Lab 6d: Self-Erecting Inverted Pendulum (SEIP)

"Life swings like a pendulum backward and forward between pain and boredom." – Arthur Schopenhauer

1 Objectives

The goal of this project is to design a controller that starts with the pendulum in the "down" position and then swings it into and maintains it in the "up" position. In this lab you have a lot of freedom in implementing the controller, and you are encouraged to experiment. That said, these instructions will guide you through the design of a non-linear controller based on a so-called "energy-pumping" method. Since this class is primarily about control techniques for linear systems, you will not be tested on the content of this lab.

2 Introduction

You already have the model of the plant when the pendulum is in the vertical position. The challenge now is to understand how to swing the pendulum up. The key to making this work is to split the controller into two parts: one part will be the "swing-up" controller while the second part will be the "balancing" controller. Once the swing up controller manages to position the pendulum almost upright, almost at rest, the balancing controller should "take over" and maintain the pendulum in a vertical state. One way to implement this in Simulink is by using the Switch block, which is depicted below.

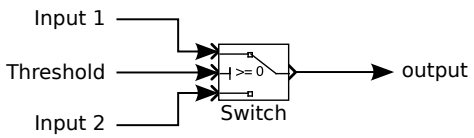

Figure 1: Simulink Switch block with relevant signals labeled

Its usage is pretty intuitive. It acts like a two signal multiplexer – it will pass on only one of the two inputs that it is given based on the threshold input. Make sure you double-click on the block to set your threshold value and other threshold options.

The usage of different controllers based on different conditions turn this system into what is known as a *switched system*. The theory of switched systems falls under the topic of *hybrid systems*, which we will not touch on here[1]. Our switched system is simple and intuitive enough to not require any knowledge of hybrid systems techniques. At this point, you should have a few (pole placement, LQR) working linear balancing controllers, so feel free to use whichever one you prefer to keep the pendulum in the "up" position, once you have reached it.

[1] Hybrid systems are the topic of EE291E / ME290Q.

3 Pre-Lab

For the swing-up controller, we will use an energy-pumping method. This technique is described in [**?**], for example. The idea is to use the control input to inject into the system the right amount of mechanical energy which will bring the pendulum to the upright position.

3.1 Energy of the Pendulum

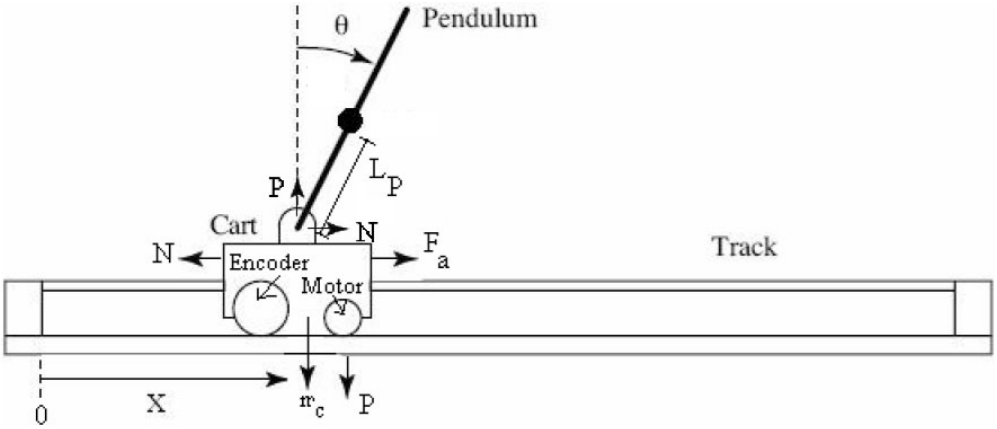

Figure 2: Free body diagram of the inverted pendulum setup (ignoring friction)

