University of Waterloo
Faculty of Mechatronics and Mechanical Engineering

ECE 484
Digital Control Applications

Lab 2: Minor Loop Design and Beam Characterization

Professor John W. Simpson-Porco

Group 89
Station 17
Alistair Fink: 20601925
Michael HoSue: 20579851

October 30, 2019

Statement of Originality

The experiments conducted in this lab report were conducted on station 17.

The authors of this report acknowledge that (a) they have jointly authored this submission, (b) this work represents their original work, (c) that they have not been provided with nor examined another person's ball & beam project report, either electronically or in hard copy, and (d) that this work has not been previously submitted for academic credit.

Alistair Fink

Michael HoSue

1.0 Introduction

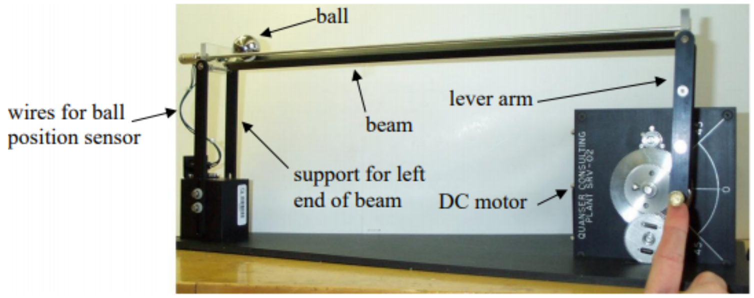

Figure 1: Ball and Beam Experimental Apparatus [1]

The experimental apparatus for this lab is a ball and beam apparatus that will be used to eventually balance a ball on the beam as shown in Figure 1. The two main sections of this experimental setup are the ball and beam and the gear transmission. The focus for this lab will be designing a controller for the previously characterized gear transmission as well as characterizing the ball and beam.

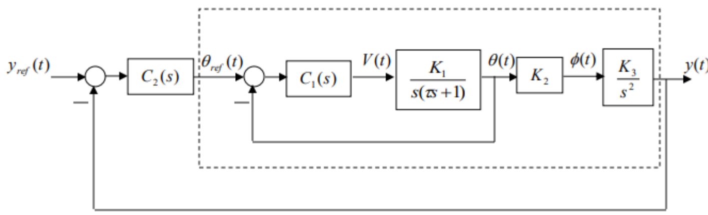

Figure 2: Full Experimental Apparatus Block Diagram [1]

As shown in Figure 2 the full experimental apparatus requires two feedback loops, the minor loop (inner loop) and the major loop (outer loop). This method of design is useful when multiple measurements are available such as this case where the angle of the gear and the position of the ball on the beam are both separately measureable. The idea for this method of design is that if we are able to design a feedback loop with a bandwidth of five to ten times faster than the bandwidth of the outer loop then the contents of the inner loop are effectively seen as a gain of one to the outer loop and can be safely ignored when designing the outer loop. [2] This means that if we can design a relatively fast feedback loop for the gear transmission then these dynamics can be ignored when designing the feedback for the beam later.

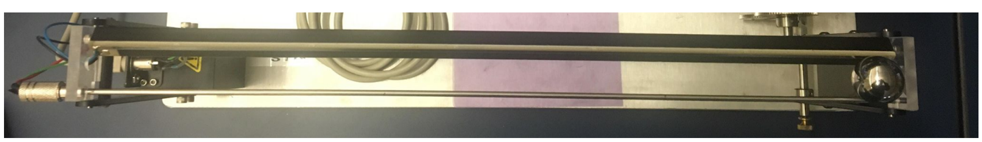

Figure 3: Ball and Beam Configuration

As can be seen in Figure 1 the beam is connected to the gear transmission through a lever arm which translates the rotational movement of the gear to angular movement of the beam allowing the beam to tilt to the left and right. The beam itself consists of a pair of rails which act as a circuit that gets completed when the ball connects them. Once these rails are connected the voltage drop can be measured across the two rails which varies due to the changing resistance of the circuit due to the changing ball's position. This essentially operates as a potentiometer and can be characterized to determine the ball's exact position on the beam.

The experiments conducted in this lab report were conducted on station 17.

