SWING UP CONTROL OF ROTARY INVERTED PENDULUM IN MATLAB

V ADARSH (21307R001)
PRADEEP CHUDASAMA (213070111)

1 Introduction

The aim of this experiment was to implement a swing up control on inverted pendulum and then balance it at upright position using LQR technique. The simulink model in MATLAB was already provided. The task was to write a MATLAB code that calculates the required torque to the motor for achieving the aforementioned task.

2 Swing Up Control

The pendulum was initially in down position. The first task was to "swing up" the pendulum. For this appropriate torque was needed for the base motor. An Energy based control law was used for this [1].
The equations of motion of how the pendulum angle would change based on a given "acceleration" to base motor shaft can be derived using Newton's laws of motion.
Let,
α= pendulum angle with respect to vertical along the "upright" position, measured positive in clockwise direction
u= acceleration given to base motor shaft
m=mass of pendulum
l=length from pivot to pendulum center of mass
J=pendulum moment of inertial with respect to pivot
g=acceleration due to gravity ($9.81 m/s^s$)
The equation of motion is :

$$J\ddot{\alpha} = mglsin(\alpha) - mulcos(\alpha) \tag{1}$$

For the system in (1) is we let the states be $\alpha, \dot{\alpha}$, then we have 2 equilibrium points: $\alpha = 0, \dot{\alpha} = 0$, the upright position, an unstable equilibrium, and $\alpha = \pi, \dot{\alpha} = 0$, the down position, a stable equilibrium point. The final aim is to maintain the pendulum at the unstable equilibrium point by giving an external input via a base motor.

The energy of a simple pendulum can be written as a sum of kinetic energy of pendulum at an instant plus the potential energy. The kinetic energy is simply proportional to square the velocity of pendulum:
$KE = \frac{1}{2}mv^2 = \frac{1}{2}m(l\dot{\alpha})^2 = \frac{1}{2}ml^2(\dot{\alpha})^2 = \frac{1}{2}J\dot{\alpha}^2$
Potential energy is given by how high the pendulum is.
$PE = mgl(cos(\alpha) - 1)$

$$E = KE + PE = \frac{1}{2}J\dot{\alpha}^2 + mgl(cos(\alpha) - 1) \tag{2}$$

Intuitively, to understand the swing up strategy consider Fig-1. Say the pen-

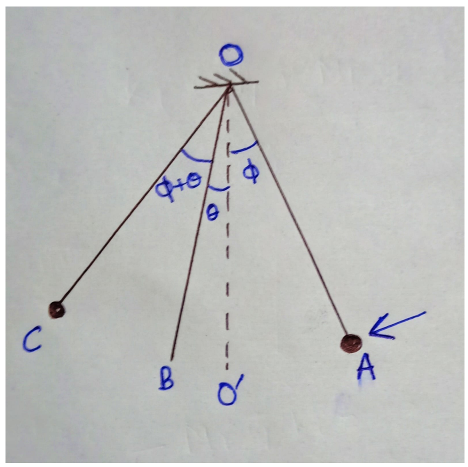

Figure 1: swing up strategy

dulum starts at "A" with 0 velocity, then it will symmetrically oscillate about OO' and will come o rest. Now say we give it a push at A in the direction as shown in figure, then the pendulum will now swing up to "C" and this swing is symmetric with respect to OB which is shifted slightly with angle θ. Notice pendulum now has reached a position such that it's angle with respect to OO' has increased by "2θ". This means pendulum is now closer to upright position than it was at A. At C again if we apply a force then it will again swing counter-clockwise to reach a position closer to upright position than at C. If we keep doing this we can achieve the desired swing up. This force is what we are trying to achieve using a reversal of base motor torque every-time the pendulum switches it's swing, until it finally reaches upright position.
From the equation (2), we can see that energy,E is $-2mgl$ at bottom when $\alpha = \pi$. If we calculate the energy by placing the pendulum at different angles, (NOTE: here we consider KE=0 since we are just placing the pendulum at different positions with zero velocity), we can see energy is always greater than $-2mgl$ and as we move closer to $\alpha = 0$ energy increases to 0. And hence, in energy control we want to find a control torque such that the pendulum at each