1. Consider the referential attached to the pivot of the pendulum. Write the expression of the total mechanical energy of the pendulum (without the cart) $E(\theta, \dot{\theta})$, as a function of θ, $\dot{\theta}$, and the system constants (m, J, L_p). The mechanical energy will have a kinetic term plus a potential term. Use a potential energy offset such that the total energy is zero when the pendulum is at rest in the upright position, i.e. $E(0, 0) = 0$.
2. Calculate the energy $E(\pi, 0)$ of the pendulum at rest at $\theta = \pi$.
3. The idea of the swing-up controller is to increase the energy from this initial value to the target value 0. In order to increase the energy, we will compute the derivative of E: Let $\ddot{x}$ be the acceleration of the pivot point. Given the dynamics of the pendulum

$$J\ddot{\theta} = mgL_p \sin(\theta) - mL_p \cos(\theta)\ddot{x}$$

where $J = \frac{4}{3}mL_p^2$, show that the derivative of the total energy of the pendulum is

$$\frac{d}{dt}E(\theta, \dot{\theta}) = -mL_p\ddot{x}\dot{\theta}\cos(\theta)$$

Therefore to increase the energy of the system as fast as possible, we would like to apply maximal positive acceleration when $\dot{\theta}\cos(\theta) < 0$, and negative acceleration when $\dot{\theta}\cos(\theta) > 0$. Recalling (from lab 6a) the relation between acceleration and voltage applied to the motor:

$$\ddot{x} = \alpha V - \beta\dot{x} - \gamma\theta$$

(where $\alpha > 0, \beta$ and γ are constants that depend on the physical parameters of the system), we

observe that $\ddot{x}$ is maximal when V is maximal. This results in the following simple control law (sometimes called bang-bang control)

$$V = \begin{cases} 0 & \text{when } E(\theta, \dot{\theta}) \geq 0 \\ V_{\max} & \text{when } E(\theta, \dot{\theta}) < 0 \text{ and } \dot{\theta}\cos(\theta) < 0 \\ -V_{\max} & \text{when } E(\theta, \dot{\theta}) < 0 \text{ and } \dot{\theta}\cos(\theta) > 0 \end{cases}$$

where $V_{\max} = 6V$.
In practice, we'll approximate this bang-bang control with a control policy that doesn't have discontinuities, as you will see in the Lab section.

3.2 Obtaining Derivatives

Obtaining estimates for $\dot{x}$ and $\dot{\theta}$. Recall from the previous labs that the numerical derivatives of the state and angle are of very poor quality. To remedy this, in Lab 6b we built an observer to obtain an estimate $\hat{x}$ of the system state. However, the observer uses the linear system model and is therefore only valid for $\theta \approx 0$. Since we are building a Swing-Up controller, we are using the full range $\theta \in [0, 2\pi)$, so we have to take the non-linearities in the system into account. The linear observer would not be adequate anymore, and in fact might diverge[2]. Instead, we will use a dynamical system approximating a derivative for obtaining estimates of $\dot{x}$ and $\dot{\theta}$.

Recall that the derivative operation $\dot{x}(t)$ in the Laplace domain is described by $sX(s)$ (assuming initial conditions are zero). Since we cannot implement a pure differentiator with transfer function $G(s) = s$ in Matlab, we will approximate it by a transfer function of the form

$$D(s) = sH_{lp}(s) = s\frac{1}{c_{lp}s + 1}$$

with parameters c_{lp} (lp for low-pass). Note that this can be viewed as a differentiator combined with a lowpass filter. Multiplying by the low-pass filter will result in a phase-shift. We would like to design in such a way as to ensure that the phase shift remains small, while not attenuating frequencies of interest too much.

- Calculate the largest value of c_{lp} such that at the natural frequency $\omega_0 = \sqrt{\frac{g}{l}}$, the phase shift is $|\phi(H_{lp}(j\omega_0))| \leq 5$ degrees.

[2]Observer design for nonlinear systems is very hard and for general cases largely remains an open problem.

4 Lab

We will design a switched Simulink system that starts with the pendulum pointing vertically down and then self-erects and balances it vertically upwards. The controller used in this lab is more complex than the ones we have seen in previous labs, so these instructions will walk you through the design. You are encouraged to also try other methods and report your findings.

