deg)`, and `Control Effort: Vm (V)`. You should now be able to monitor on-the-fly the simulated cart position as it tracks your pre-defined reference signal, the simulated pendulum angle, as well as the corresponding control effort. However, at this stage, the responses obtained should all be zero since the feedback gain has been initialized to zero in Step 1.

Step 5. You are now ready to start the on-line tuning of your LQR which would give us an optimal state feedback vector *K*. First, the Matlab script `setup_lab_ip01_2_sip.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_sip.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. This is identical to the dialog box you have used in Experiment #5 before.

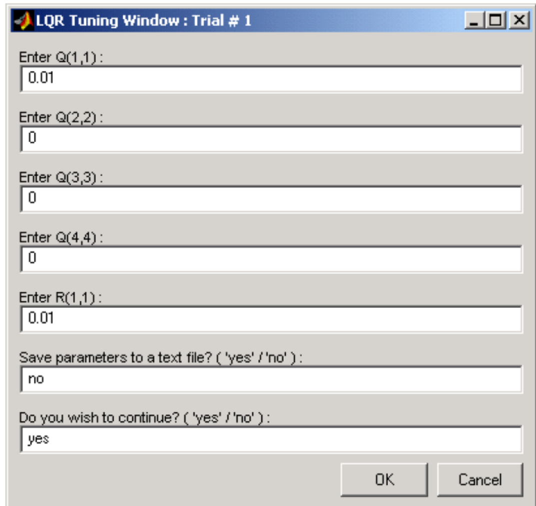


Figure 9 – On-Line LQR Tuning dialog box

The dialog box pictured in Figure 9 allows the user to enter the diagonal elements of the LQR weighting matrices **Q** and **R**. In this laboratory, **Q** is defined to be a pure diagonal matrix of 4-by-4 size (since the system has four states). Also, because we are studying a single input system, **R** is a scalar (i.e., a 1-by-1 matrix), of strictly positive value. It is reminded that, by design, the LQR attempts to return the system to a zero-state vector by minimizing the cost function defined by the **Q** and **R** matrices. 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 function `'lqr'` function 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. At his 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 6. Keep the initial, untuned, values for **Q** and **R** as is 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 7. 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 state component, and the amount of control effort supplied by the input (i.e., the energy spent).
The objective of this simulation is not really to make you find the perfect LQR tuning for the SIP-plus-IP02 system. This will be done during the actual hardware implementation part of this laboratory. It is more to make you infer, feel, and comprehend the basic principles at work during LQR tuning.

It is also reminded that the LQR design objectives are given in the Preliminary Work. The primal goal is, of course, to keep the pendulum balanced in the inverted position.

Now, you should decrease and/or increase the values of *Q(1,1)*, *Q(2,2)*, 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. Leave *Q(3,3)* and *Q(4,4)* equal to zero. Experiment with values included, for example, between 0.01 and 10 for *Q(1,1)* and *Q(2,2)* and between 0.0001 and 1 for *R(1,1)*. To keep a linear behaviour, ensure that the motor voltage V_m never goes into saturation. Also, the pendulum angle should stay within the validity range of the linearized state-space model, as determined in the inverted pendulum free fall simulation. Try to figure out trends on how our three tuning parameters *Q(1,1)*, *Q(2,2)*, and *R(1,1)*, affect the simulated cart position, pendulum angle, and control effort spent. Provide plots as needed to support your conjunctures.

Hint #1: *Q(1,1)* can be seen as a position error gain, *Q(2,2)* as an angle error gain, and *R(1,1)* as a control effort factor.
Hint #2: You can also refer to the definition of the LQR cost function to be minimized.

In the given table, fill in the trials that you find significant to draw conclusions about suitable **Q** and **R** matrices.

Step 8. Now, finalize the tuning procedure in order to meet the desired closed-loop specifications as stated in 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 is 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 the Preliminary Work.
***Hint #1*:** Use the Matlab function `'eig(.)'` to determine the eigenvalues of the closed-loop state-space matrix.
Hint #2*:** The closed-loop state-space matrix can be expressed as: **A**-**BK**.