2.0 Inner Loop Controller Design and Discretization

In order to design a controller we need to know the specifications to aim for. The lab manual gives us the following:

- Zero Steady State Tracking Error
- Maximum 0.5 Seconds Settling Time
- Maximum 5% Overshoot
- Motor Voltage Does Not Saturate

In order to design a control loop for the gear transmission we must also be able to model this gear transmission in the frequency domain. From lab 1 we know that we can model the gear transmission using the experimentally determined K_1 and τ as shown in equation 1. The values of K_1 and τ were found to be 1.88 and 0.068 respectively. As can be shown the plant contains a pole at $s = 0$ which means that for any step input there will be perfect steady state tracking which means that we do not need to worry about this when designing the controller.

$$P(s) = \frac{K_1}{s(\tau s+1)} \Rightarrow \frac{1.88}{s(0.068s+1)} \tag{1}$$

Using the previously calculated plant and the given specifications we can use pole placement to design a controller. As shown in equation 2 and 3 we can use the equations for damping ratio, ζ, and natural frequency, ω_n, which can then be used to find the safe area for our poles as shown in Figure 4.

$$\zeta = \frac{-ln(\%Overshoot)}{\sqrt{\pi^2+(ln(\%Overshoot))^2}} = 0.69 \tag{2}$$

$$\omega_n \approx \frac{4}{\zeta T_{settling}} = 11.59 \tag{3}$$

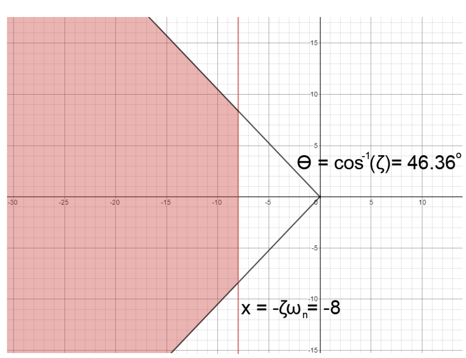

Figure 4: Pole Placement Valid Region

Since the plant's characteristic polynomial is second degree we know we need to select three poles in order to get a first degree controller. The poles we select are the complex conjugate pair $s = -8 \pm 3j$ and a real pole at $s = -50$. We can then utilise the matlab script shown in Appendix B which in turn uses Professor Simpson-Porco's matlab script "pp.m" which automatically solves for the controller given the plant and chosen poles. Utilising these scripts we end up with the controller as shown in equation 4 with a bandwidth frequency of 6.1377 rad/s. From the matlab script in Appendix B we find that this control function satisfies both the settling time and overshoot specifications with values of 0.5 seconds and 1% overshoot respectively.

$$C(s) = \frac{63.12s+1941}{14.71s+754.3} \tag{4}$$

We can then take this function and simulate our control system in simulink as shown in Figure 5. Inputting a step input from -0.7 to 0.7 simulates the gear apparatus moving from -0.7 radians to 0.7 radians, the bounds of our system.

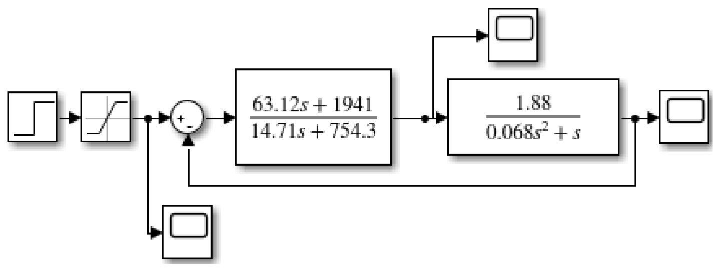

Figure 5: Continuous Time Controller Design Block Diagram

Running this simulation we find the response to be relatively stable as shown in Figure 6 and the control voltage of the motor to be within our saturation points of 6v and -6v as shown in Figure 7.