4.1 Implementing the Swing-Up Controller on the Hardware

4.1.1 The Full System

Figure 3 shows the Simulink block diagram of the overall controller. The main tasks are abstracted away in the following subsystems:

- **IP Hardware**: Interface to the pendulum hardware.
- **Swing-Up Controller**: Controller capable of swinging up the pendulum from the "down" position. This block implements the control strategy discussed in the PreLab.
- **Mode Selector**: Chooses between the Swing-Up and the linear stabilizing controller, depending on the current system state.
- **Observer**: Observer for getting a good state estimate when the pendulum is in the "up" position.

Note the delay block that outputs into Switch1 in Figure 3. This is to ensure that we use our linear observer only after it has had sufficient time to converge to a good estimate.

These subsystems are discussed in detail in the following sections.

4.1.2 The Inverted Pendulum Hardware

The interface to the pendulum hardware[3] is shown in Figure 4. Since we have to take the non-linearities in the system into account when swinging up we cannot simply use a linear observer for providing estimates of the cart velocity $\dot{x}$ and pendulum angular velocity $\dot{\theta}$. Instead, we will use the dynamical systems approach discussed in section 3.2 of the Pre-Lab to obtain estimates of $\dot{x}$ and $\dot{\theta}$ from the measured outputs x and θ. This happens in the "Derivative_x" and "Derivative_theta" blocks in Figure 4. Use the value of c_{lp} determined in the Pre-Lab.

The saturation block should be familiar from Lab 6a (if you implement the custom saturation block, you can use that one instead). The "Angle Initialization" block is used to transform the coordinates of the system such that $\theta = 0$ in the "up" position (recall that the zero positions of x and θ are the respective positions when connecting to the target). This can be achieved by letting

$$\theta = \mathrm{mod}(\theta', 2\pi) - \pi$$

[3]Ignore the "Bad Link" blocks in this and the following figures – this is due to the fact that the model was printed from a machine that does not have the QuaRC software installed.