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

Step9. 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 SIP-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 SIP-plus-IP02 plant.
- To tune on-the-fly and iteratively the LQR observing the actual system response.
- To run the LQR closed-loop system simulation in parallel and simultaneously, 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 SIP-plus-IP02 plant and checked the type of responses obtained from the system's two outputs (i.e., cart position and pendulum angle), you are now ready to implement your designed controller in real-time and observe its effect on your actual inverted pendulum system. To achieve this, please follow the steps described below:

Step 1. Open the Simulink model file `q_sip_lqr_ip02.mdl`. Ask the Lab Assigned 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 10.

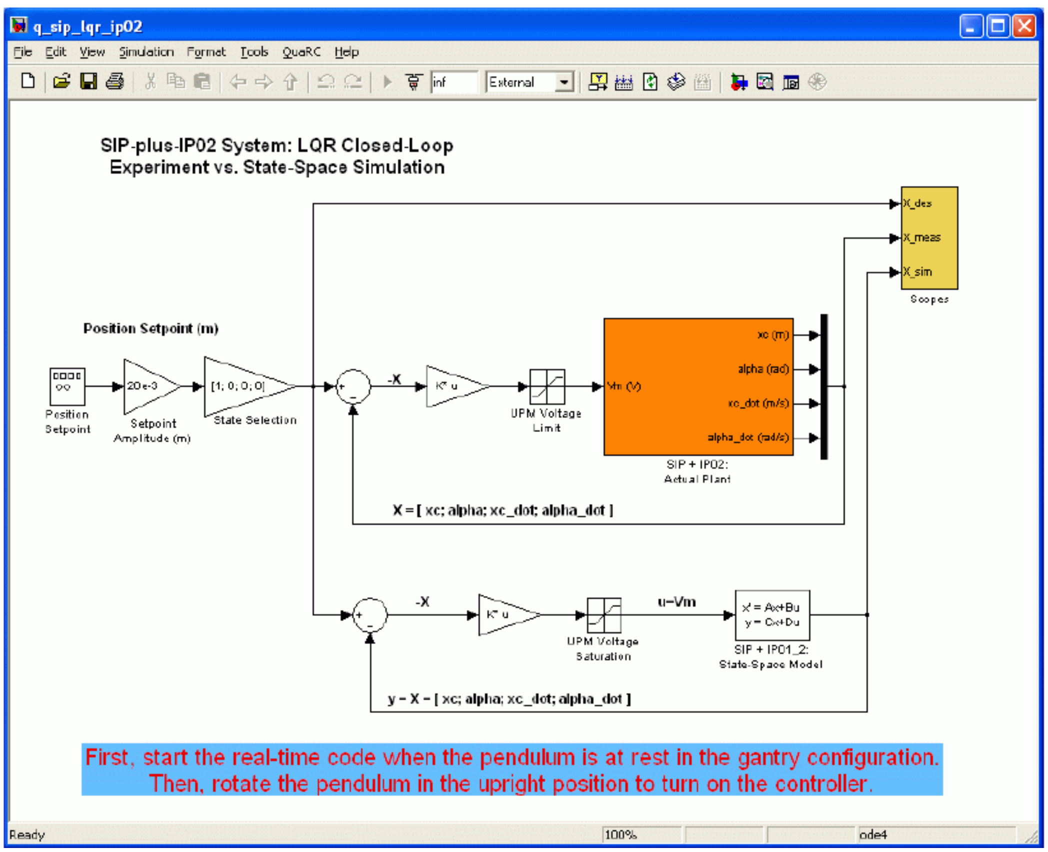


Figure 10 – Simulink model for the Real-Time Implementation of the LQR Closed-Loop

The model has two parallel and independent control loops: one runs a pure simulation of the LQR-plus-SIP-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 inverted pendulum mounted on 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 reference signal generated for the cart position is a square wave of amplitude 20 mm and frequency 0.2 Hz. Also, 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 had 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))`

Make sure that you applied the initial values of Q and R given in the newly opened “Tuning GUI”. The K matrix you found in the simulations can turn out to be large and can damage the plastic motor gear on IP02 system. In other words, you will start your LQR tuning from scratch. You can slowly converge to your Q and R values found in simulation but take your time and progress slowly.