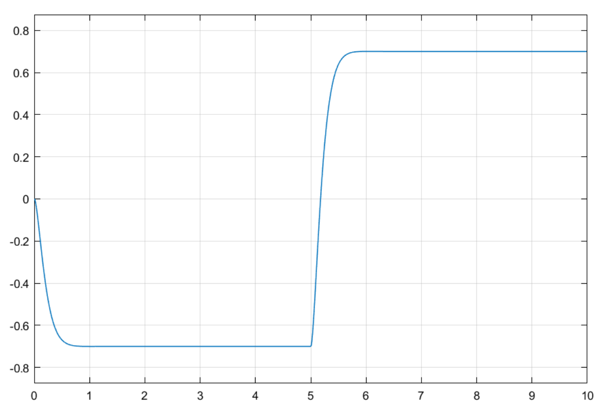

Figure 6: Continuous Time Controller Unit Step Response

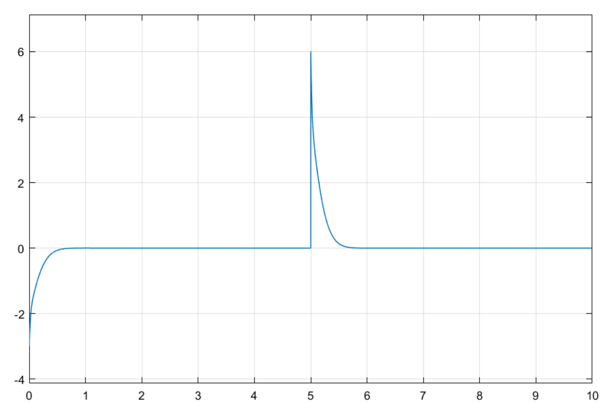

Figure 7: Continuous Time Controller Control Voltage

Since this response seems relatively decent we can continue with discretizing it by running the function through matlab's "c2d" function with a sampling period calculated using the previously mentioned bandwidth frequency. Doing this gives us the discrete time control system as shown in equation 5 with a sampling time of 0.040948 seconds.

$$C[z] = \frac{3.412z - 0.7753}{z + 0.02448} \tag{5}$$

We can then again run this through a similar simulation as shown in Figure 8.

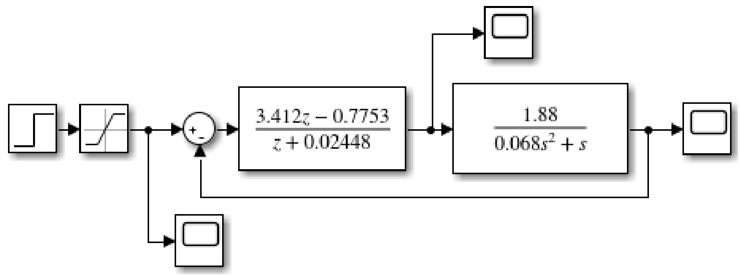

Figure 8: Discrete Time Controller Block Diagram

Running this simulation results in a response similar to the continuous time system with a control input graph also within our motor voltage limits as shown in Figures 9 and 10. This tells us that the discretized controller still performs well enough.

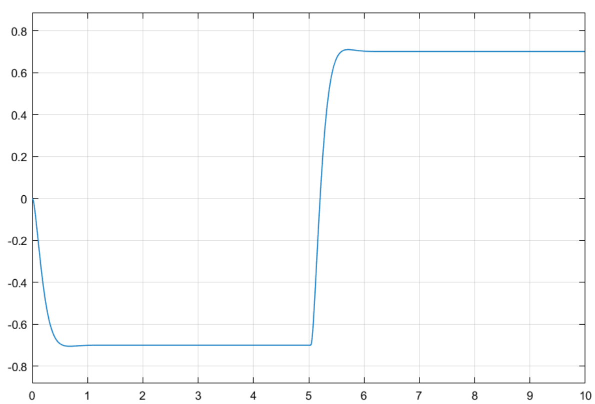

Figure 9: Discrete Time Controller Unit Step Response

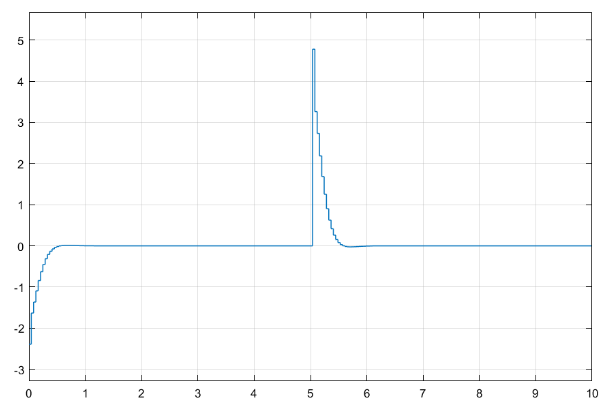

Figure 10: Discrete Time Controller Control Voltage

3.0 Inner Loop Controller Redesign Accounting for Time Delay

From the ECE 484 course notes we know that a discretized signal better represents the original signal with a time delay of half of the sampling time. [2] In order to account for this we need to introduce a time delay into our plant's transfer function, P(s). We can do this by multiplying the plant by the Pade approximation to create an augmented plant, $P_{aug}(s)$, as shown in equation 6 where T is the sampling time, 0.040948 seconds, found previously.

$$P_{aug}(s) = e^{-\frac{T}{2}s}P(s) = \frac{(1-\frac{T}{4}s)}{(1+\frac{T}{4}s)} \cdot \frac{1.88}{s(0.068s+1)} \tag{6}$$

Like we did previously, we can use a pole placement approach to design a controller but this time there will be five poles to choose as the augmented plant is now cubic. By selecting poles at $s = -8 \pm 5j$, $s = -60 \pm 5j$, and $s = -90$ we get the controller as shown in equation 7 which has a settling time of 0.399 seconds and an overshoot of 1.68% which satisfies both the settling time and overshoot requirements.

$$C(s) = \frac{102.7s^2 + 6.561x10^5 s + 1.544x10^7}{1437s^2 + 1.632x10^5 s + 5.27x10^6} \tag{7}$$

In order to verify this controller we must again run a simulation as shown in Figure 11.

