The Control of an Inverted Pendulum

AAE 364L

This experiment is devoted to the inverted pendulum. Clearly, the inverted pendulum will fall without any control. We will design a controller to balance the pendulum upright.

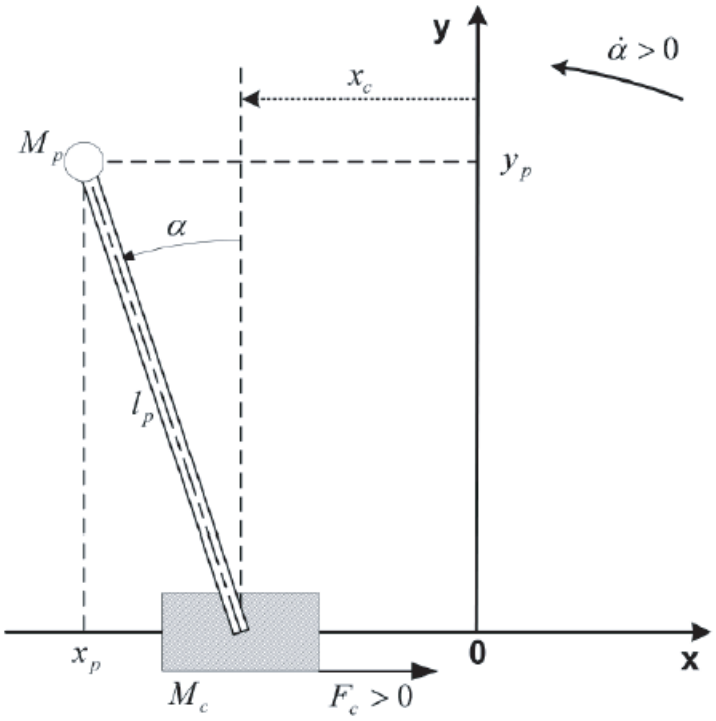

Figure 1: The inverted pendulum.

This experiment consists of a cart with mass M_c on a one dimensional track with a pendulum attached to the cart. The pendulum starts in the upward position. The cart is driven by a force from a servo motor. The position of the cart is denoted by $x_c(t)$, and the Voltage to the servo motor is denoted by $v(t)$. The angle between the pendulum and its vertical or upright position is denoted by α; see Figure 1. The nonlinear equations of motion

are given by

$$\begin{aligned}
&\left((M_c+M_p)I_p + M_cM_pl_p^2 + M_p^2l_p^2\sin(\alpha)^2\right)\ddot{x}_c + \left(I_p + M_pl_p^2\right)B_{eq}\dot{x}_c \\
&= \quad -M_pl_pB_p\cos(\alpha)\dot{\alpha} - \left(M_p^2l_p^3 + I_pM_pl_p\right)\sin(\alpha)\dot{\alpha}^2 + \left(I_p + M_pl_p^2\right)F_c + M_p^2l_p^2g\cos(\alpha)\sin(\alpha);
\end{aligned}$$

$$\begin{aligned}
&\left((M_c+M_p)I_p + M_cM_pl_p^2 + M_p^2l_p^2\sin(\alpha)^2\right)\ddot{\alpha} + (M_c+M_p)B_p\dot{\alpha} \\
&= \quad (M_c+M_p)M_pgl_p\sin(\alpha) - M_pl_p\cos(\alpha)B_{eq}\dot{x}_c - M_p^2l_p^2\sin(\alpha)\cos(\alpha)\dot{\alpha}^2 + M_pl_p\cos(\alpha)F_c;
\end{aligned}$$

$$F_c = \frac{\eta_gK_g\eta_mK_t\left(vr_{mp} - K_gK_m\dot{x}_c\right)}{R_mr_{mp}^2}. \tag{0.1}$$

Here F_c is the force on the cart. The notation and values are given in the table in the Appendix. These values are given also given in the MATLAB file "setup_lab_ip01_2_sip.m" posted on the Web page for the course. So you do not have to calculate any of these values for the Lab. The linearized equations of motion about $\alpha = 0$ are determined by

$$\begin{aligned}
\ddot{x}_c &= \frac{M_p^2l_p^2g\alpha - (I_p + M_pl_p^2)B_{eq}\dot{x}_c - M_pl_pB_p\dot{\alpha} + (I_p + M_pl_p^2)F_c}{(M_c+M_p)I_p + M_cM_pl_p^2} \\
\ddot{\alpha} &= \frac{(M_c+M_p)M_pgl_p\alpha - (M_c+M_p)B_p\dot{\alpha} - M_pl_pB_{eq}\dot{x}_c + M_pl_pF_c}{(M_c+M_p)I_p + M_cM_pl_p^2} \\
F_c &= \frac{\eta_gK_g\eta_mK_t\left(vr_{mp} - K_gK_m\dot{x}_c\right)}{R_mr_{mp}^2}.
\end{aligned} \tag{0.2}$$