Step 4. **Configure DAQ**: Double-click on the HIL Initialize block inside the `SPG+IP02 Actual Plant\IP02` subsystem and ensure it is configured for the DAQ device (Q8-Q2) that is installed in your system. See Reference [6] for more information on configuring the HIL Initialize block.

Step 5. 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 6. After successful compilation and download, you should be able to run in real-time your actual system. However, before doing so, **manually move your IP02 cart 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. Additionally, ensure that the pendulum is now mounted and free to rotate over its full rotational range (±360º if configured on an IP02) anywhere within the cart's full linear range of motion.** Before starting the real-time controller, also follow the starting procedure for the inverted pendulum, as described in the following Step.

Step 7. **Inverted Pendulum Starting Procedure.** One important consideration is that encoders take their initial position as zero when the real-time code is started. In our linearized model, it is reminded that the zero pendulum angle corresponds to a perfectly upright position.

IP02 Starting Procedure: This starting procedure finds the exact vertical position from the pendulum hanging straight down at rest, in front of the linear cart. Therefore, this method can only be practiced with an **IP02** cart, where the pendulum can be in downright configuration. The starting procedure **consists first of letting the pendulum come to perfect rest in the downright configuration**. The real-time code can be started so that the exact ±α-radian angle can now be computed from relative encoder values. Note that since the downright position is used to initialize (zero) the encoder values, your Simulink model includes some blocks to remove the angle offset at the upright position (where α is by definition zero).

To start the control system, click on the `QuaRC | Start` from the Simulink model menu bar. The model also includes a "trigger" feature that only activates the controller when α-angle is brought (manually) to a sufficiently small neighborhood of zero as required by the designed linear controller.

Hence, manually rotate the pendulum to its upright position. The LQR, initially turned off, now automatically becomes enabled and in effect. Your cart position should now be tracking the desired position signal while maintaining the inverted pendulum balanced.

Step 8. From the *Scopes* subsystem block, open the two sinks `Cart Position (mm)` and `Pend Angle (deg)`. 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 subsystem `SIP + IP02: Actual Plant\IP02\`. On the `Cart Position (mm)` scope, you should now be able to monitor on-line, as the cart moves, the actual cart position as it tracks your pre-defined reference input signal and compare it with the simulation result produced by the SIP-plus-IP02 state-space model. On the `Pend Angle (deg)` scope, you should be able to monitor on-line, as the cart moves, the actual inverted pendulum angle and its fluctuation about the vertical axis at any given time. You can also compare it, on the same scope, to the simulated angle resulting from the state-space model.

Step 9. What are your observations at this point? Does your actual LQR closed-loop implementation meet the desired design specifications? If it does not, then you should finely tune the LQR weighting matrices, **Q** and **R**, in order for the actual inverted-pendulum-linear-cart system to meet 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_sip.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_sip.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 10. Iterate your manual LQR tuning as many times as necessary so that your actual system's performances meet the desired design specifications. If you are still unable to achieve 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.

Step 11. Once your results are in agreement with the closed-loop requirements, they should look similar to those displayed in Figure 11.

