Junjie Gao
Yuxuan Ma
Joe Yu

ECE 486: Final Project Report

Sp 2023

TA: Yuju Chang

Section: Friday ABB

Introduction

In this project, our goal is to design a control system for a reaction wheel pendulum that allows it to stabilize at 180 degrees angle. We used two sensors which directly measure the angles θ_p and θ_r. To obtain the values of $d\theta/dt$ we used a approximation continuous integration: $\frac{s}{\tau s+1}$

We only have one actuator, which is a motor mounted at the end of the pendulum. There are two equilibrium positions: one at $\theta_p = 0$ degrees (stable), and one at $\theta_p = 180$ degrees (unstable). In order to design a system that allows the pendulum to stabilize at the unstable equilibrium position, we designed four different systems using Matlab and Simulink.

Mathematical Model

1. Derivation of differential equations from Lagrangian

To define the mathematical model, we define the following variables:

- m_p - mass of the pendulum and motor housing/stator
- m_r- mass of the rotor
- m - combined mass of the rotor and pendulum
- J_p- moment of inertia of the pendulum about its center of mass
- J_r- moment of inertia of the rotor about its center of mass
- l_p- distance from pivot to the center of mass of the pendulum
- l_r- distance from pivot to the center of mass of the rotor
- l - distance from pivot to the center of mass of pendulum and rotor
- k - torque constant of the motor
- i - input current to motor

Firstly, we compute the value of kinetic energy K and potential energy P in terms of θ_p *and* θ_r.

$$KE_{pendulum+rotor} = \frac{1}{2}J\omega_p{}^2 = \frac{1}{2}J\left(\frac{\partial\theta_p}{\partial t}\right)^2$$

$$PE_{pendulum+rotor} = mgh = mgl_p(1-\cos(\theta_p))$$

$$KE_{rotor} = \frac{1}{2}J_r\omega_r{}^2 = \frac{1}{2}J_r\left(\frac{\partial\theta_r}{\partial t}\right)^2$$

$$PE_{rotor} = 0$$

After calculating values of kinetic and potential energies, the Lagrange (q is θ and $\dot{q}$ is ω)

$$L(q_1, \ldots, q_n, \dot{q_1}, \ldots, \dot{q_n})$$

is then defined as the difference between the kinematics and potential energies. Therefore

$$L_{p+r} = KE_{p+r} - PE_{p+r} = \frac{1}{2}J\dot{q}_{p+r}{}^2 - mgl_p(1-\cos(q_{p+r}))$$

$$L_r = KE_r - PE_r = \frac{1}{2}J_r(\dot{q_r})^2 - 0$$

According to Lagrange's equation, it can be shown that the equations of motion all have the form

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q_k}}\right) - \frac{\partial L}{\partial q_k} = \tau_k$$

Where τ_k *represents the generalized force in the* q_k *direction*. Since the RWP includes the motor, the motor cannot produce a net torque on the RWP. Therefore, when it exerts a torque on the rotor it must exert an equal and opposite torque on the pendulum. Here we use the relation τ = ki for motor torque. In this case, the pendulum rotates in clockwise direction, so the torque should be negative.

The new Lagrange's Equation would be:

$$J\ddot{q}_{p+r} + mgl_p \sin(q_{p+r}) = -ki$$

Plugging in ${\omega_{np}}^2 = \frac{mgl}{J}$, we get:

$$\ddot{\theta}_p + {\omega_{np}}^2 \sin(\theta_p) = -\frac{k}{J} i$$

Similarly, for the rotor we get:

$$\ddot{\theta}_r = \frac{k}{J_r} i$$

After that, we take friction into consideration. Due to current feedback, the current is proportional to the control command u from the computer. The control variable used in the computer is scaled so that 10 units correspond to maximum current. Therefore, we can write the equations as

$$ki = k_u u, \qquad |u| \le 10$$

Assuming the friction is a function of the rotor speed $F(\omega_r)$, we can get the following.

$$\begin{cases} \ddot{\theta}_p + {\omega_{np}}^2 \sin(\theta_p) = -\frac{k_u}{J}(u + F(\dot{\theta}_r)) \\ \ddot{\theta}_r = \frac{k_u}{J_r}(u + F(\dot{\theta}_r)) \end{cases}$$