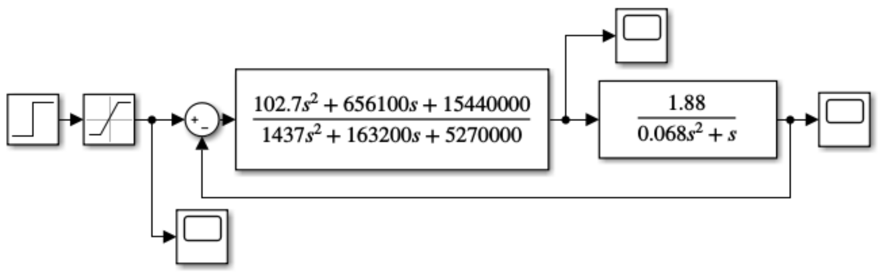

Figure 11: Continuous Time Controller Block Diagram Accounting for Time Delay

Running this simulation results in a relatively decent response as shown in Figure 12 and a control voltage well within our 6V control limit as shown in Figure 13.

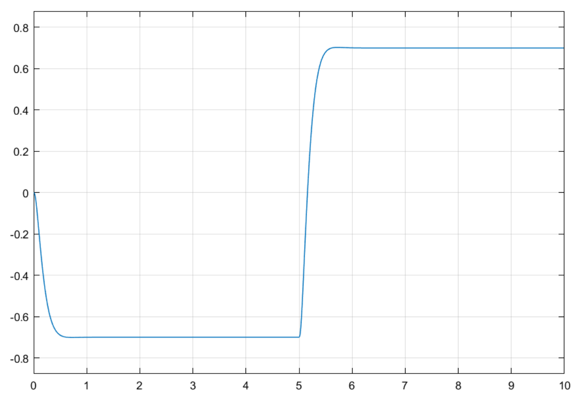

Figure 12: Time Delay Continuous Time Controller Unit Step Response

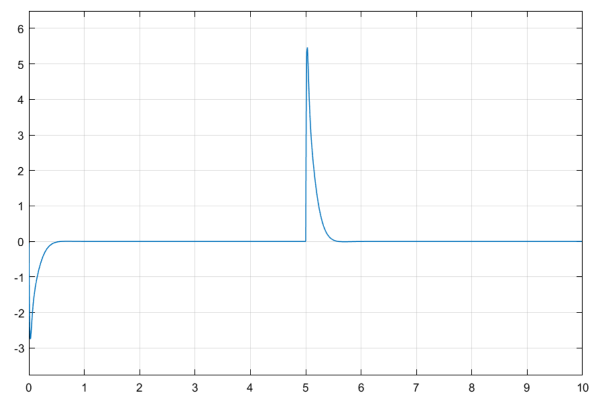

Figure 13: Time Delay Continuous Time Controller Control Voltage

With the controller validated we can continue to discretize it using the bandwidth frequency of 8.6627 rad/s. As a result we get the discrete controller shown in equation 8 with a sampling time of 0.029012 seconds.

$$C[z] = \frac{2.619z^2 + 1.281z - 1.255}{z^2 - 0.1333z + 0.03625} \tag{8}$$

In order to validate our new discrete controller we need to again run a simulation as shown in Figure 14.

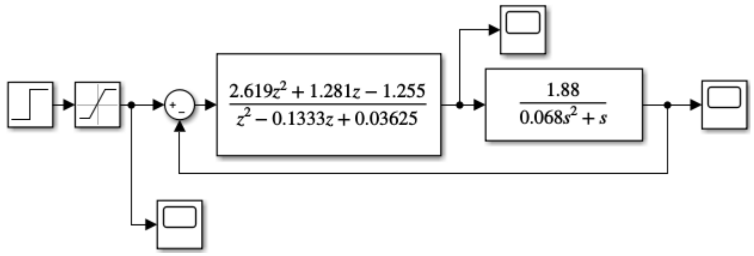

Figure 14: Discrete Time Controller Block Diagram Accounting for Time Delay

As shown in Figure 15 we find that the response of our discrete function is very similar to our continuous time controller and the control voltage is well within our limit as shown in Figure 16.

