MECA-482 Project: Ball on Plate

Team #5

MWF 12:00-12:50

Members:

AbdulAziz Alqatami

Steven Augusto

Jose Luis Leon

Jarod Mica

Alec Mitchell

1. Introduction

   This project will track the ball's motion on a plate and adjust its motor angles accordingly so that the ball does not fall off the plate. This will be done by using Matlab and Simulink to model the controller as a PD controller that will be used to control the servo motors. The controller will be tested by using Simulink linked to CoppeliaSim to model it virtually. The model will use the integrated positioning on CoppeliaSim to give input to the controller that will control two servo motors that control the X/Y axis to position the ball.

2. Procedure

   Equipment List

   - Matlab
   - Simulink

- CoppeliaSim
- 3d ball on plate model

Mathematical Model

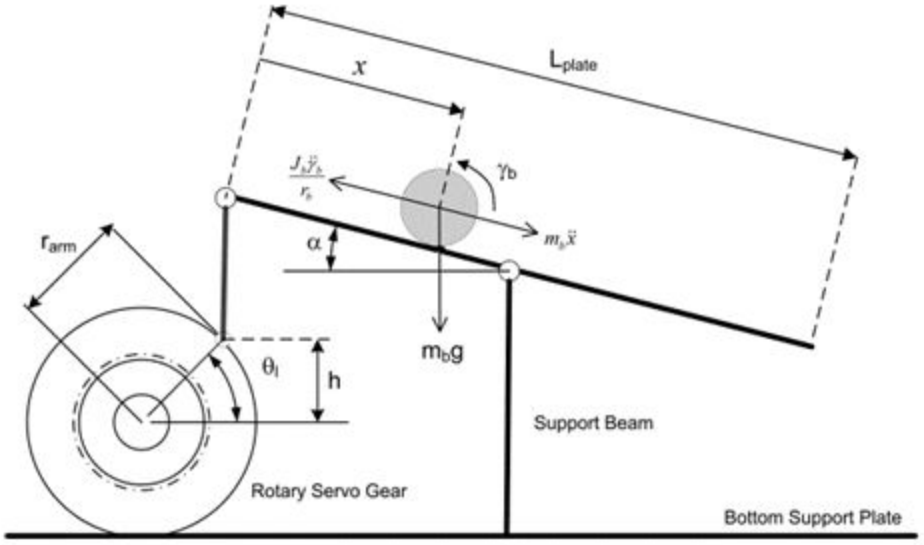


Figure 1: Free Body Diagram

This system can be modeled by doing the mathematics on a 2 dimensional version of a ball balancer and then combining two of these together to model the 3 dimensional case.

$mgsin(\alpha) = Jx''/r^2 + m''$

This equation represents the motion of the ball on the plate in reference to the plate tilt

$mgsin(\alpha) = 2mx''/5 + mx''$

By expanding the moment of inertia we can get like terms of mass, which will allow us to cancel out the mass portion of the equation.

$gsin(\alpha) = 7x''/5$

We can make the small angle approximation since the angles will be less than 15 degrees.

$g\alpha = 7x''/5$

Taking the Laplace transform of the equation above we get the transfer function for the position of the ball after an input tilt to the plate is given.

$$\frac{x(s)}{\alpha(s)} = \frac{5g}{7s^2}$$

The next step is to make a transform for the plate tilt in regards to the motor angle.

$$sin(\alpha) = 2sin(\theta) * r/L$$

The small angle approximation is once again used.

$$\alpha = 2\theta * r/L$$

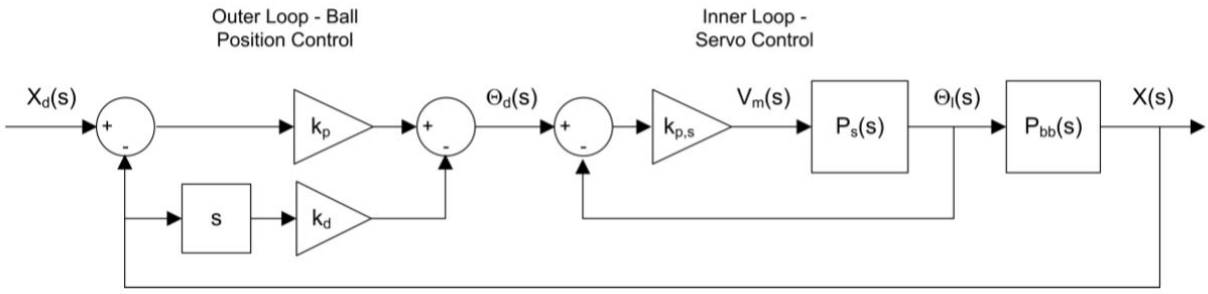
Taking the Laplace transform of the above equation we get the following equation

$$\frac{\alpha(s)}{\theta(s)} = 2r/L$$

By multiplying the two transfer functions above we get the x position based on the input angle for the motor which is our plant

$$\frac{x(s)}{\theta(s)} = \frac{10rg}{7Ls^2} = \frac{c}{s^2}$$

$$c = \frac{10rg}{7}$$



$$\theta_d(s) = \theta(s)$$

Since the motors are controlled by direct input on CoppeliaSim they are ideal and the inner loop can be neglected.

We then take the sum junctions before and after the outer loop to determine our transfer function for our controller.

$$\theta_d(s) = k_p(X_d(s) - X(s)) - k_d sX(s)$$

By plugging in this equation into the plant transfer function we get our system transfer function

$$\frac{X(s)}{X_d(s)} = \frac{ck_p}{s^2 + c\,k_d + ck_p}$$

We can then use these values to determine our natural frequency and damping ratio. However since we can change kp and kd we can alter them to get the values we require.

$\omega_n^2 = ck_p = 1.448$

$\varsigma = \frac{ck_d}{2\sqrt{ck_p}} = .4559$

For a settling time of 5 seconds and a PO% of 20% we got the following values for kp and kd

$k_p = 3.4476$

$k_d = 2.612$

All the previous equations were for the position in the x direction and can be used for the y position as well. This is because the systems are independent of each other so we can build a MIMO system by just combining two SISO systems.

Position Sensor

We were unable to use the camera sensor for the positioning of the balls so instead we used a built in function from CoppeliaSim in our matlab code that reads positions of objects in reference to another objects reference frame. By doing this we were able to get the ball's position relative to the plate's x and y coordinates.

Controller Design

The controller we decided to go with was a PD controller in order to minimize settling time. An integral portion was not added since this system has negligible steady state error and this would overcomplicate the system.

Simulation

With the simulations we were able to balance the ball on a set location on the plate indefinitely, however it was more complicated to make code that would allow the ball to follow a set path. In the Simulation folder there is a video of the ball balancing on the center of the plate.

Conclusion

The System was much harder to control than expected and the CoppeliaSim model had issues with the plate rotating if the plate was tilted too far. We were able to make a ball balance but were unable to make a path for the ball so there is room for improvement. Another way to

improve the code would be to read the output form the camera instead of using the built in function. However, this did help us learn the importance of controllers and how to use transfer functions with them