Now let us convert the linear equations to a state space model of the form

$$\dot{x} = Ax + Bv.$$

Here A is a 4×4 matrix, B is a column vector of length 4 and the input v is the voltage to the motor. To convert the system in (0.2) to state space we will use the following state variables:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_c \\ \alpha \\ \dot{x}_c \\ \dot{\alpha} \end{bmatrix}. \tag{0.3}$$

By consulting the values in the table in the Appendix, we computed A and B, that is,

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1.5216 & -11.6513 & -0.0049 \\ 0 & 26.1093 & -26.8458 & -0.0841 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 \\ 0 \\ 1.5304 \\ 3.5261 \end{bmatrix}. \tag{0.4}$$

It is emphasized that this A and B in (0.4) is computed for the IP02 cart with the 0.37 kg weight attached and the **long** pendulum (if you are using the short pendulum, these values may differ). The matrices A and B in (0.4) are computed in the MATLAB setup

file "setup_lab_ip01_2_sip.m" posted on the course Web page (Blackboard). So do not manually type A and B in MATLAB. Because the inverted pendulum is unstable and the state equation $\dot{x} = Ax + Bv$ is the linear approximation for the inverted pendulum, the matrix A is unstable. In fact, the eigenvalues for this A matrix are given by

$$\text{eig}(A) = \{0, 4.8231, -12.0133, -4.5452\}. \tag{0.5}$$

As expected, A is unstable.

Let us compute the transfer function from the voltage v to the output x_c. Since $x_1 = x_c$, the corresponding output matrix C is given by

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}. \tag{0.6}$$

By using $ss2tf$ in MATLAB, we see that the transfer function from the voltage v to the position x_c of the cart is given by

$$\frac{X_c(s)}{V(s)} = C(sI - A)^{-1}B = \frac{1.53s^2 + 0.1114s - 34.59}{s^4 + 11.74s^3 - 25.26s^2 - 263.4s}. \tag{0.7}$$

Notice that this is a fourth order system. Because the denominator contains terms with negative coefficients, it follows that the transfer function $\frac{X_c(s)}{V(s)}$ is unstable. In fact, the poles of $\frac{X_c(s)}{V(s)}$ are the eigenvalues of A. In other words, the poles of $\frac{X_c(s)}{V(s)}$ are given by $\text{eig}(A) = \{0, 4.8231, -12.0133, -4.5452\}$. Therefore $\frac{X_c(s)}{V(s)}$ is unstable.

Now let us compute the transfer function from the voltage v to the angle α. Since $x_2 = \alpha$, the corresponding output matrix C is given by

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}. \tag{0.8}$$

By using $ss2tf$ in MATLAB, we see that the transfer function from the voltage v to the angle α is determined by

$$\frac{\alpha(s)}{V(s)} = C(sI - A)^{-1}B = \frac{3.526s}{s^3 + 11.74s^2 - 25.26s - 263.4}. \tag{0.9}$$

Finally, it is noted that this is an unstable third order system.

The displacement of the end of the pendulum along the track is given by $x_e = x_c + L_p \sin(\alpha)$ where L_p is the length of the pendulum; see Figure 1. (Notice that l_p is the distance to the center of mass of the pendulum and L_p is the distance to the end of the pendulum.) For small angles $x_e \approx x_c + L_p\alpha$. Since $x_1 = x_c$, $x_2 = \alpha$ and $L_p = 0.6413$ m, the output matrix C corresponding to x_e is given by

$$C = \begin{bmatrix} 1 & 0.6413 & 0 & 0 \end{bmatrix}. \tag{0.10}$$

By using $ss2tf$ in MATLAB we see that the transfer function from the voltage v to the end of the pendulum x_e is determined by

$$\frac{X_e(s)}{V(s)} = C(sI - A)^{-1}B = \frac{3.792s^2 + 0.1114s - 34.59}{s^4 + 11.74s^3 - 25.26s^2 - 263.4s}. \tag{0.11}$$

Finally, it is noted that this is an unstable fourth order system.

1 The linear quadratic regulator

You can use pole placement to design a feedback controller for the inverted pendulum. One disadvantage of the pole placement procedure is that placing the poles at a desired location can lead to large gains. In this section, we will present the linear quadratic regulator (LQR) problem, and show how one can use the LQR for control design. It turns out that in many instances, the LQR method is superior to the pole placement method. Finally, you can choose either pole placement or LQR for your control design of the inverted pendulum. *In fact, you can use any control method that you want as long as it works.* However, we believe you will discover that the LQR method will work very nicely for the inverted pendulum.