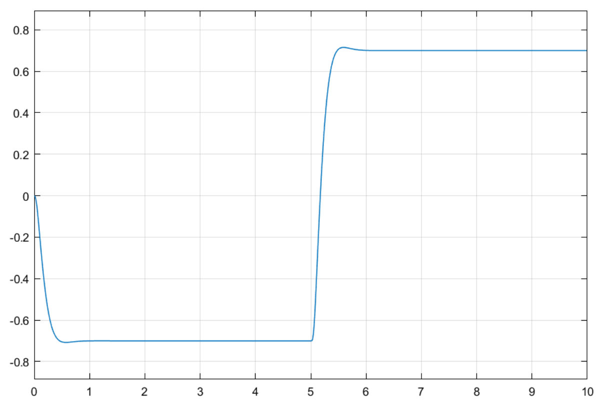

Figure 15: Time Delay Discrete Time Controller Unit Step Response

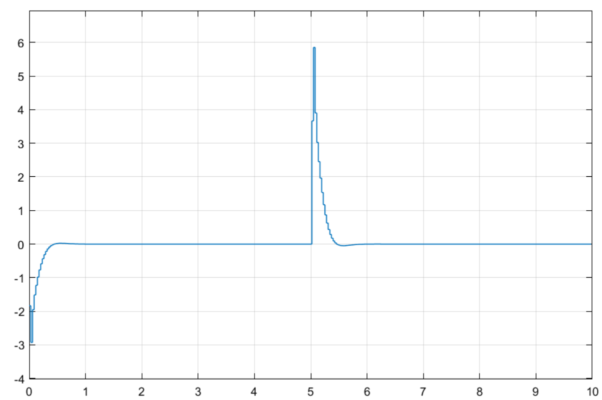

Figure 16: Time Delay Discrete Time Controller Control Voltage

With this, each specification originally laid out has been satisfied with the time delay additionally being accounted for.

4.0 Discretized Inner Loop Controller Implementation

After discretizing the inner loop controller, it was then converted to the discrete time (k) domain using inverse z transforms. Next it was written in the form of a difference equation (as outlined in Appendix 2 of the lab manual) as shown in equation 9.

$$u[k] = 0.133u[k-1] - 0.03625u[k-2] + 2.619e[k] + 1.281e[k-1] - 1.255e[k-2] \qquad (9)$$

This was easily implemented in the labview control code using two shift registers (“control” and “error”) to track previous inputs and outputs of the controller.

The controller was then tested with the beam detached using a unit step input with a magnitude of 0.7 (shown in Figure 17) and compared to the output obtained from the simulated discrete controller (Figure 15 and Figure 16).

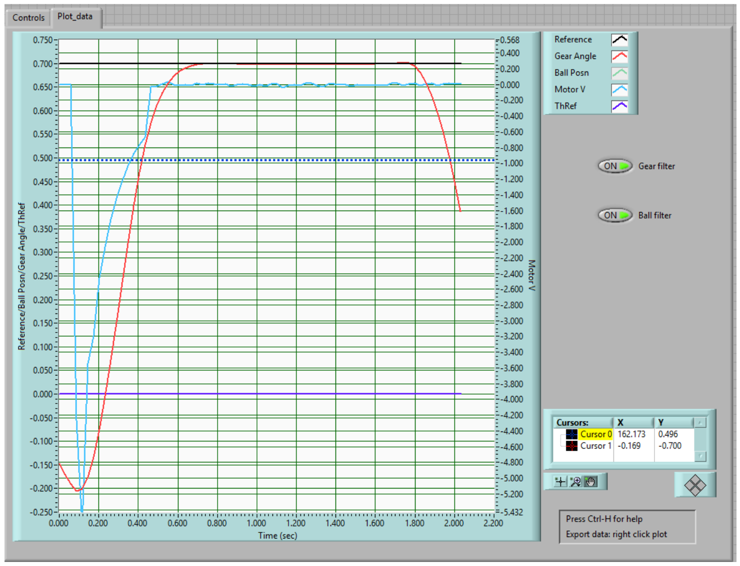

Figure 17: Experimental Step Response of Inner Loop Controller w/o Beam

It was observed that the step response of the simulated system and the response of the tested controller were quite similar. Both responses had a settling time of approximately 0.5s and virtually 0 steady state error. However, the simulated step response had an overshoot of 2.1% while the experimental step response had no overshoot and appeared to be critically damped. This difference is very small and almost negligible and is likely due to inconsistencies in the operation of the motor between when the plant was characterized and when the test was executed.