In order to get a satisfactory representation of the RWP, we clear up the clutter:

$$\begin{cases} \ddot{\theta}_p + a \sin(\theta_p) = -b_p(u + F(\dot{\theta}_r)) \\ \ddot{\theta}_r = b_r(u + F(\dot{\theta}_r)) \end{cases}$$

$$a = {\omega_{np}}^2 = \frac{mgl}{J}$$

$$b_p = \frac{k_u}{J}$$

$$b_r = \frac{k_u}{J_r}$$

2. Linearization into state space form

To stabilize the system to equilibrium point π, we define the delta states as

$$\delta\theta_p = \theta_p - \pi$$
$$\delta\theta_r = \theta_r$$

Using the techniques to convert the following system to state-space model:

$$\begin{cases} \ddot{\theta}_p + a\sin(\theta_p) = -b_p(u + F(\dot{\theta}_r)) \\ \ddot{\theta}_r = b_r(u + F(\dot{\theta}_r)) \end{cases}$$

$$\dot{x} \quad = \quad A \quad x \quad + \quad B\,u$$

$$\begin{bmatrix} \delta\dot{\theta}_p \\ \ddot{\theta}_p \\ \delta\dot{\theta}_r \\ \ddot{\theta}_r \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \omega_{np}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta\theta_p \\ \dot{\theta}_p \\ \delta\theta_r \\ \dot{\theta}_r \end{bmatrix} + \begin{bmatrix} 0 \\ -b_p \\ 0 \\ b_r \end{bmatrix} u$$

Full State Feedback Control with Friction Compensation

1. development of the PD control

At the beginning, we implemented the PD controller as a two-state feedback controller, which gives us a 2nd order system with two dimensions. In this case, we are completely ignoring the rotor angle and velocity and only regulate the pendulum angle. We use pole placement to maintain the pendulum at the inverted position. Due to the lack of restriction on the rotor velocity, the two-state feedback controller is hard to stabilize. Consequently, we enhance the model by adding the velocity of the rotor as the third state. In this case, our goal is to pull the velocity eigenvalue in LHP while ignoring its position, and the coefficients obtained from pole placement achieve better control of the system.

Additionally, the friction compensation is beneficial for our control system. Before adding friction compensation to the feedback controller, the pendulum is difficult to stabilize, and the rotor velocity is unstale. The reaction wheel tends to overcompensate the arm movement. To derive the coefficients for friction compensator, we treat it as a linear function. First, we remove our controller and fix the motor. Then, we manually adjust our reference signal, and the Simulink will automatically adjust for friction by counteracting its value as a function of speed. In this case, we will have a list of values for reference signals and the list of values for control effects. In this case, we can obtain the coefficients for the friction compensator.

2. <u>Provide a mathematical *proof* that the linearized, frictionless closed-loop system is stable in the inverted position</u>

We have the following state-space model:

$$\begin{array}{ccccc} \dot{x} & = & A & x & + \; B\,u \end{array}$$

$$\begin{bmatrix} \delta\dot{\theta}_p \\ \ddot{\theta}_p \\ \delta\dot{\theta}_r \\ \ddot{\theta}_r \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \omega_{np}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta\theta_p \\ \dot{\theta}_p \\ \delta\theta_r \\ \dot{\theta}_r \end{bmatrix} + \begin{bmatrix} 0 \\ -b_p \\ 0 \\ b_r \end{bmatrix} u$$

Where $\omega_{np}{}^2 = 63.8394$

We know that $u = -Kx$, where $K = [k_1 \;\; k_2 \;\; k_3 \;\; k_4]$

So, we plug in the equation and get:

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

In order to find the equilibrium, we set $\dot{x} = 0$,

$$(A - BK)x = 0$$

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ a + b_p k_1 & b_p k_2 & b_p k_3 & b_p k_4 \\ 0 & 0 & 0 & 1 \\ -b_r k_1 & -b_r k_2 & -b_r k_3 & -b_r k_4 \end{bmatrix} x = 0$$