reversal of swing is placed at an higher angle and hence a higher energy, i.e we want to increase the energy to 0.
For this we look at time derivative of E:
Using (1) we get:
$\frac{dE}{dt} = (-mulcos\alpha)\dot{\alpha}$ (3)
Since we want energy to increase we want the derivative of E to be positive means "u" should be chosen such that the derivative is positive. To get the control law we can use Control Lyapunov Function (CLF).
CLF is used to test whether a system is asymptotically stabilizable, i.e for any given state $\exists$ a control input "u" such that if we apply it at that instant the system can be brought to it's equilibrium state. If one can find a CLF $V : D \subseteq R^n \rightarrow R$ and a control input "u" such that:
$\dot{V}(x,u) < 0$ then it means the system can be brought to the desired equilibrium point. In pendulum the desired equilibrium point is $\alpha = 0, \dot{\alpha} = 0$. [1] uses a CLF,
$V = \frac{(E-E_0)^2}{2}$ where E_0 is the desired energy (0 in our case)
we obtain,
$\dot{V} = -mulcos\alpha(E - E_0)\dot{\alpha}$
define
$u = k(E - E_0)\dot{\alpha}cos\alpha$ (4)
then
$\dot{V} = -mlk(cos\alpha(E - E_0)\dot{\alpha})^2$
This proves that using the control law in (4) one can bring the pendulum to upright position. (4) ensures CLF V decreases to 0, i.e $E \rightarrow E_0$, but to ensure fast change of energy we use:
$u = u_{max}sign((E - E_0)\dot{\alpha}cos(\alpha))$ (5)
In the control law in (5) the "sign(...)" part is crucial which tells when to reverse the motor torque so as to achieve a swing up with a pump in energy. To understand this, refer to Fig-1, and consider that $\alpha = 0$ in upright position and increases to π clockwise. Then at "A" when we apply a "positive u" and it reaches "C", there α is between π and $3\pi/2$ and since pendulum now starts to swing towards its stable equilibrium at "C", $\dot{\alpha} < 0$ and $E - E_0 < 0$ so $sign((E - E_0)\dot{\alpha}cos(\alpha)) < 0$ means we apply a "negative u", i.e switch the motor torque.

3 LQR Control

In this section we consider the state vector:
$x = [\theta, \alpha, \dot{\theta}, \dot{\alpha}]$
where θ is the base motor angle. LQR is used to balance the pendulum in upright position. LQR is used for linear systems, but pendulum is a nonlinear system and hence the system is approximted as a linear system in the neighbourhood of it's unstable equilibrium $\alpha = 0$ using linearization. One can then find the state space model. Since our aim is to balance the pendulum at $\alpha = 0$ we want to minimize the deviation of the state α from 0 and also we want to do it with

minimal deviation of angle of base motor with and avoid drastic changes in pendulum or base motor angle while doing this. Hence we can say we want to minimze the deviation of the state vetor "x" from 0. Here comes LQR, which uses a "Quadratic" cost function which is basically a quadratic weighted sum of state deviations and control input. LQR basically uses a full state feedback and requires that the system is controllable. LQR provides a feedback gain matrix "K" such that the cost function is minimized and the closed loop system is stable.
The cost function used in LQR is:
$J = \int_0^\infty (x^T Qx + u^T Ru)dt$
The integrand measures the state and input deviations and Q,R matrices are positive definite matrices, usually diagonal, whose entries act as weights for state deviation which signify which state is how important. So if say α's weight in Q-matrix is high this means even for a small deviation of it from 0 imposes a huge penalty and hence the resultant feedback matrix ensures minimal deviation of α. The integral goes from 0 to infinity which indicates that the "balancing" task is a infinite horizon control problem, meaning we need an input that always keeps the pendulum at upright position. So assume we start at some non zero α , then LQR gives a feedback gain by solving ricatti differential equation, such that if we use input $u = -Kx$ and evolve the system using the state space model derived, then the area under curve for α is minimized.
The "u" term in cost function is to ensure actuation effort is minimal.