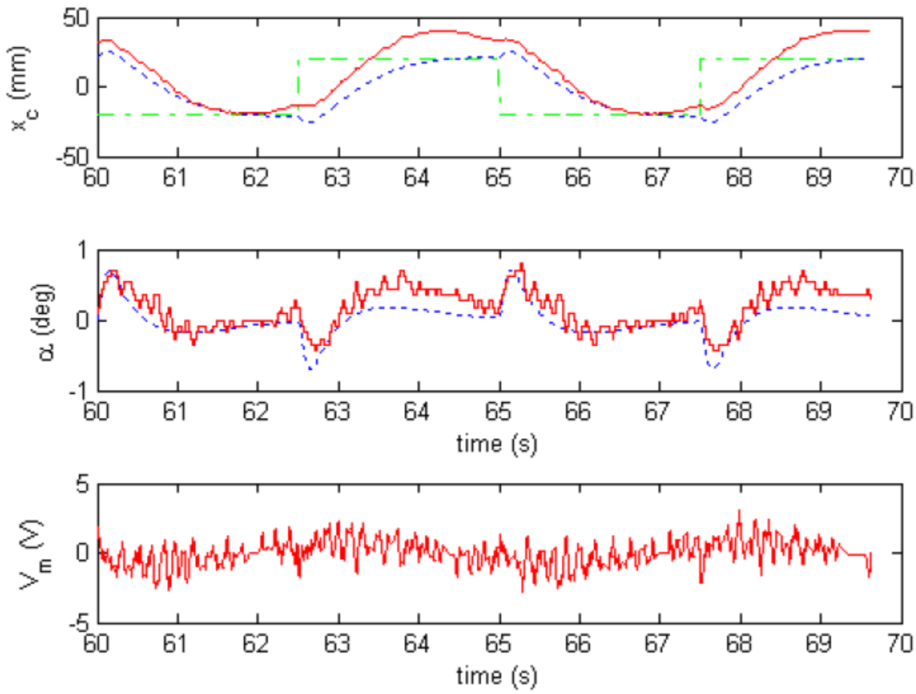


Figure 11 – Actual (solid red) and Simulated (blue dotted line) Cart Position Response in top plot and Pendulum Angle in middle plot. Bottom plot shows input motor voltage

Do you notice a steady-state error on your actual cart position response? If so, find some of the possible reasons. Can you think of any improvement on the closed-loop scheme in order to reduce, or eliminate, that steady-state error?

Inside the given area, include in your lab report the plots that you obtained, as equivalent to Figure 11, above. Ensure to properly document all your results and observations before moving on the next section.

Step 12. Remember that there is no such thing as a perfect model. Specifically discuss in your lab report the following points:

- How does your actual inverted pendulum angle 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.
- How much different the actual LQR feedback gain was from the one you determined through simulation (which was based on a theoretical and ideal plant model)?

7.5 Assessment of the System's Disturbance Rejection

This part of the experiment is provided to give you some basic insights on the regulation problem through a few disturbance rejection considerations. In this version of the problem, the system is asked to maintain an equilibrium point where the reference signal is kept constant (may be assumed to be at zero).

7.5.1 Objectives

- To observe and investigate the disturbance response of the stabilized inverted-pendulum-linear-cart system in response to a tap to the pendulum (an external disturbance).
- To study the LQR effectiveness in maintaining the inverted pendulum in its upright equilibrium state while simultaneously recovering the cart position to the origin.

7.5.2 Experimental Procedure

Follow the experimental procedure described below:

Step 1. Start and run your inverted-pendulum-linear-cart system around the mid-stroke position, i.e., near the centre of the track. Use the same LQR closed-loop as the one previously developed. However, this time, set the cart position setpoint amplitude to zero, so that the LQR regulates both cart position and pendulum angle around zero. This is the regulation configuration (i.e., there is no tracking).

Step 2. Once the system has stabilized, gently tap the inverted pendulum, but not more than (plus or minus) 4 degrees from its upright equilibrium position. You may exceed this limit to see the angle where the model and hence the designed controller fails. But be gentle while experimenting! Visually observe the response of the linear cart and its effect on the pendulum angle using the scopes. Also, open the `V Command (V)` sink to plot the corresponding motor input voltage V_m.

Step 3. How do the three responses behave, as a result to the tap, in the recovery of the inverted pendulum? How does the cart catch the pendulum? Describe the system's response from both your visual observations and the obtained response plots.

Step 4. How does the controller fail? Why is the controller not failing with diverging controller effort on the motor? Examine the Simulink diagram of the actual plant and controller (diving into sub-blocks if necessary) in order to give a plausible explanation. You may wish to plot motor voltage `V Command (V)` to help with your reasoning.

Appendix A. Nomenclature

Table A.1, below, provides a complete listing of the symbols and notations used in the IP02 mathematical modeling, as presented in this laboratory. The numerical values of the system parameters can be found in Reference [2].

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-ElectroMotive-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_{c1}	IP01 Cart Mass (Cart Alone)	Mc1
M_{c2}	IP02 Cart Mass (Cart Alone)	Mc2
M_w	IP02 Cart Weight Mass	Mw
M	IP01 or IP02 Cart Mass, including the Possible Extra Weight	
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 IP01 and IP02 Model Nomenclature

Table A.2, below, provides a complete listing of the symbols and notations used in the mathematical modelling of the single inverted pendulum, as presented in this laboratory. The numerical values of the pendulum system parameters can be found in Reference [3].

