Double Inverted Pendulum Balance Using Optimal Control*

Erich Meissner[1], Erick Medina[2], and Anto Meliksetian[3]

***Abstract*— As part of the course curriculum, students were given the opportunity to pick from several choices of control projects. The listed projects all involved developing a model for a system, implementing the model in MATLAB, simplifying the model, and finding a control law to provide some level of performance. We chose to develop a dynamical model for the Furuta pendulum and develop control laws to stabilize the pendulum. We adapted the project to tackle the difficulty of stabilizing a double inverted pendulum mounted on a cart. This scenario is a highly unstable system, more so than the original Furuta pendulum. We sought to linearize the system around its equilibrium, and then the two segments are kept in an upright position using a common method that we learned in the course. We do it through LQR control functions. The system is then simulated in MATLAB.**

I. INTRODUCTION

The original project outline describes the Furuta pendulum to consist of a ”driven arm which rotates in the horizontal plane and a pendulum attached to that arm which is free to rotate in the vertical plane.” The control objective was ”to move the arm to ’spin-up’ the pendulum and then stabilize it vertically.” Within the project specification, it states that ”this project is easier than the others, and so, your team will have to do something truly novel to get a good grade.” Therefore our team decided to add another segment and place both segments on a moving cart.

We find it important to define the new specifications of our project. A double inverted pendulum system is an extension of the single inverted pendulum, and we have now mounted it on a cart. We define the new approach by referencing the original publications by K. Furuta [6]. As stated in the project, we recognize the experiment of balancing an inverted pendulum to be one of the most common control engineering problems. This system is especially interesting as it can be used to accurately describe many real life problems. Possibly the most obvious example of a stabilized inverted pendulum is a human being. When we stand upright, we are acting as an inverted pendulum with our feet as the pivot. Without constant small muscular adjustments, we would fall over. The human nervous system contains an unconscious feedback control system, which is the senses of balance and reflex, that uses input from the eyes, muscles, and joints. Even orientation input from the vestibular system consisting of the semicircular canals in our inner ears continuously make small adjustments to the skeletal muscles to keep us standing upright. Walking, running, or balancing on one leg puts additional demands on this system that mimics an inverted pendulum.

The double inverted pendulum is a nonlinear system with a high concentration of nonlinearities. We noticed that the linear quadratic regulator (LQR) feedback design, which we understand to be a nonlinear extension of the state-dependent Riccati equation, was the best approach to this dynamic problem. We acknowledge the matrices of the state space system to be dependent on the position of the pendulum and test the LQR method.

II. PENDULUM METHODOLOGY

A. Modeling

The procedure of deriving the differential equations that describe the model of a double pendulum was aided from several references below [1] [2] [3].

We outline our procedure in a simple way. We begin by dening the three objects of our model. These three objects are the cart, the lower pendulum and the upper pendulum. Each object has its mass *m-i*. Also, for the lengths of each pendulum, we defined them to be *l-1* and *l-2* respectively. Also, we denoted the kinetic energy for each mass as *KE-i* and the potential energy for each mass as *PE-i*. The energy of the system, therefore, is found by:

KE = *KE-1* + *KE-2* + *KE-3* and
PE = *PE-1* + *PE-2* + *PE-3*

Furthermore, our team investigated the Langrangian. We defined L to be the difference between kinetic and potential energy *L* = *KE-i* *PE-i* . The following figure illustrates our approach:

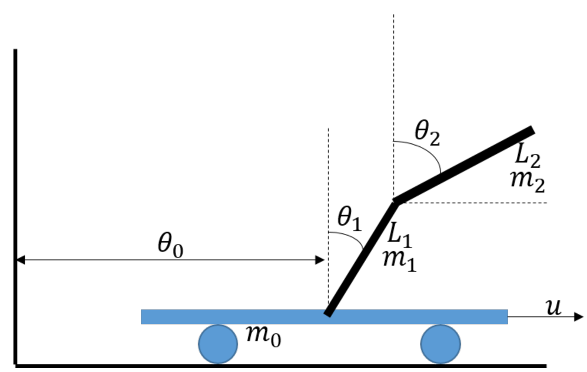

Fig. 1. Figure of our double inverted pendulum on a cart

*This work was a project of the ENEE463 course within the Electrical and Computer Engineering Department at the University of Maryland

[1] me@erichmeissner.com
[2] medina.erick.1ried@gmail.com
[3] anto1violin@gmail.com