Using the K values we found before and b_p, b_r

$$K = [-180.37 \quad -12.4 \quad 0 \quad -0.0215]$$

$$B = \begin{bmatrix} 0 \\ -1.034 \\ 0 \\ 192.7882 \end{bmatrix}$$

$$A - BK = 10^4 * \begin{bmatrix} 0 & 0.0001 & 0 & 0 \\ 0.025 & 0.0013 & 0 & 0 \\ 0 & 0 & 0 & 0.0001 \\ -3.4773 & -0.2391 & 0 & -0.0003 \end{bmatrix}$$

Use MATLAB function to calculate the eigenvalues of the matrix A-BK:

$$\begin{bmatrix} 0.0000 + 0.0000i \\ -3.1059 + 9.8846i \\ -3.1059 - 9.8846i \\ -2.4649 + 0.0000i \end{bmatrix}$$

Since all the eigenvalues are in the LHP, the system is stable.

To achieve the equation (A-BK)x = 0, the equilibrium points of the closed loop system should be:

$$x = \begin{bmatrix} 0 \\ 0 \\ whatever \\ 0 \end{bmatrix}$$

From what we defined in part2,

$$\delta\theta_p = \theta_p - \pi = 0$$
$$\delta\theta_r = \theta_r = whatever$$

By plugging in the values, we can get the following equation:

$$\begin{bmatrix} \theta_p \\ \dot{\theta}_p \\ \theta_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \pi \\ 0 \\ whatever \\ 0 \end{bmatrix}$$

Therefore, we can tell that as t goes to infinity, θ_p goes to π.

3. Table

	$\delta\theta_p$	$\delta\dot{\theta}_p$	Max Pulse	Max Disturbance
Two-state Feedback	7.7	7.6	6.6	4.3
Three-state Feedback	10.8	10.8	6.8	6.2

Table 1: Robustness of Two-state & Three State Feedback

4. Our system was able to stabilize once we manually moved the rotor to top. It will keep swinging slightly back and forth, but it is capable of withstanding a small outside force. One thing to note is that the swing in positive direction and negative direction is uneven, this could be caused by a slight offset in the measurement of the angle of the pendulum. Alternatively, this error could be a result from the friction compensation model, because due to hardware issues, we have to switch the pendulum that we used to obtain the friction coefficients.

Full State Feedback Control with Decoupled Observer

1. Observers are used because we can not directly obtain all the internal value of a system. An observer can let us have some good estimation on the internal values based on values that are known. We can decouple a 4-state observer design into two 2-state observers because we can make the two corners of the 4-state observer zero, which would not change the behavior of the observer dramatically, and then decouple it into two 2-state observers. One advantage of the two 2-state observers is that by zeroing out the conners, it ignored some of the noise in the system, which actually resulted in a system with better performance (Higher Max Pulse and Max Disturbance values).
2. Prove that the observer states converge to the real states over time

$$e = x - \tilde{x}$$
$$\dot{x} = Ax + Bu$$
$$\dot{\tilde{x}} = (A - LC)\tilde{x} + Bu + Ly = (A - LC)\tilde{x} + \dot{x} - Ax + LCx$$
$$\dot{e} = \dot{x} - \dot{\tilde{x}} = (A - LC)(x - \tilde{x}) = (A - LC)e$$

If we set e_dot to zero, and solve for e (as shown below)

```
A = [0 1 0 0; 69.84 0 0 0;0 0 0 1; 0 0 0 0];
C = [1 0 0 0; 0 0 1 0];


C1 = [1 0];
C2 = [1 0];

M = [0 1;69.84 0];
N = [0 1;0 0];


poles=[-48.87+i*9.57, -48.87-i*9.57, -51,-45];

poles1 = [-48.87+i*9.57, -48.87-i*9.57];
poles2 = [-51,-45];


L1 = place(M',C1',poles1)';
L2 = place(N',C2',poles2)';
L=[L1, zeros(2, 1); zeros(2, 1), L2];


e=linsolve(A-L*C,[0;0;0;0])
```

```
e =

     0
     0
     0
     0
```

As we can see the e = 0, which means that x_tilde does converge to x in the end.