Now let us present the linear quadratic regulator (LQR) method. The proof behind the LQR is given in A&AE 564. Here we will just use MATLAB to design a LQR controller. To set up the LQR problem, consider a controllable state space system of the form

$$\dot{x} = Ax + Bv. \tag{1.1}$$

Here A is a $n \times n$ matrix and B is a column vector of length n and the state

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

is a vector of length n. Let $R > 0$ and Q be a diagonal matrix of the form

$$Q = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & q_n \end{bmatrix}, \tag{1.2}$$

where $q_j \geq 0$ for all j. Now consider the optimization problem

$$\begin{aligned} &\min \int_0^\infty \left(x^* Q x + R v^2\right) dt = \min \int_0^\infty \left(\sum_{j=1}^n q_j x_j^2 + R v^2 \right) dt \\ &\text{subject to} \quad \dot{x} = Ax + Bv. \end{aligned} \tag{1.3}$$

(The conjugate transpose of a vector z is denoted by z^*.) The control engineer chooses the weights $\{q_j\}_1^n$ and $R > 0$. It is emphasized that the weight R must be strictly positive. The optimal control v or solution to this optimization problem is unique and given by $v = -Kx$ where K is a state gain matrix, that is, K is a row matrix of length n. The MATLAB command to compute K is given by

$$K = \text{lqr}(A, B, Q, R). \tag{1.4}$$

In this case, the closed loop system corresponding to the optimal control $v = -Kx$ is given by

$$\dot{x} = (A - BK)x.$$

Finally, it is noted that $A - BK$ is stable.

The state weights $\{q_j\}_1^n$ are chosen to emphasize the response of certain states, while the control weight R is chosen to select how much control effort v is used to solve the optimization problem in (1.3). For example, if q_1 and q_3 are large, then the weights q_1 and q_3 are placing a large penalty on the states x_1 and x_3, that is, if x_1 or x_3 is large, then the large weights q_1 and q_3 will amplify the effect of x_1 or x_3 in the optimization problem. Since the optimization problem is trying to minimize $\int_0^\infty \left(\sum_{j=0}^n q_j x_j^2 + Rv^2\right) dt$, the optimal control v must force the states x_1 and x_3 to be small. So if q_1 and q_3 are large, then the control designer is trying to make the states x_1 and x_3 small and places less emphasis on how the other states respond.

If the control weight R is large relative to $\{q_j\}_1^n$, then the weight R is placing a large penalty on the control effort v, that is, if the control v is large, then the large weight R relative to $\{q_j\}_1^n$ will amplify the effect of the control v in the optimization problem. Since the optimization problem is trying to minimize $\int_0^\infty \left(\sum_{j=0}^n q_j x_j^2 + Rv^2\right) dt$, the optimal control v must be small in order to solve the optimization problem in (1.3). In other words, if R is large relative to $\{q_j\}_1^n$, then control effort is expensive. So for large R, the gain K should be small and the response might be slow. On the other hand, if R is small relative to $\max\{q_j\}_1^n$, then there is virtually no penalty on the control effort v in the minimization problem (1.3), and the optimal control v can be large. In other words, if R is small relative to $\max\{q_j\}_1^n$, then control is cheap. So for small R, the gain K might be large and the response should be faster. Finally, it is noted that a large gain may cause problems. Recall that our system has saturation. So a large gain may lead to instability. Furthermore, a large gain has a faster response. This can lead to *chattering* in the system, that is, the system moves so fast that the gears chatter back and forth even when the system is trying to remain at rest. Summing up this analysis we obtain:

- If q_j is large, then the control will try to make the state x_j small relative to the other states.
- If q_j is small, then the control will not place much emphasis on x_j and x_j could be large relative to the other states.
- If R is large relative to $\{q_j\}_1^n$, then control is expensive and the response may be slow.
- If R is small relative to $\max\{q_j\}_1^n$, then control is cheap and the response should be faster.

Now consider our state space for in the inverted pendulum in (0.4). The manufacturer suggests starting with $R = 3 \times 10^{-4}$ and

$$Q = \begin{bmatrix} 0.75 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} q_1 & 0 & 0 & 0 \\ 0 & q_2 & 0 & 0 \\ 0 & 0 & q_3 & 0 \\ 0 & 0 & 0 & q_4 \end{bmatrix}. \tag{1.5}$$