Figure 1: Double Inverted pendulum on a cart

θ_0	Cart position
θ_0'	Cart velocity
θ_1	Angle of the lower pendulum
θ_1'	Angular velocity of the lower pendulum
θ_2	Angle of the upper pendulum
θ_2'	Angular velocity of the upper pendulum

We know the Langrangian equations of motion to be:

$$\frac{d}{dt}\frac{}{d\phi'} - \frac{dL}{d\theta} = Q$$

Here, we know Q to be the external forces acting on this complex system. Therefore, we define our dynamics as such:

$$D(\theta)\theta'' + C(\theta, \theta')\theta' + G(\theta) = Hu$$

Furthermrore,

$$D(\theta) = \begin{pmatrix} d1 & d2cos\theta1 & d3cos\theta2 \\ d2cos\theta1 & d4 & d5cos(\theta1 - \theta2) \\ d3cos\theta2 & d5cos(\theta1 - \theta2) & d6 \end{pmatrix}$$

$$C(\theta, \theta') = \begin{pmatrix} 0 & -d2sin(\theta1)\theta'1 & -d3sin(\theta2)\theta'2 \\ 0 & 0 & d5sin(\theta1 - \theta2)\theta'2 \\ 0 & -d5sin(\theta1 - \theta2)\theta'1 & 0 \end{pmatrix}$$

$$G(\theta) = \begin{pmatrix} 0 \\ -f1sin\theta1 \\ -f2sin\theta2 \end{pmatrix}$$

$$H = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

These equations we have defined for this two-segment inverted pendulum describe the system's motion. However, these equations are clearly nonlinear. As we were tasked in this project, we sought to linearize the control law. We linearized around the equilibrium:

$$(\theta_0, \theta_1, \theta_2, \theta', \theta', \theta') = (0, ..., 0)$$

This linearizations follows to the state space system:

$$x'(t) = Ax(t) + Bu(t)$$
$$y(t) = x(t)$$

The state vector, therefore, is:

$$x = \begin{pmatrix} \theta \\ \theta' \end{pmatrix} = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_0' \\ \\ \end{pmatrix}$$

We created an example to demonstrate our control law. If we plug in the following data for the martices above: m0 = 1.5kg, m1 = 0.5kg, m2 = 0.75g, L1 = 0.5m, L2 = 0.75m
Then, we end up with

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -7.4920 & 0.7985 & 0 & 0 & 0 \\ 0 & 74.9266 & -33.7147 & 0 & 0 & 0 \\ 0 & -59.9373 & 52.1208 & 0 & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -0.6070 \\ 1.4984 \\ -0.2839 \end{pmatrix}$$

B. *Linear Quadratic Regulator (LQR) resolution*

Now that we have defined the system, we sought to create a cost function that depends on the position and the input. We then minimized it with respect to the parameters we created. For our work, these equations hold true:

*J = x^T*Q*x + u^T*R*u*

Here, we defined *Q* and *R* to be our weight matrices for each parameter. We can set the dynamics to be anything in this model. An example could be:

Q = diag(5, 50, 50, 20, 700, 700) R = 1

However, we can input any other cost functions that we would like to define. These cost functions depend on the capabilities of a mechanical system used in a real life simulation that we found with respect to the Furuta Pendulum. The input that is used is of the form $u = -Kx$, where K is the state feedback matrix and is the solution of the Riccati equation. We took this approach due to the method we learned in class.

We can obtain this matrix from Matlab using the command k=lqr(sys,Q,R), where sys is the above state space system, to which we have set y = x so that the states are equal to the outputs. We consider the response for these initial conditions:

$$\begin{pmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_0' \\ \theta_1' \\ \theta_2' \end{pmatrix} = \begin{pmatrix} 0 \\ 10 \\ -10 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

We defined these initial conditions for any almost negligible changes from the equilibrium. Within Matlab, we only use radians even though we define the angles of deflection to be in degrees here. The states that we modeled are in the appendix below.

We also tested for different initial conditions to prove how holistic our control law is. Here, we tested for larger deflections toward the same direction. Our control law proved robust as we can see that the system can be balanced, although the cart is stabilized far from the point 0, to which it returns to with a very low pace. The graphs are also in the appendix below. These initial conditions are as follows:

$$\begin{pmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta'_0 \\ \theta'_1 \\ \theta'_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 20 \\ 20 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

REFERENCES

[1] Tobias Brull, Equations of motion for an inverted double pendulum on a cart (in generalized coordinates), Systems and Control Theory Course from the University of Berlin Institute of Mathematics.

