EE 324

Control Systems Lab
Experiment 3
Inverted Pendulum

Tuesday Batch I
Group 9

Members

Parth Kothari – 14D070019

Sudeep Salgia – 14D070011

Arka Sadhu – 140070011

AIM :-

The aim of the experiment is to balance an inverted pendulum with the help of a micro-controller arduino using the Linear Quadratic Regulator (LQR) method for pole placement to realise a stable real time system under constraints of maximum plane sweep ($|\theta| < 30°$) and vertical deviation ($|\alpha| < 3°$) and voltage required ($V < 12V$) .

CODE :-

A sample code for getting the real time values of the θ from the inverted pendulum was given for the Arduino. We extrapolated the same to get the real time values of α as well. We wrote a matlab script to output the K matrix of the lqr method, and plotted it versus to time to ensure that the system is indeed stable and that all the transient conditions are met. The inputs to this are the A,B,C,D matrices which were given the datasheet. Along with this we also need to input the Q,R matrices which correspond to the weights of the state variables. We had to carefully choose the weights (Q,R matrices) so as to satisfy all the constraints.
From the theory of lqr, we know that the matrix K is independent of the states and hence constant for the run time of the program. So we directly hard-code this value of K into our arduino file.

In Arduino IDE, we first get the values of α and θ , and get the sampling time between the consecutive values to get an estimate of α° and θ° which are the corresponding time derivatives, and in total these make up the states of the system. Now we directly use the value of K to get the input for the next input and output it using the Arduino pins, which drive the system.

OBSERVATION & INFERENCES :-

The most dominant state was θ which corresponded to the horizontal sweep. This is directly inferred for the cost matrix Q and R, which are the inputs to the lqr method. We also noted that if we were to increase the cost of one particular (say for θ), while the system became faster in responding, and adhered the constraints more rigidly, the system became less robust. So if for the lower cost system, even if we hit the pendulum hard, it was able to become stable, even if it had to break the constraints for horizontal and vertical deflection, but if the cost is made to high, if we hit it harder than a threshold, it lost its ability to come back to the initial configuration. Therefore we had to a binary search to obtain a value which satisfied both of them.

PROBLEMS FACED AND SOLUTIONS :-

1. Getting to a stable value. Even though the MATLAB routine showed that the system should die down to zero, in practice it never became stable. We changed Q and R matrices appropriately (mainly by hit and trial) to get the first stable solution. After this we were able to change the Q and R matrices systematically to get the stable inverted pendulum.
2. We had to use a small delay in between two samplings. Extremely fast sampling rather deteriorated the performance, may be because reaction time of the inverted pendulum was much lesser than frequency of Arduino.

Values used by us:

x = [θ ; α ; $\dot{\theta}$; $\dot{\alpha}$]'

Q =

[18	0	0	0;
0	0.75	0	0;
0	0	2	0;
0	0	0	0.75]

R = 1

K = [-4.242641; 72.910965; -2.655214;9.602371]

θ deviation:

$\theta_{max\ left} = -0.25\ rad = -14.26°$

$\theta_{max\ right} = -0.46\ rad = 26°$

Total Deviation = $0.71\ rad = 40°$

α deviation

$|\alpha_{max}| = -0.03\ rad = 1.8°$