Recall that $x_1 = x_c$ is the position of the cart, $x_2 = \alpha$ is the angle of the pendulum from the vertical position, while $x_3 = \dot{x}_c$ and $x_4 = \dot{\alpha}$. Since R is small the control engineer is

allowing the optimal control to use a large control effort. So the control should be fast. The 0.75 weight on $x_1 = x_c$ says that the control engineer wants to keep the position of the cart from moving too much. Notice that 4 is the largest element in Q. By choosing the weight 4 on $x_2 = \alpha$ the designer is placing a larger emphasis on the angle α, over any other state. In other words, the designer is saying that making α small is more important than making any other state small. Since $q_3 = 0$ and $q_4 = 0$, this design is not trying to place any particular emphasis on the velocity $\dot{x}_c$ of the cart or the angular velocity $\dot{\alpha}$ of the pendulum. Finally, it is noted that the gain K corresponding to Q in (1.5) and $R = 3 \times 10^{-4}$ is given by

$$\begin{bmatrix} -50 & 180.16 & -50.26 & 28.49 \end{bmatrix} = \text{lqr}(A, B, \text{diag}(\begin{bmatrix} \frac{3}{4} & 4 & 0 & 0 \end{bmatrix}), \frac{3}{10^4}). \tag{1.6}$$

There are all kinds of LQR controllers which will work. For example, $R = 1$ and

$$Q = \begin{bmatrix} 20 & 0 & 0 & 0 \\ 0 & 20 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}. \tag{1.7}$$

In this case, the gain K corresponding to Q in (1.7) and $R = 1$ is given by

$$\begin{bmatrix} -4.47 & 54.69 & -18.15 & 11.3 \end{bmatrix} = \text{lqr}(A, B, \text{diag}(\begin{bmatrix} 20 & 20 & 10 & 3 \end{bmatrix}), 1). \tag{1.8}$$

In practice, the designer will simulate and test many different R and Q matrices in the LQR design, before arriving at the final controller.

2 Part (i): The inverted pendulum

In this section we will use the pole placement and the LQR method to design controllers to balance the pendulum upright. Then we shall compare the system response due to the controllers we obtain via these design methods, both in simulation and experiment.

Consider the state space system given by

$$\dot{x} = Ax + Bv. \tag{2.1}$$

The input is the voltage v and the vector x is the state; see (0.3). Here A is the 4×4 matrix, B is a column vector of length 4, determined by the cart and pendulum; see (0.4). Your problem is to design a state feedback controller $K = \begin{bmatrix} K(1) & K(2) & K(3) & K(4) \end{bmatrix}$ which makes the pendulum stand up. *It is very important to note that in practice, you can use any control method that you like as long as it works.* The idea here is to introduce you to the LQR method to design a state feedback gain K.

To begin your control design, go to the course Web page (Blackboard) and load the following MATLAB files in your computer:

- setup_lab_ip01_2_sip.m
- setup_ip01_2_configuration.m
- setup_sp_configuration.m
- SSIP_ABCD_eqns.m
- d_gui_lqr_tuning.m
- d_ip01_2_sip_lqr.m
- s_sip_lqr.mdl

Run the MATLAB file "setup_lab_ip01_2_sip.m" and open the Simulink file "s_sip_lqr.mdl". This Simulink file is displayed in Figure 2. You can use this Simulink file to simulate your controller design. In the Simulink file set the amplitude of the position set point equal to zero; see Figure 2. Check to make sure that the setup program "setup_lab_ip01_2_sip.m" has specified the following variables in MATLAB:

(i) CART_TYPE = 'IP02'

(ii) IP02_LOAD_TYPE = 'WEIGHT'

(iii) PEND_TYPE = 'MEDIUM_12IN'

(iv) IC_ALPHA0 = 0.2.

This means that we are using the manufacturer's cart IP02, the *SHORT* pendulum, and we have added the 0.37 kg weight on the cart, and initial condition $\alpha(0) = 12$ degrees (about 0.2 radians). If (i) to (iii) does not hold go into the "setup_lab_ip01_2_sip.m" file and add or take off the comment % on the appropriate lines. Then run the file "setup_lab_ip01_2_sip.m" again. If IC_ALPHA0 $\neq 0.2$, then simply set IC_ALPHA0 = 0.2 in MATLAB before you run the Simulink file. All the variables that you need to run the simulation are now loaded in the computer. MATLAB contains all the values in the Table in the Appendix.

Now using the pole placement method, find a state feedback gain K to balance the pendulum upright. Recall that the command used in MATLAB is

$$K = \text{place}(A, B, \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 \end{bmatrix}). \tag{2.2}$$

Recall that to obtain a stable system, the real parts of $\{\lambda_j\}_1^4$ must all be negative. Moreover, since the system is real, if we choose any of the poles to be complex then its complex conjugate must also be chosen. In our case, you may choose all poles to be negative real, or λ_1, λ_2 are negative real and λ_3 is complex with negative real part with $\lambda_4 = \bar{\lambda}_3$, or λ_1, λ_3 complex with negative real parts whereas $\lambda_2 = \bar{\lambda}_1, \lambda_4 = \bar{\lambda}_3$. Finally, recall that the complex part of any pole corresponds to the sinusoidal frequency of the response. Note that the natural frequency of the pendulum is about $5\, rad/s$. So we may want some poles with frequencies much faster than $5\, rad/s$.