Symbol	***Description***	***Matlab / Simulink Notation***
α	Pendulum Angle from the Upright Position	alpha
$\frac{\partial}{\partial t}\alpha$	Pendulum Angular Velocity	alpha_dot
α_0	Initial Pendulum Angle (at t=0)	IC_ALPHA0
M_p	Pendulum Mass (with T-fitting)	Mp
L_p	Pendulum Full Length (from Pivot to Tip)	Lp
l_p	Pendulum Length from Pivot to Center of Gravity	lp
I_p	Pendulum Moment of Inertia	Ip
x_p	Absolute x-coordinate of the Pendulum Centre of Gravity	
y_p	Absolute y-coordinate of the Pendulum Centre of Gravity	

Table A.2 Single Inverted Pendulum 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 SIP-plus-IP02 System	A, B, C, D
X	State Vector	X
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	

Table A.3 LQR Nomenclature

Appendix B. Derivation of Non-Linear Equations of Motion (EOM)

This Appendix derives the general dynamic equations of the Single Inverted Pendulum (SIP) module mounted on the IP02 linear cart. The Lagrange's method is used to obtain the dynamic model of the system. In this approach, the single input to the system is considered to be F_c.

To carry out the Lagrange's approach, the Lagrangian of the system needs to be determined. This is done through the calculation of the system's total potential and kinetic energies.

According to the frame definition, illustrated in Figure 2, the absolute Cartesian coordinates of the pendulum's center of gravity are characterized by:

$$x_p(t) = x_c(t) - l_p \sin(\alpha(t)) \quad \text{and} \quad y_p(t) = l_p \cos(\alpha(t)) \qquad \text{[B.1]}$$

Let us first calculate the system's total potential energy V_T. The potential energy is a system is the amount of that system, or system element, has due to some kind of work being, or having been, done to it. It is usually caused by its vertical displacement from normality (gravitational potential energy) or by a spring-related sort of displacement (elastic potential energy).

Here, there is no elastic potential energy in the system. The system's potential is only due to gravity. The cart linear motion is horizontal, and as such, never has vertical displacement. Therefore, the total potential energy if fully expressed by the pendulum's gravitational potential energy, as characterized below:

$$V_T = M_p\, g\, l_p \cos(\alpha(t)) \qquad \text{[B.2]}$$

It can be seen from Equation [B.2] that the total potential energy can be expressed in terms of the generalized coordinate(s) alone.

Let us now determine the system's total kinetic energy T_T. The kinetic energy measures the amount of energy in a system due to its motion. Here the total kinetic energy is the sum of the translational and rotational kinetic energies arising from both the cart (since the cart's direction of translation is orthogonal to that of the rotor's rotation) and its mounted gantry pendulum (since the SPG's translation is orthogonal to its rotation).

First, the translational kinetic energy of the motorized cart, T_{ct}, is expressed as follows:

$$T_{ct} = \frac{1}{2} M \left(\frac{d}{dt} x_c(t) \right)^2 \qquad \text{[B.3]}$$

Second, the rotational kinetic energy due to the cart's DC motor, T_{cr}, can be characterized by:

$$T_{cr} = \frac{1}{2} \frac{J_m K_g^2 \left(\frac{d}{dt} x_c(t) \right)^2}{r_{mp}^2} \qquad \text{[B.4]}$$

Therefore, as a result of Equations [B.3] and [B.4], T_c, the cart's total kinetic energy, can be written as shown below:

$$T_c = \frac{1}{2} M_c \left(\frac{d}{dt} x_c(t) \right)^2 \quad \text{where} \quad Mc = M + \frac{J_m K_g^2}{r_{mp}^2} \qquad \text{[B.5]}$$