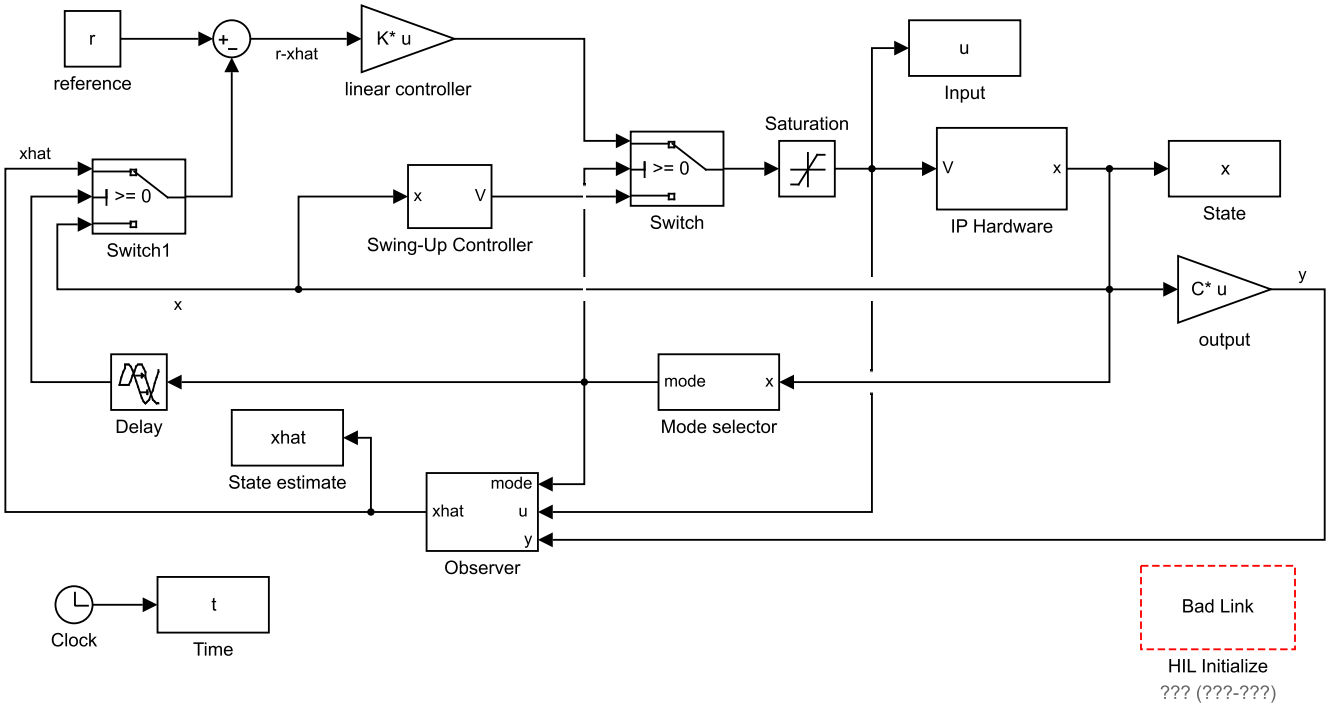

Figure 3: SEIP block diagram with subsystems

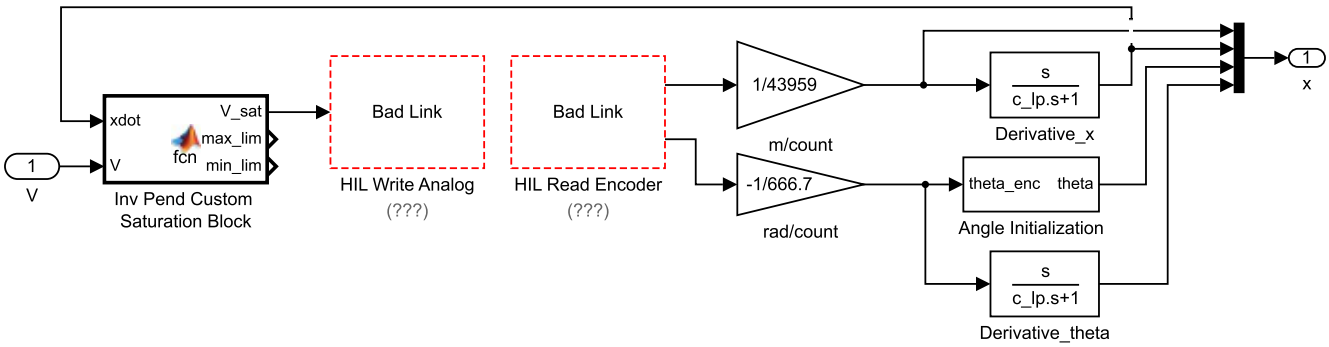

Figure 4: IP Hardware subsystem

where θ' is the angle reading from the encoder (after the conversion factor). Implement this function using Simulink blocks and create a subsystem just as done in Figure 4.

4.1.3 The Mode Selector

The Mode selector block is shown in Figure 5. In this block we check whether the absolute value of θ is less than some threshold `thresh` and choose the value of the mode signal accordingly. In particular, we want the following behavior:

$$\text{mode} = \begin{cases} 1 & \text{if } |\theta| \leq \text{thresh} \\ -1 & \text{otherwise} \end{cases}$$

The mode signal is used to switch between Swing-Up and stabilizing controller, and also to switch to using the state estimate $\hat{x}(t)$ provided by the observer for the stabilizing control. You will be tuning the value of the threshold during the lab. A good starting point is thresh $= 10\,\text{deg}$.