Now using the LQR method, design a state feedback gain K to balance the pendulum upright. The LQR command in MATLAB is

$$K = \text{lqr}(A, B, \text{diag}(\begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}), R). \tag{2.3}$$

The poles of this feedback system are the eigenvalues of $A - BK$.

You can test the state gain K you obtain from both methods by typing K in MATLAB and then running the Simulink file "s_sip_lqr.mdl". This Simulink file will see if your controllers can balance the pendulum upright when the pendulum is initially leaning 0.2 radians off center. Bring your best state feedback gains K you obtain from both design methods along with the corresponding plots of the angle and position to the Lab with you.

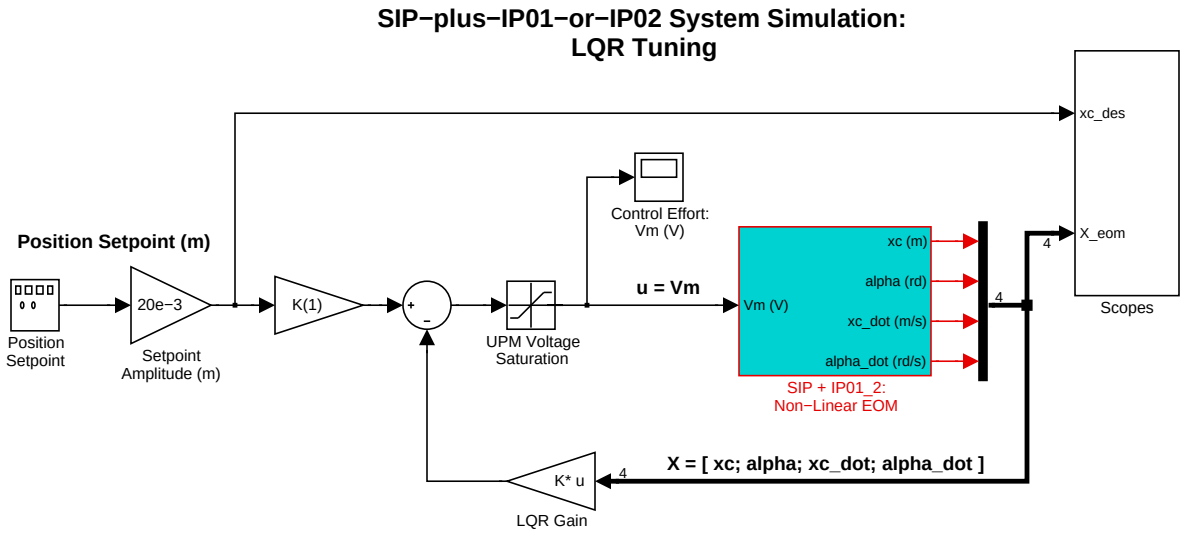

Figure 2: The closed loop nonlinear model.

2.1 Pre-Lab to balance the pendulum. Due at the beginning of the lab experiment. You will not be allowed run the lab experiment with out a complete pre-lab.

(i) Hand in the values of the feedback gains K (from pole placement *and* LQR) that you obtained using Simulink to balance the ***short*** inverted pendulum, **as well as the computed A and B matrices**. You will be given no credit for using the state feedback gain K in (1.6) or (1.8).

(ii) Hand in the plots of the poles of the feedback system with pole placement and LQR method. Put them on separate plots, but with the same scale, same range, same position of origin.

(iii) Hand in the plots for the angle α for the both gains K that you used in Simulink on the same graph. Hand in the plots for the position x_c for the both gains K that you used in Simulink on the same graph.

2.2 The Lab steps to balance the pendulum

(i) Open MATLAB and change the directory to $Desktop : \backslash AAE364L \backslash section\# \backslash labinvpend$.

(ii) Open the setup file `setup_ip02_sip.m`. Make sure that PEND_TYPE = 'MEDIUM_12IN'. Run the setup file.

(iii) In the MATLAB command window, type: `X_MAX = 0.6; X_MIN = -0.6;`.

(iv) Enter your best gain K you obtained from the pole placement method in MATLAB, that is,

$$K = \begin{bmatrix} K(1) & K(2) & K(3) & K(4) \end{bmatrix}.$$

(v) Open `labpinv1` in Simulink. Select `Tools - External Mode Control Panel - Signal & Triggering`, and change the duration to 15,000.

(vi) In the Simulink window, set simulation time to 29 seconds.

(vii) **In order to properly record your data, open the position scope and select Parameters. Deselect the *"Limit to 5000 data points"* check-box. Select *Save data to workspace*. Enter an appropriate variable name, in this case *x_c*. Make sure the format is "Structure with time".** Repeat for angle scope (*alpha*).

(viii) Put the cart in the middle of the track, with the pendulum in the gantry or downward position. The computer thinks that this position is the angle π, so make sure that the pendulum is not swinging.