4 MATLAB CODE

The basic algorithm was:

1. Start with swing up control with a control input given by
   $u = u_{max} sign((E - E_0)\dot{\alpha} cos(\alpha))$

2. If pendulum reaches close to upright position switch to LQR

3. If while balancing pendulum falls off beyond a certain range switch back to swing up

```
function [y]= f(x1,x2,x3,x4)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% x1= pendulum angle,       %
%%% x2= deriv_pendulum angle  %
%%% x3=base motor angle,      %
%%% x4=deriv_base motor angle %
%%% y=control torque to motor %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global A
global theta_ref

```

```
theta=0;
x1=mod(x1,2*pi);
if(x1>pi)
x1=x1-2*pi;
end
x3=mod(x3,2*pi);
E=0.5*0.00011*(x2^2)+(0.027*9.81*0.1832*(cos(x1)-1));  % Energy of pendulum
coder.extrinsic('lqr');
umax=2*45193.6548/100000;


u=-umax*sign(x2*cos(x1));



 if(abs(x1)<=20 && A==0) % check if pendulum near upright position
     A=1;
     theta_ref=x3;
 end


  if(abs(x1)>45*pi/180) % switch back to swing up, stop LQR
       A=0;
  end


y=0.0022127*u;

if(E>-0.02 && A==1) % perform LQR if energy greater than a certain value or pendulum close to upright
      alpha=x1;
      theta=x3-theta_ref;



     x_1=[theta;alpha;x4;x2];


     KM=-[    -0.3808   ;  21.5450    ;-0.4358   ;   2.5490];

    y=sum(KM.*x_1);


```

```

end
```

5 Simulation and Results

The A,B,C,D matrices were:

A=$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 136.8 & -1.66 & 0 \\ 0 & 112 & -0.669 & 0 \end{bmatrix}$

B=$\begin{bmatrix} 0 \\ 0 \\ 59.7 \\ 23.9 \end{bmatrix}$

C=I_4

D=0
For LQR,

Q=$\begin{bmatrix} 2.9 & 0 & 0 & 0 \\ 0 & 35 & 0 & 0 \\ 0 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$

R=[0]
Figure-2 shows the step response of the above system with above Q,R matrices generated using matlab's "lsim" command:

When we used the gain matrix obtained from above Q,R matrices in the MATLAB code specified in previous section, we were not able to achieve the upright balancing task. However we were able achieve swing up control, and the pendulum took a few swings before reaching upright position in about 5 seconds of simulation time.
Fig-3,4 shows the plot of pendulum energy and pendulum angle vs time. It can be seen that around 5 seconds, the energy as well as pendulum angle is very close to 0, meaning swing up control has swung the pendulum to upright position.