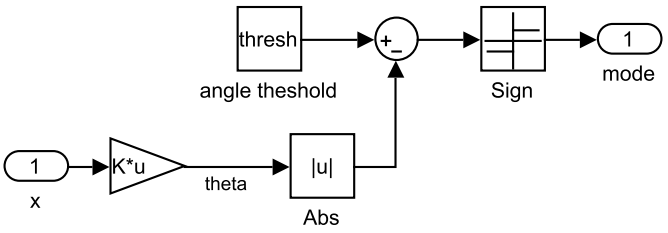

Figure 5: Mode selector subsystems

4.1.4 The Swing-Up Controller

Figure 6 shows the internals of the Swing-Up controller. The Block E computes the energy of the pendulum as a function of the angle θ and the angular velocity $\dot{\theta}$, as derived in the Pre-Lab. Implement this function using Simulink blocks and create a subsystem.

Note that, when the pendulum is at rest in the "down" position then $\dot{\theta} = 0$, so $E\dot{\theta}\cos(\theta) = 0$. Thus, it is ambiguous whether the swing-up controller will swing left or right initially; small noise factors would likely decide. Thus, Switch1 in Figure 6 ensures that for the first 50 ms, the pendulum will go to the right.

The gain K_{pump} should be large so as to maximize the rate at which energy is "pumped" into the pendulum. A good starting value is $K_{pump} \approx 50$. The switch in 6 makes sure that the Swing-Up controller stops pumping energy into the pendulum as soon as $E \geq 0$.

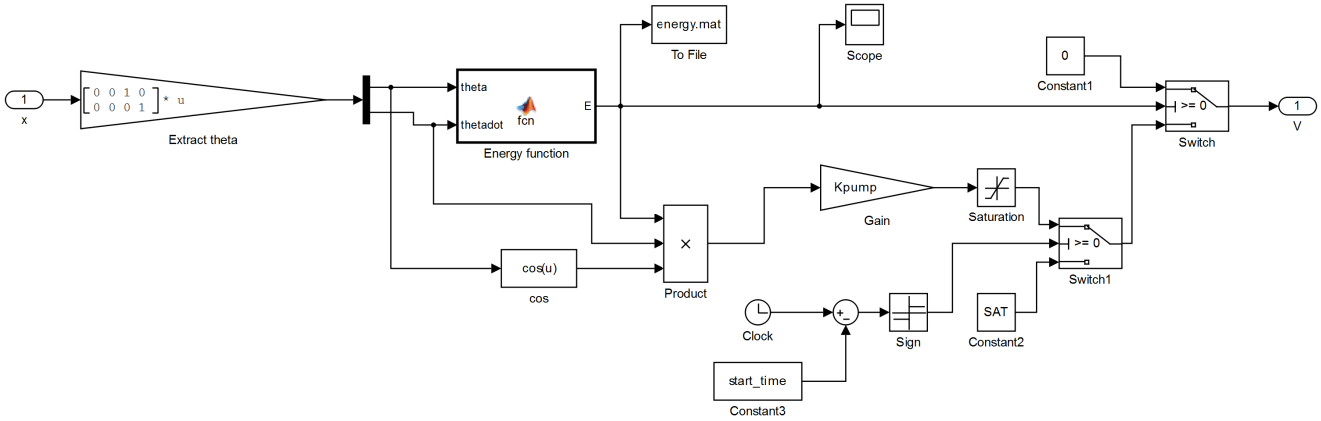

Figure 6: Swingup Controller subsystem

4.1.5 The Observer

Figure 7 shows the observer subsystem. The part on the right should be familiar from Lab 6b. The part of the left is necessary to ensure that the observer is only in use when the linearization of the system is reasonably accurate (i.e. when the mode signal is non-negative, which is the case whenever $|\theta| \leq$ thresh. In particular, if $|\theta| >$ thresh the observer is provided with zero voltage input signal, $V = 0$, and zero output signal, $y = [0\ 0]^T$. As soon as mode $= 1$, i.e. when $|\theta| \leq$ thresh, the observe is provided with the true inputs V and the true system output y. Note that this will force the observer state $\hat{x}$ to converge to

zero whenever the pendulum is not near its upright position (this ensures that the observer is only used when the linearized model is valid). When the pendulum gets close to the upright position ($\theta \approx 0$), then the observer will converge quickly to the true state x. The main point here is that if the observer were to run the whole time, the estimate $\hat{x}$ might diverge since the linear model used in the observer is not a good approximation anymore.