(ix) Click `Quarc - Build`. Select `Simulation - Connect to Target`, and `Quarc - Start`. Now slowly move the pendulum to the upright position. As soon as the angle reaches zero, the controller will be activated. In case your controller does not work, be prepared to hit stop. In other words, if the pendulum falls or the cart runs off the track hit stop. (There is a built in safety switch, and this switch will stop the experiment if the cart runs off the track.)

(x) Have the TA tap the pendulum to see if the controller can balance the pendulum and return the cart to the zero position.

(xi) Save the position and angle in MATLAB (`File - Save - Save as Mat file`).

(xii) Hold the cart still (TA will tell you how to do it). Gently push the tip of the pendulum so that the angle is about 0.2 rad. You will feel the force from the controller trying to push the cart against you. Let go of the cart and the pendulum and wait until the system returns to zero state.

(xiii) After the system response returns to zero, click Stop.

(xiv) Save the position and angle in MATLAB (`File - Save - Save as Mat file`).

(xv) Enter your best gain K you obtained from the LQR method in MATLAB, that is,

$$K = \begin{bmatrix} K(1) & K(2) & K(3) & K(4) \end{bmatrix}.$$

Repeat steps (viii) to (xiv).

2.3 In your lab report include the following under Part (i):

(a) Hand in the plots of the poles of the feedback system with pole placement and LQR method. Put them on separate plots, but with the same scale, same range, same position of origin.

(b) Clearly state the values of these poles in a Table, NOT in text.

(c) Clearly state the Matrix Q and R you choose for your LQR design in equation, NOT in text.

- Initial condition $x_c = 0, \alpha = 0.2, \dot{x}_c = 0, \dot{\alpha} = 0$

(d) Hand in the plots of position you obtain from the simulation using pole placement and LQR, from the experiment using pole placement, and LQR on the SAME graph (you should have four lines). Make sure the initial condition x_c here is zero. If your experimental results do not have zero initial x_c, scale them to zero (Subtract nonzero initial condition off).

(e) Hand in the plots of angle you obtain from the simulation using pole placement and LQR, from the experiment using pole placement, and LQR on the SAME graph (you should have four lines). Your initial condition from the simulation should be 0.2 rad. However, the initial condition from the experiment will not be exactly 0.2 rad.

- Initial condition $x_c = 0, \alpha = 0, \dot{x}_c = 0, \dot{\alpha} = \dot{\alpha}$

  For the following parts, the size of $\dot{\alpha}$ can not be measured and must be determined from the experiment data. So use the simulation to determined the approximate value of $\dot{\alpha}$. This can be done by first changing the parameter IC_ALPHA0 to zero. Then in the Simulink model, go into the block "SIP+IP01_ 2: Non-linear EOM", then go into the block "xc_ ddot EOM", then go into the block "alpha_ ddot EOM", finally go into the integrator block "1/s" behind the words "alpha_ ddot" and change the "initial condition". Simulate various $\dot{\alpha}$ so that your simulation result resembles the experimental result.

(f) Hand in the plots of position you obtain from the simulation using pole placement, from the experiment using pole placement, on the SAME graph (you should have two lines). Make sure the initial condition here is zero. If your experimental results do not have zero initial condition, scale them to zero (Subtract nonzero initial condition off). Do the same with the simulation and experiment results using LQR.

(g) Hand in the plots of angle you obtain from the simulation using pole placement, from the experiment using pole placement on the SAME graph (you should have two lines). Your initial condition from both the simulation and the experiment should be 0 rad. Do the same with the simulation and experiment results using LQR.

3 Part (ii): Moving the balanced pendulum

In this part we will use a state space integral controller to move the inverted pendulum from one point on the track to another without letting the pendulum fall. In this section we will be using the cart IP02 with the 0.37 kg weight and the *SHORT* pendulum. For these specifications run the MATLAB file "setup_lab_ip01_2_sip.m". Make sure that the following parameters are specified in MATLAB:

(i) CART_TYPE = 'IP02'

(ii) IP02_LOAD_TYPE = 'WEIGHT'

(iii) PEND_TYPE = 'MEDIUM_12IN'

(iv) IC_ALPHA0 = 0.

Finally, it is noted that in this part we have set IC_ALPHA0 = 0.