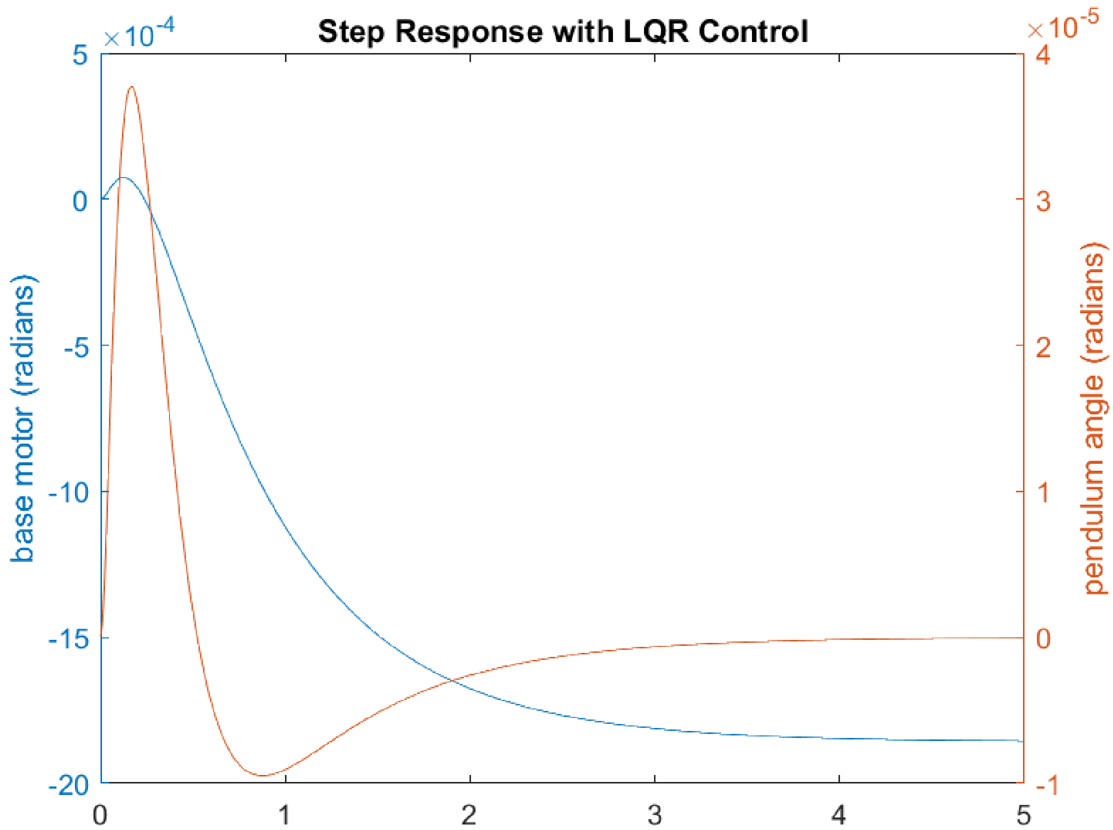

Figure 2: step response

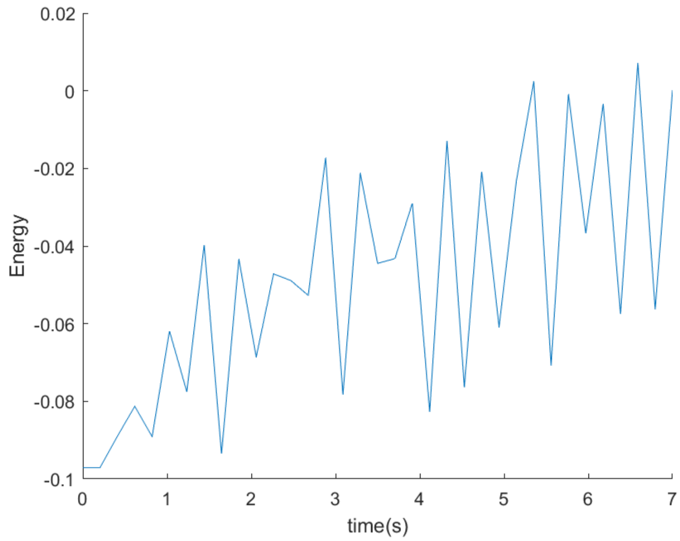

Figure-3

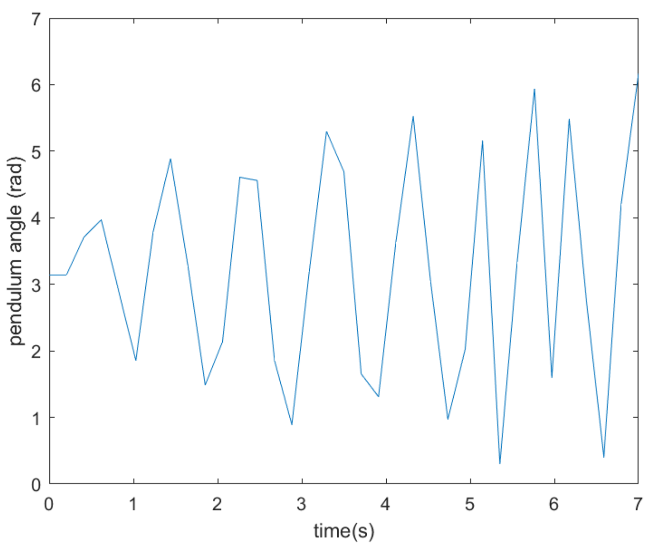

Figure-4

6 References

1. Åström, Karl Johan, and Katsuhisa Furuta. "Swinging up a pendulum by energy control." Automatica 36.2 (2000): 287-295.