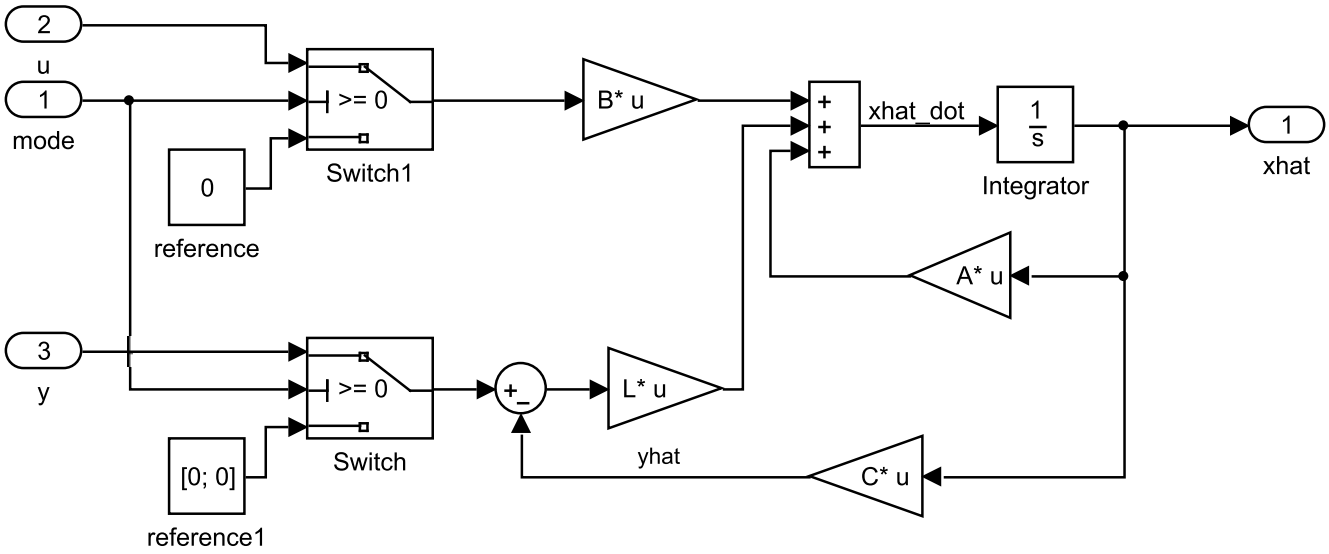

Figure 7: Observer subsystems

4.2 Testing and debugging the Swing-Up Controller

Before testing your swing-up controller, re-verify the following:

- make sure the angle estimate is correct. The angle should be 0 in the upright position and π when pointing down. In particular, it is important that the angle is exactly 0 when upright, otherwise the linear controller will try to stabilize around a non-vertical angle. You may need to adjust the conversion factor of the angle encoder.
- make sure you have the correct energy estimate. The energy should be minimal when the pendulum is at rest pointing down, and 0 when the pendulum is at rest upwards. Also test that the energy transfer between the kinetic and potential terms occurs as you expect. To do this, disconnect the controller and plot the energy estimate when the pendulum is oscillating freely. The total energy of the system should be roughly constant (it will be decreasing because of friction, which is not modeled here). In particular if the total energy oscillates, this means the model is slightly off, and you need to adjust one of the constants in the kinetic or potential term (J or L_p).

Now, test the switched controller. If the controller fails to swing up the pendulum or to stabilize around the $\theta = 0$ position, you can tweak the parameters of the system (switching threshold, gains, etc.) until you obtain a satisfactory performance.

4.3 Lab Report

When you submit your report, provide a link to a video with your controller in action and include the following plots:

- position x and angle θ. Show at which point the switch occurs (by plotting the angle threshold)
- control input u
- observer state estimates $\hat{x}$ (include velocity and angular velocity)
- energy E

Also, write a short description of the process for designing the swing up controller and what each part is doing, along with the block diagrams.