The idea behind our design is to incorporate an integral controller in our LQR design. (You can use pole placement. However, LQR will be more effective.) To this end, consider the four dimensional state variable system in (0.4). Now let us define a new state variable

$$x_5 = \int_0^t (u(\sigma) - x_1(\sigma))\, d\sigma \quad \text{or equivalently} \quad \dot{x}_5 = u - x_1. \tag{3.1}$$

Here u is the new input. Recall that x_1 is the position of the cart. In our problem $u = u_0$ will be a constant. So x_5 will integrate $u_0 - x_c$. Using $\dot{x}_5 = u - x_1$, the state variable system in (0.4) now becomes

$$\dot{x} = A_i x + B_i v + Du. \tag{3.2}$$

Recall that the new state matrix A_i and B_i are given by

$$A_i = \begin{bmatrix} & A & & \begin{bmatrix}0\\0\\0\\0\end{bmatrix} \\ \begin{bmatrix}-1 & 0 & 0 & 0\end{bmatrix} & & & 0 \end{bmatrix} \quad \text{and} \quad B_i = \begin{bmatrix} B \\ 0 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ 0 \end{bmatrix}$$

$$x = \begin{bmatrix} x_c \\ \alpha \\ \dot{x}_c \\ \dot{\alpha} \\ \int (u - x_c) dt \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Using the matrices A and B in (0.4) one can easily construct A_i, B_i and D in MATLAB. Recall that you already have A and B in MATLAB from your setup file. In MATLAB the matrices A_i and B_i are computed by

$$\begin{aligned} A_i &= [A, \text{zeros}(4,1); -1, 0, 0, 0, 0] \\ B_i &= [B; 0]\,. \end{aligned}$$

Finally, it is noted that A_i is a 5×5 matrix and B_i is a column vector of length 5.

It turns out that the pair $\{A_i, B_i\}$ is controllable. So one can use the LQR (or place) command in MATLAB to design a state feedback controller K. Using $v = -Kx$ in the new state variable system in (3.2), we arrive at

$$\dot{x} = (A_i - B_i K)\, x + Du. \tag{3.3}$$

Now choose u to be the step $u(t) = u_0$ for all $t \geq 0$. Because all the eigenvalues of $A_i - B_i K$ are in the open left hand plane, the state space system in (3.3) will move to steady state,

that is, $\dot{x} = 0$ in steady state. In particular, $0 = \dot{x}_5 = u_0 - x_1$. In other words, in steady state $x_c = x_1 = u_0$, and the cart will move to the position u_0 on the track.

Now we would like to design a controller to balance the pendulum and move the cart to any position from -0.2 to 0.2 meters on the track. To accomplish this choose the weights $\{q_i\}_1^5$ and R that you think will work. Notice that in this case, there are five state weights $\{q_i\}_1^5$. The weights q_1, q_2, q_3, q_4 and q_5 respectively correspond to the position x_c of the cart, the angle α of the pendulum, the velocity $\dot{x}_c$ of the cart, the angular velocity $\dot{\alpha}$ of the pendulum, and the integral $\int(u_0 - x_c)dt$. Then in MATLAB type

$$K = \text{lqr}(A_i, B_i, \text{diag}(\begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 \end{bmatrix}), R). \tag{3.4}$$

To test your control design, you can use the Simulink file "aae364pinv2.mdl" on the Web site (Blackboard), displayed in Figure 3, or simply rebuild the Simulink file "s_sip_lqr.mdl" to the one specified in Figure 3. To test your design set the step block to 1 and slider gain to any specified position u_0 in $[-0.2, 0.2]$. Then choose the weights Q and R that you think will work and run the LQR command in MATLAB. Finally, run the Simulink file in Figure 3 using various positions on the slider gain. Keep changing the weights Q and R until you find the state feedback gain which achieves your desired response. Now set the slider gain to 0.2 and in the step block change step time to 5, initial value to -1, and final value to 1. Simulate the system and keep the result. Bring your best gain matrix K and this simulation result to the Lab.

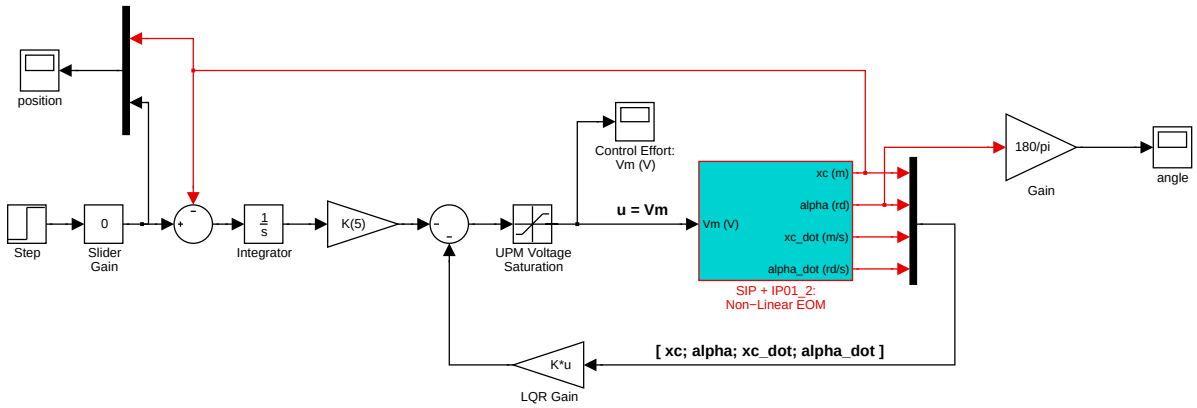

Figure 3: The closed loop nonlinear model.