The motor control signal is also quite similar between both the experiment and the simulation. Both signals observe an initial spike in voltage with an exponential decay. The simulation however had a slightly higher initial spike of 5.8V while the experimental control signal maxed at -5.5V. This difference is likely due to the inconsistency in operation of the motor as mentioned above.

5.0 System Validation with Beam Attached

Once the controller was validated in section 4, the beam was attached to the gear and the test repeated. As expected, there was an increase in the steady state error.

The input was then changed to a square wave so that the steady state tracking error could be measured in both turning directions of the gear. The steady state error was corrected through careful trial and error by incrementally increasing the magnitude of the CCW stiction (to counteract the moment from the weight of the beam on the gear) and vice versa for the CW stiction until the error was minimized.

The initial stiction values for the motor were ± 0.48V. The final stiction values were determined to be 0.25V for CW stiction and -1V for CCW. The steady state error with the new stiction values is shown below in Figure 18.

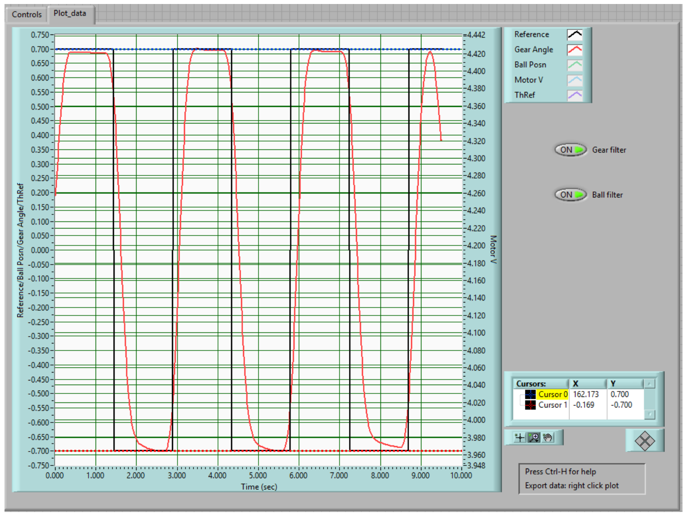

Figure 18: Experimental Square Wave Response of Inner Loop Controller With Beam

6.0 Ball and Beam Position Characterization

In order to convert the ball position potentiometer voltage to cm, the ball was moved in 4cm increments across the length of the beam and the voltage measured and recorded. Linear regression was then used to determine the relationship between the position voltage and the position in cm and the relationship was implemented in the Labview code as shown in Appendix A. The scaling relationship is shown in equation 10.

$$BallPosn\,[cm] = 9.357 * posV\,[V] - 30.56 \tag{10}$$

With $K_{sensor\_scale\_beam}$= 9.375 and $Sensor\_Offset_{beam}$ = -30.56cm.

7.0 Outer Loop Plant Constants Validation

Given by the lab manual, a model for the beam and lever arm are as shown in equations 11 and 12.

$$\frac{\Phi(s)}{\theta(s)} = K_2,\ where\ K_2 = 0.061 \tag{11}$$

$$\frac{Y(s)}{\Phi(s)} = \frac{K_3}{s^2},\ where\ K_3 = 4.78 \tag{12}$$

As the values of K_2 and K_3 are given rather than calculated it is important to ensure that these values are accurate before proceeding.

In order to validate the constant K_2 we must look at the geometry of the experimental setup as shown in Figure 19. As can be seen in Figure 19 the amount of radians the gear moves, θ, is directly proportional to the radians that the lever arm moves, ϕ. Using this knowledge we know that we can use the beam length, L, and the gear radius, r, to calculate the conversion between these angles. Taking our gear radius and dividing it by our beam length gets us a conversion constant of 0.0609, approximately equal to our constant value K_2.