Mass of the single pendulum is assumed to be concentrated at its Center of Gravity (COG). Therefore, the pendulum's translational kinetic energy, T_{pt}, can be expressed as a function of its center of gravity's linear velocity, as shown by the following equation:

$$T_{pt} = \frac{1}{2} M_p \left(\sqrt{\left(\frac{d}{dt} x_p(t)\right)^2 + \left(\frac{d}{dt} y_p(t)\right)^2} \right)^2 \qquad [B.6]$$

where, the linear velocity's x-coordinate of the pendulum's center of gravity is determined by:

$$\frac{d}{dt} x_p(t) = \left(\frac{d}{dt} x_c(t)\right) - l_p \cos(\alpha(t)) \left(\frac{d}{dt} \alpha(t)\right) \qquad [B.7]$$

and the linear velocity's x-coordinate of the pendulum's center of gravity is expressed by:

$$\frac{d}{dt} y_p(t) = -l_p \sin(\alpha(t)) \left(\frac{d}{dt} \alpha(t)\right) \qquad [B.8]$$

In addition, the pendulum's rotational kinetic energy, T_{pr}, can be characterized by:

$$T_{pr} = \frac{1}{2} I_p \left(\frac{d}{dt} \alpha(t)\right)^2 \qquad [B.9]$$

Thus, the total kinetic energy of the system is the sum of the four individual kinetic energies, as previously characterized in Equations [B.5], [B.6], [B.7], [B.8] and [B.9]. By expanding, collecting terms, and rearranging, the system's total kinetic energy, T_T, results to be such as:

$$\begin{aligned} T_T = {} & \frac{1}{2}(M_c + M_p)\left(\frac{d}{dt} x_c(t)\right)^2 - M_p l_p \cos(\alpha(t)) \left(\frac{d}{dt} \alpha(t)\right)\left(\frac{d}{dt} x_c(t)\right) \\ & + \frac{1}{2}(I_p + M_p l_p^2)\left(\frac{d}{dt} \alpha(t)\right)^2 \end{aligned} \qquad [B.10]$$

It can be seen from Equation [B.10] that the total kinetic energy can be expressed in terms of both the generalized coordinates and of their first-time derivatives.

Let us now consider the Lagrange's equation for our system. By definition, the two Lagrange's equations, resulting from the previously defined two generalized coordinated, x_c and α, have the following formal formulations:

$$\frac{\partial}{\partial t \, \partial\left(\frac{d}{dt} x_c(t)\right)} L - \frac{\partial}{\partial x_c(t)} L = Q_{x_c} \qquad [B.11]$$

and:

$$\frac{\partial}{\partial t \, \partial\left(\frac{d}{dt} \alpha(t)\right)} L - \frac{\partial}{\partial \alpha(t)} L = Q_{\alpha} \qquad [B.12]$$

The first term of Equation [B.11] means taking derivative of L with respect to $\left(\frac{d}{dt} x_c(t)\right)$ and then taking derivative of the result with respect to time (t).

In Equations [B.11] and [B.12], L is called the Lagrangian and is defined to be such that:

$$L = T_T - V_T \qquad [B.13]$$

In Equation [B.11], Q_{xc} is the generalized force applied on the generalized coordinate x_c. Likewise in Equation [B.12], Q_α as the generalized force applied on the generalized coordinate α. Our system's generalized forces can be defined as follows:

$$Q_{x_c}(t) = F_c(t) - B_{eq}\left(\frac{d}{dt}x_c(t)\right) \quad \text{and} \quad Q_\alpha(t) = -B_p\left(\frac{d}{dt}\alpha(t)\right) \qquad \text{[B.14]}$$

It should be noted that the (nonlinear) Coulomb friction applied to the linear cart has been neglected. Moreover, the force on the linear cart due to the pendulum's action has also been neglected in the presently developed model.

Calculating Equation [B.11] results in a more explicit expression for the first Lagrange's equation, such that:

$$\begin{aligned}&(M_c + M_p)\left(\frac{d^2}{dt^2}x_c(t)\right) - M_p\, l_p\cos(\alpha(t))\left(\frac{d^2}{dt^2}\alpha(t)\right) + M_p\, l_p\sin(\alpha(t))\left(\frac{d}{dt}\alpha(t)\right)^2 = \\ &\qquad F_c - B_{eq}\left(\frac{d}{dt}x_c(t)\right)\end{aligned} \qquad \text{[B.15]}$$