3.1 Pre-Lab for the integral controller. Due at the beginning of the lab experiment. You will not be allowed to run the lab experiment with out a complete pre-lab.

(i) Hand in your best values for the state gain K that you computed from Simulink, as well as your A_i and B_i matrices.

(ii) Hand in the plots of position and angle from the simulation.

3.2 The Lab steps to incorporation of integral control

(i) In the same directory, open the setup file `setup_ip02_sip.m` and verify that PEND_TYPE = 'MEDIUM_12IN'. Run the setup file.

(ii) In the MATLAB command window, type: `X_MAX = 0.6; X_MIN = -0.6;` .

(iii) Enter your best gain K that you computed from Simulink, that is,

$$K = \begin{bmatrix} K(1) & K(2) & K(3) & K(4) & K(5) \end{bmatrix}.$$

(iv) In the same directory, open `labpinv2` in Simulink. Select `Tools - External Mode Control Panel - Signal & Triggering`, and change the duration to 30,000.

(v) In the Simulink window, set simulation time to 59 seconds.

(vi) **To properly record your data**, repeat Step (vi) from Section 2.2 for the angle scope (alpha) and the 'x and command' scope (xac).

(vii) Put the cart in the middle of the track, with the pendulum in the gantry or downward position. The computer thinks that this position is the angle π. So make sure that the pendulum is not swinging when you click the start button.

(viii) You will now test your response using a Pulse Generator. Replace the Step and Slider Gain blocks with a Pulse Generator block. Set the Amplitude to 0.2, the Period to 10 seconds, the Pulse Width to 50%, and the Phase Delay to 10 seconds.

(ix) Click `Quarc - Build`. Select `Simulation - Connect to Target`, and `Quarc - Start`. Now slowly move the pendulum to the upright position. As soon as the angle reaches zero, the controller will be activated. In case your controller does not work, be prepared to hit stop. In other words, if the pendulum falls or the cart runs off the track click Stop.

(x) Save the cart position and pendulum angle in MATLAB (`File - Save - Save as Mat file`).

3.3 In your lab report include the following under Part (iii):

(a) Hand in the plots of the poles of the feedback system.

(b) Clearly state the values of these poles in a Table, NOT in text.

(c) Clearly state the Matrix Q and R you choose for your LQR design in equation, NOT in text.

(d) Hand in the plots of position you obtain from the simulation and the experiment on the SAME graph.

(e) Hand in the plots of angle you obtain from the simulation and the experiment on the SAME graph.

4 Appendix: The notation for the pendulum and cart

Symbol	Description	Value	Unit
R_m	motor armature resistance	2.6	Ω
L_m	motor armature inductance	0.18	mH
K_t	motor torque constant	0.00767	$N.m/A$
η_m	motor efficiency	100%	%
K_m	back-electromotive-force(EMF) constant	0.00767	$V.s/rad$
J_m	rotor moment of inertia	3.9×10^{-7}	$kg.m^2$
K_g	planetary gearbox ratio	3.71	
η_g	planetary gearbox efficiency	100%	%
M_{c2}	cart mass	0.57	kg
M_w	cart weight mass	0.37	kg
M_c	total cart weight mass including motor inertia	1.0731	kg
B_{eq}	viscous damping at motor pinion	5.4000	$N.s/m$
L_t	track length	0.990	m
T_c	cart travel	0.814	m
P_r	rack pitch	1.664×10^{-3}	$m/tooth$
r_{mp}	motor pinion radius	6.35×10^{-3}	m
N_{mp}	motor pinion number of teeth	24	
r_{pp}	position pinion radius	0.01482975	m
N_{pp}	position pinion number of teeth	56	
K_{EP}	cart encoder resolution	2.275×10^{-5}	$m/count$
M_p	long pendulum mass with T-fitting	0.230	kg
M_{pm}	medium pendulum mass with T-fitting	0.127	kg
L_p	long pendulum length from pivot to tip	0.6413	m
L_{pm}	medium pendulum length from pivot to tip	0.3365	m
l_p	long pendulum length: pivot to center of mass	0.3302	m
l_{pm}	medium pendulum length: pivot to center of mass	0.1778	m
J_p	long pendulum moment of inertia ↻ center of mass	7.88×10^{-3}	$kg.m^2$
J_{pm}	medium pendulum moment of inertia ↻ center of mass	1.20×10^{-3}	$kg.m^2$
B_p	viscous damping at pendulum axis	0.0024	$N.m.s/rad$
g	gravitational constant	9.81	m/s^2