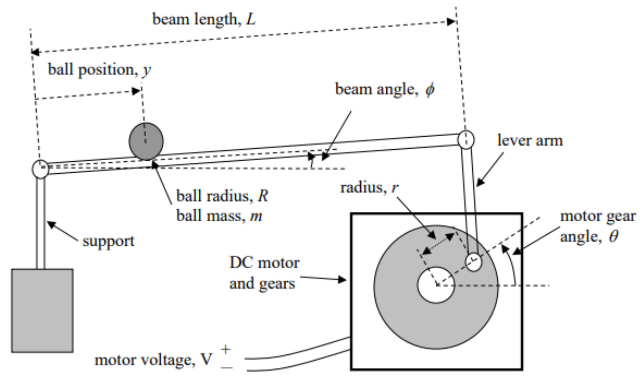

Figure 19: Ball and Beam Apparatus Physical Dimensions [1]

Since we can't geometrically verify the constant K_3 we must instead devise a way to experimentally test it. Since K_3 is a physical characteristic of the ball and beam setup we decided to record the ball's position and movement across the beam at a particular angle. Theoretically if K_3 is correct we should see similar behaviour in a simulated version of the ball and beam. Conducting the physical experiment produces the results as shown in Figure 20. As can be seen we allowed the ball to start rolling at around 0.7 seconds and it reached the end of the beam just before 6 seconds at a gear angle of -0.1 radians.

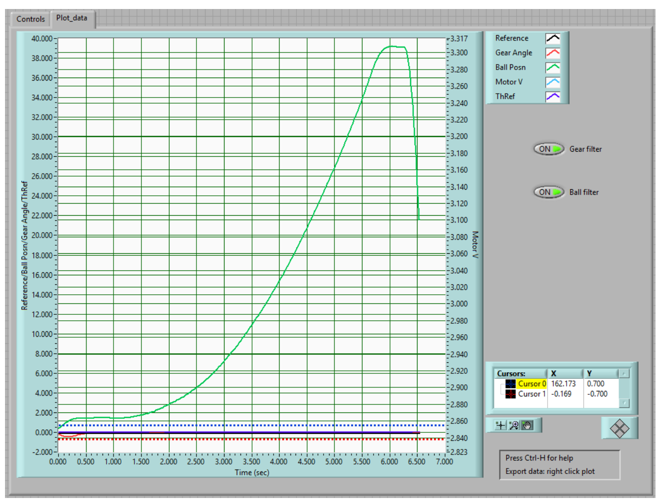

Figure 20: Beam Characteristic Verification Experiment

Next, we can simulate the system as shown in Figure 21 accounting for the start time and gear angle within the step function and the final time within the simulation settings. As a side note the results are multiplied by 100 at the end to get the results in centimeters to be a better comparison for the experimental data.

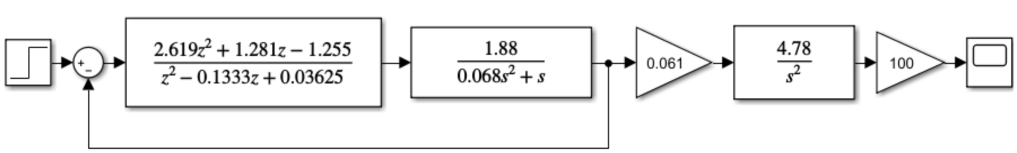

Figure 21: Beam Characteristic Verification Simulation Block Diagram

As can be seen in Figure 22 the simulation produced results very similar to the experimental results. Not only is the curve of the graph the same but the end value is almost the exact same as that of the experimental data. This means that the original value for K_3 is indeed accurate and can be used without being readjusted.