Likewise, calculating Equation [B.12] also results in a more explicit form for the second Lagrange's equation, as shown below:

$$\begin{aligned}&-M_p\, l_p\cos(\alpha(t))\left(\frac{d^2}{dt^2}x_c(t)\right) + (I_p + M_p\, l_p^2)\left(\frac{d^2}{dt^2}\alpha(t)\right) - M_p\, g\, l_p\sin(\alpha(t)) = \\ &\qquad -B_p\left(\frac{d}{dt}\alpha(t)\right)\end{aligned} \qquad \text{[B.16]}$$

Finally, solving the set of the two Lagrange's equations, as previously expressed in Equations [B.15] and [B.16], for the second-order time derivative of the two Lagrangian coordinates results in the following two non-linear equations:

$$\frac{d^2}{dt^2}x_c(t) = \frac{\left(\begin{aligned}&-(I_p + M_p\, l_p^2)B_{eq}\left(\frac{d}{dt}x_c(t)\right) - (M_p^2\, l_p^3 + I_p\, M_p\, l_p)\sin(\alpha(t))\left(\frac{d}{dt}\alpha(t)\right)^2 \\ &- M_p\, l_p\cos(\alpha(t))B_p\left(\frac{d}{dt}\alpha(t)\right) + (I_p + M_p\, l_p^2)F_c + M_p^2\, l_p^2\, g\cos(\alpha(t))\sin(\alpha(t))\end{aligned}\right)}{\left((M_c + M_p)I_p + M_c\, M_p\, l_p^2 + M_p^2\, l_p^2\sin(\alpha(t))^2\right)}$$

[B.17]

$$\frac{d^2}{dt^2}\alpha(t) = \frac{\left(\begin{aligned}&(M_c + M_p)M_p\, g\, l_p\sin(\alpha(t)) - (M_c + M_p)B_p\left(\frac{d}{dt}\alpha(t)\right) + F_c\, M_p\, l_p\cos(\alpha(t)) \\ &- M_p^2\, l_p^2\sin(\alpha(t))\cos(\alpha(t))\left(\frac{d}{dt}\alpha(t)\right)^2 - M_p\, l_p\cos(\alpha(t))B_{eq}\left(\frac{d}{dt}x_c(t)\right)\end{aligned}\right)}{\left((M_c + M_p)I_p + M_c\, M_p\, l_p^2 + M_p^2\, l_p^2\sin(\alpha(t))^2\right)}$$

[B.18]

Equations [B.17] and [B.18] represent the Equations Of Motion (EOM) of the system.

Equations [B.17] and [B.18] use force applied on cart, F_c. However, IP02 block in the laboratory takes motor voltage, V_m, as the input. But one can easily find the transform between F_c and V_m using the block diagram of IP02 setup. It can be expressed as:

$$F_c(t) = -\frac{\eta_g K_g^2 \eta_m K_t K_m \left(\frac{d}{dt} x_c(t)\right)}{R_m r_{mp}^2} + \frac{\eta_g K_g \eta_m K_t V_m(t)}{R_m r_{mp}} \quad \text{[B.19]}$$

Equations [B.17] and [B.18] should be reconfigured by integrating Equation [B.19]. So, the final form of the EOMs can be written as:

$$\frac{d^2}{dt^2} x_c(t) = \frac{\left(\begin{array}{l} -\left(I_p + M_p l_p^2\right) B_{eq} \left(\frac{d}{dt} x_c(t)\right) - \left(M_p^2 l_p^3 + I_p M_p l_p\right) \sin(\alpha(t)) \left(\frac{d}{dt} \alpha(t)\right)^2 \ldots \\ + \left(I_p + M_p l_p^2\right) \left(-\frac{\eta_g K_g^2 \eta_m K_t K_m \left(\frac{d}{dt} x_c(t)\right)}{R_m r_{mp}^2} + \frac{\eta_g K_g \eta_m K_t V_m(t)}{R_m r_{mp}} \right) \ldots \\ - M_p l_p \cos(\alpha(t)) B_p \left(\frac{d}{dt} \alpha(t)\right) + M_p^2 l_p^2 g \cos(\alpha(t)) \sin(\alpha(t)) \end{array} \right)}{\left(\left(M_c + M_p\right) I_p + M_c M_p l_p^2 + M_p^2 l_p^2 \sin(\alpha(t))^2\right)}$$