[2] Pathompong Jaiwat, Toshiyuki Ohtsuka, Real-Time Swing-up of Double Inverted Pendulum by Nonlinear Model Predictive Control. Hiroshima, Japan: 2014.

[3] A. Bogdanov, Optimal Control of a Double Inverted Pendulum on a Cart. Department of Computer Science and Electrical Engineering, OGU School of Science and Engineering, 2004.

[4] Jiao-long Zhang, Wei Zhang, LQR self-adjusting based control for the planar double inverted pendulum, Physics Procedia, 2012.

[5] C. W. Anderson, Learning to control an inverted pendulum using neural networks, IEEE Control Systems Magazine, 1989.

[6] K. Furuta, M. Yamakita, S. Kobayashi, Swing-up Control of Inverted Pendulum Using Pseudo-State Feedback, Published November 1, 1992.

```
% The parameters are defined the same way as we did in our paper
m0=1.5 ;
m1 =0.5 ;
m2 =0.75 ;
L1 =0.5 ;
L2 =0.75 ;
dt= 0.02 ;
Q =diag([5 50 50 700 700 700]);
% There are several inputs that can be tested within these dynamics
R =1;
Nh =40;
g=10;
% Here are all of the system matrices definitions as written in the paper
d0=[m0 + m1 + m2, (m1/2 + m2)*L1, (m2*L2)/2;(m1/2 + m2)*L1, (m1/3 + m2)*L1^2,(m2*L1*L2)/2;(m2
*L2)/2,(m2*L1*L2)/2,(m2*L2^2)/3];
g=[0,0,0;0,-(m1/2 + m2)*L1*10,0;0,0,-(m2*L2*g)/2];
% The state space system follows
a=[zeros(3),eye(3);-inv(d0)*g,zeros(3)];
b=[zeros(3,1);inv(d0)*[1;0;0]];
c=eye(6);
sys=ss(a,b,eye(6),0)
% The feedback k is shown
k=lqr(sys,Q,R)
% Our new control law system!
sysnew=ss(a-b*k,b,c,0)
% The initial conditions, as discussed in the paper, are here
[y,t,x]=initial(sysnew,[0;deg2rad(10);-deg2rad(10);0;0;0],7);% [y,t,x]=initial(sysnew,[0;deg2
rad(20);deg2rad(20);0;0;0],10);
% Radians to degree definitions
y(:,2:3)=rad2deg(y(:,2:3));
y(:,5:6)=rad2deg(y(:,5:6));
% Finally, plot!
close all
for i=1:1:6
subplot(2,3,i)
plot(t,y(:,i))
grid
end
```

```
sys =

  A =
           x1       x2       x3       x4       x5       x6
   x1       0        0        0        1        0        0
   x2       0        0        0        0        1        0
   x3       0        0        0        0        0        1
   x4       0     -7.5   0.8036        0        0        0
   x5       0       75   -33.75        0        0        0
   x6       0      -60    52.14        0        0        0

  B =
           u1
   x1       0
   x2       0
```

```
   x3       0
   x4  0.6071
   x5    -1.5
   x6  0.2857

  C =
       x1  x2  x3  x4  x5  x6
   y1   1   0   0   0   0   0
   y2   0   1   0   0   0   0
   y3   0   0   1   0   0   0
   y4   0   0   0   1   0   0
   y5   0   0   0   0   1   0
   y6   0   0   0   0   0   1

  D =
       u1
   y1   0
   y2   0
   y3   0
   y4   0
   y5   0
   y6   0

Continuous-time state-space model.


k =

    2.2361 -597.3662  835.3448   28.6134   -6.4329  131.2304


sysnew =

  A =
            x1        x2        x3        x4        x5        x6
   x1        0         0         0         1         0         0
   x2        0         0         0         0         1         0
   x3        0         0         0         0         0         1
   x4   -1.358     355.2    -506.4    -17.37     3.906    -79.68
   x5    3.354      -821      1219     42.92    -9.649     196.8
   x6  -0.6389     110.7    -186.5    -8.175     1.838    -37.49

  B =
           u1
   x1       0
   x2       0
   x3       0
   x4  0.6071
   x5    -1.5
   x6  0.2857

  C =
       x1  x2  x3  x4  x5  x6
   y1   1   0   0   0   0   0
   y2   0   1   0   0   0   0
   y3   0   0   1   0   0   0
   y4   0   0   0   1   0   0
```

```
    y5    0    0    0    0    1    0
    y6    0    0    0    0    0    1

  D =
        u1
    y1   0
    y2   0
    y3   0
    y4   0
    y5   0
    y6   0

Continuous-time state-space model.
```