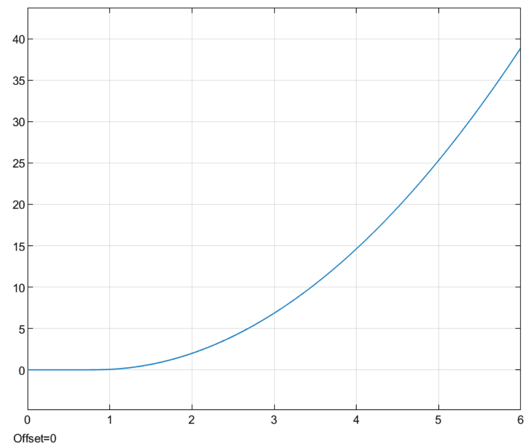

Figure 22: Beam Characteristic Verification Simulation

8.0 Outer Loop Block Diagram

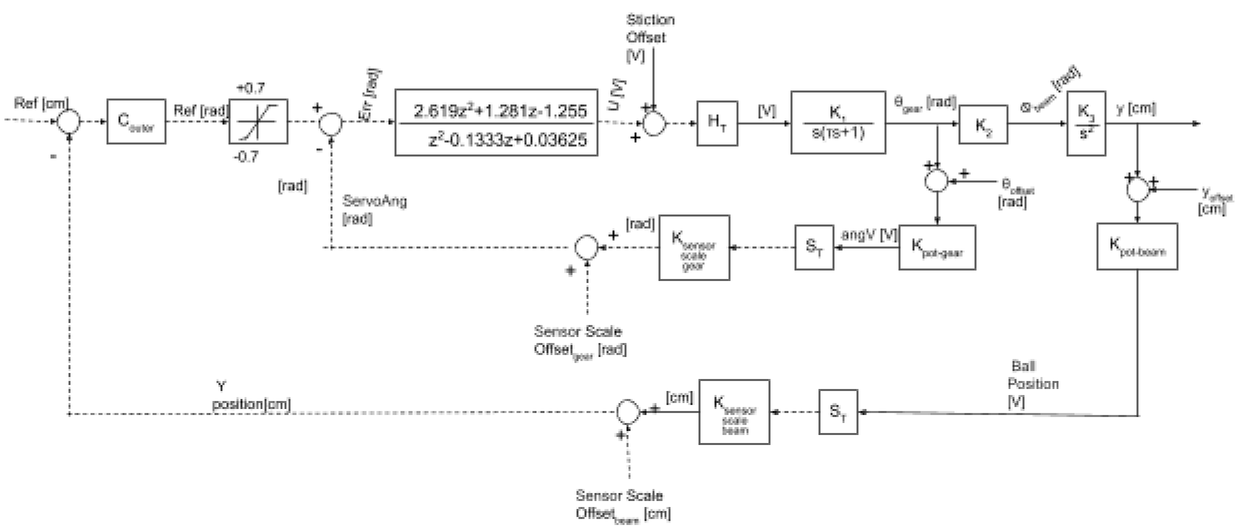

Figure 23: Block Diagram of Control System

Conclusion

In conclusion the inner loop controller was successfully designed and implemented using a pole placement method. The controller was designed to account for sampling delay while still meeting specs outlined within the lab manual as proven in experiments.

In addition to the inner loop controller, the transfer functions for conversion from gear angle to ball position were proven geometrically and experimentally. The relationship of the ball position potentiometer voltage and the actual ball position was also characterized and accounted for in the formula node in Labview. This concludes the steps required to obtain useful feedback for the outer loop controller which will be designed in lab 3.

References

[1] C. Caradima, *Ball and Beam Lab Project Student Handout*. 2019.

[2] J. Simpson-Porco, "ECE 484: Digital Control Applications Lecture Notes", 2019.

Appendix A

Formula Node Code

```
float Temp1;
float eGearAng;
BallPosn = 9.357 * posV - 30.56;
ServoAng = (angV - 6.287) / -0.7334;
if (Loop < 3) {
  u = e = ThRef = posV = angV = ServoAng = BallPosn = 0;
} else {
  if (Manual) {
    u = MotV;
  } else {
    if (ref >= 0.7) {
      ref = 0.7;
    } else if (ref <= -0.7) {
      ref = -0.7;
    }
    error = ref - ServoAng;
    float b0, b1, b2, a0, a1, a2;
    b0 = 1;
    b1 = -0.1333;
    b2 = 0.03625;
    a0 = 2.619;
    a1 = 1.281;
    a2 = -1.255;
    control = -(-b1 * control1 - b2 * control2 + a0 * error + a1 * error1 + a2 *
             error2) / b0;
    u = control;
  }
}
```

Appendix B

Pole Placement and Discretization Script

```
format long
s = tf('s');
t = 0.040948;
p = 1.88/(0.068*s^2+s);
timeDelay = (1-(t/4)*s)/(1+(t/4)*s);
P = p*timeDelay;
%P = p;
Lambda = [-8+5*1i, -8-5*1i, -60+5*1i, -60-5*1i, -90];
%Lambda = [-8+3*1i, -8-3*1i, -50];
C = pp(P,Lambda)
stepinfo(feedback(P*C, 1))
bw = bandwidth(feedback(P*C, 1))
T = 2*pi/(25*bw);
discreteSystem = c2d(C, T, 'tustin')
[y,t] = step(minreal(feedback(P*C,1)));
figure('Position', [300, 100, 800, 395]);
plot(t, y, 'b', 'linewidth', 2);
grid on;
set(gca, 'FontSize', 16);
xlabel('Time (s)','interpreter','latex','FontSize',20);
ylabel('Angular position (radians)','interpreter','latex','FontSize',20);
xlim([0,max(t)])
```

cRIO-9076_RT.vi
N:\GWizFiles\cRIO-9076_RT.vi
Last modified on 2019-10-27 at 5:24 AM
Printed on 2019-10-27 at 5:25 AM