[B.20]

$$\frac{d^2}{dt^2} \alpha(t) = \frac{\left(\begin{array}{l} \left(M_c + M_p\right) M_p g l_p \sin(\alpha(t)) - \left(M_c + M_p\right) B_p \left(\frac{d}{dt} \alpha(t)\right) \ldots \\ + \left(-\frac{\eta_g K_g^2 \eta_m K_t K_m \left(\frac{d}{dt} x_c(t)\right)}{R_m r_{mp}^2} + \frac{\eta_g K_g \eta_m K_t V_m(t)}{R_m r_{mp}} \right) M_p l_p \cos(\alpha(t)) \ldots \\ - M_p^2 l_p^2 \sin(\alpha(t)) \cos(\alpha(t)) \left(\frac{d}{dt} \alpha(t)\right)^2 - M_p l_p \cos(\alpha(t)) B_{eq} \left(\frac{d}{dt} x_c(t)\right) \end{array} \right)}{\left(\left(M_c + M_p\right) I_p + M_c M_p l_p^2 + M_p^2 l_p^2 \sin(\alpha(t))^2\right)}$$

[B.21]

As a final remark on derivations, the single suspended pendulum's moment of inertia about its center of gravity is characterized by:

$$I_p = \frac{1}{12} M_p L_p^2 \quad \text{[B.22]}$$

8 Report

Names, Student IDs, and Signatures of Group Students:

1) ____

2) ____

Date of Experiment:

Name of Lab Assistant:

7.2 Assessment of the State Space Model Operating Range

7.2.4 Simulation of the Inverted Pendulum Free Fall

Step 4:

Sketch of Cart Position (mm)

Sketch of Pendulum Angle (deg)

Comments:

Step 5:

7.3 Simulation and Design of a Linear Quadratic Regulator (LQR)

Step 7:

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

Further comments and observations.

Step 8:

a)

$|\alpha|_{max}$ =

t_r =

PO (%) for cart position =

V_m (min) =

V_m (max) =

e_{ss} for cart position =

b)

K = [ ]

Q = diag([ ])

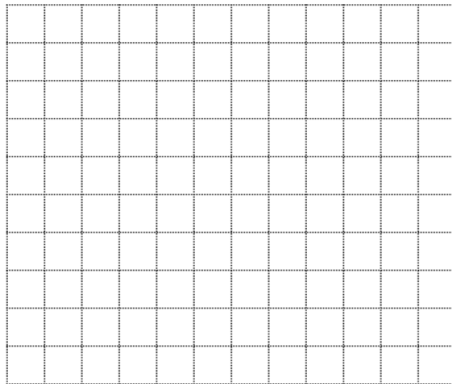
R = [ ]

c)

d)

x_c(mm)	α(degrees)

Vm (V)



7.4 Real-Time Implementation of the LQR

Step 9:

Step 11:

Step 12:

Sketch of Cart Position x_c(mm)
Sketch of Pendulum Angle α(degrees)

Sketch of motor voltage V_m(V)

Step 13:

7.5 Assessment of the System's Disturbance Rejection

Step 2:

Sketch of Cart Position x_c(mm) for disturbance rejection
Sketch of Pendulum Angle α(degrees) for disturbance rejection
Sketch of motor voltage V_m(V) for disturbance rejection

Step 3:

Step 4:

9 Knowledge Test

1. Why we started inverted pendulum when pointing downward? In the Simulink setup, control was initialized when pendulum was on the stable equilibrium point. What would happen if the control was set to start it from the inverted position? Does the inverted starting have any advantages or disadvantages?

2. The physical system used in the 4^{th} experiment, Single Pendulum Gantry, and in this experiment, Single Inverted Pendulum, were the same. But the derived equations of motion are not the same. What is the difference? Can we also use the EOMs derived in the 4^{th} experiment? If we can, what we should do to obtain ABCD matrices correctly from the EOMs of the 4^{th} experiment?

3. What is the aim of LQR? Explain the LQR design, also give basic formulation.

4. Compare the derivation process of ABCD matrices for this experiment and the 5th experiment (Single Linear Flexible Joint). Comment on all differences.