3.

$\delta\theta_p$	$\delta\dot{\theta}_p$	Max Pulse	Max Disturbance
0.038	N/A	4.6	4.5

Table 2: Robustness of Decoupled Observer

4. Our system was able to stabilize once we manually moved the rotor to top. Just like the full state feedback control with friction compensation, it will keep swinging slightly back and forth. It is still capable of withstanding a small outside force, but smaller than the full state feedback control with friction compensation.

Conclusions

In this lab, we designed four controllers that all achieved stabilization: Two-state feedback, Three-state feedback, Observer controller, and Decoupled and redesigned observer controller. The standard of judging is the stability of the controllers. The higher pulse and disturbance the controller can take before becoming uncontrollable, the better we rate the controller. According to this standar, we found out that the Three-state feedback controller is the best. It can withstand about 10% more pulse and 50% more disturbance than other controllers.

Overall this lab went really well for our group, we learned a lot on how to design a desired controller and observer for a controller system.

Extra Credit

Part 6: In part 4 and 5 of the lab, we have built a model that can stabilize the Rotor at top. For the rotor to also stabilize quickly at bottom, we “tricked” the system by not adding the “-π” to theta_p, so it would stabilize as if it was trying to stabilize on top. (See appendix for our Windows Target Model).
However, this design does come with a slight problem: the rotor will keep spinning at the bottom unless it is perfectly aligned. This problem can be solved by adding another switch to cut the power for the rotor off when it is below a certain level (close to bottom). Unfortunately, we didn’t have enough time to implement and test this idea in our design.

Part 7: To accomplish the swing up action, we divided the rotor behavior into four conditions: based on the sign (+ or -) of the sin (theta_p), and the sign (+ or -) of the first derivative of theta_p, which represent the direction of motion. We then assign the direction of the acceleration of the rotor so it will slowly build up speed until a certain height. Once a certain height is reached, it will switch to the design that we used in part 4 and 5 to stabilize the system. Unfortunately, we didn’t have enough time to implement and test this idea in our design.

Appendix

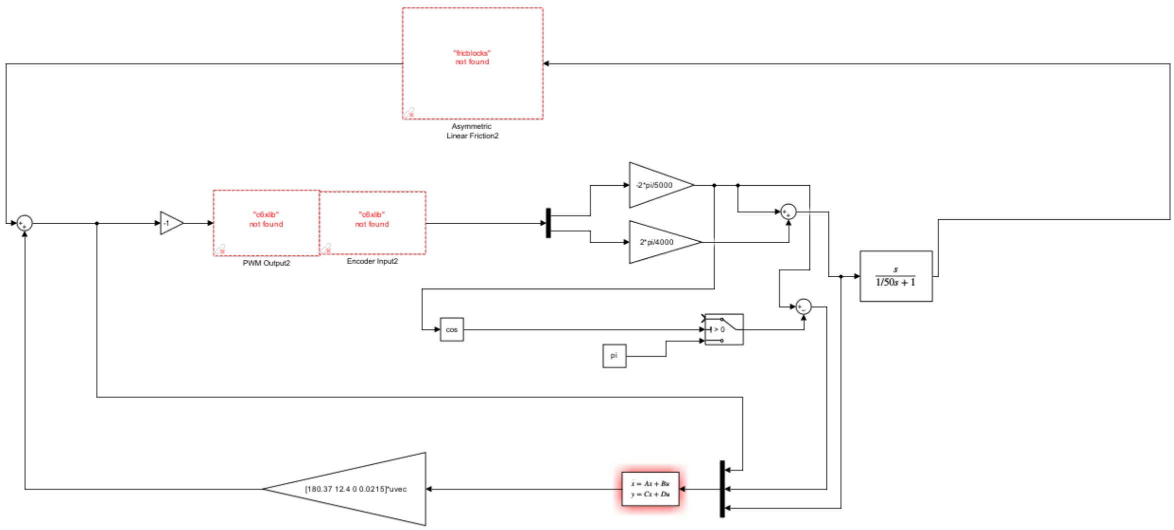

Figure 2: Windows Target Model