University of Waterloo

Faculty of Engineering

Department of Electrical and Computer Engineering

ECE 481: Lab 3

By Nathaniel Johnston, 20710059

Aug 3rd, 2021

Table of Contents

Lab 3: Design of the Outer-Loop Controller

Introduction

In this final lab report, a controller is developed to control the position of a ball on a beam. To accomplish this, the emulation approach is used to convert a continuous time controller to a discrete time controller. The controller must ensure that the ball does not overshoot the goal by more than 30% and it must stop on target within 6 seconds. In order to design the continuous time controller, intuition on the root locus and how the placement of poles and zeros in the s plane is used to meet the required specifications.

Values From Previous Labs

For reference, the values I calculated for constants from previous labs are shown in table 1. Note that the value for K_3 is the negative of the value I found in lab 2. This is because I erroneously said that the distance y was the 2nd derivative of the acceleration down the incline for positive ϕ. This is not correct because of the way that y was defined in the lab manual. However, this mistake does not really affect the controller design because I just need to take the negative of the controller I design. However, without correcting this error, the physics would not be realistic since the ball would be rolling up hill.

Table 1. Values of important constants from previous labs

K_1	1.7
τ	0.016
K_2	0.05644
K_3	-5.6884

Overall System Plant Simplification

The overall plant transfer function can be calculated by multiplying the inner closed loop transfer functions by the transfer functions from θ to ϕ and from ϕ to y. The calculations are shown below.

$$C_1(s) = \frac{2s + 160}{s + 32}$$

$$P(s) = \frac{1.7}{0.016s^2 + s}$$

$$P_{ov} = \frac{C_1(s) * P(s)}{1 + C_1(s)P(s)} * 0.05644 * \frac{-5.6884}{s^2}$$

Where $P_{ov}(s)$ is the overall plant, $C_1(s)$ is the inner loop controller, and P(s) is the motor plant. The 0.05644 value is the transfer function from θ to ϕ and $-5.6884/s^2$ is the transfer function from ϕ to y.

This plant transfer function was calculated using Matlab and was found to be the following.

$$P_{ov} = -68.224 \frac{s(s + 80)(s + 62.5)(s + 32)}{s^3(s + 64.14)(s + 62.5)(s + 32)(s^2 + 30.36s + 265.1)}$$

Which has the following root locus plot (Note that it was plotted without the negative factor in the transfer function since root locus methods are for positive gains)

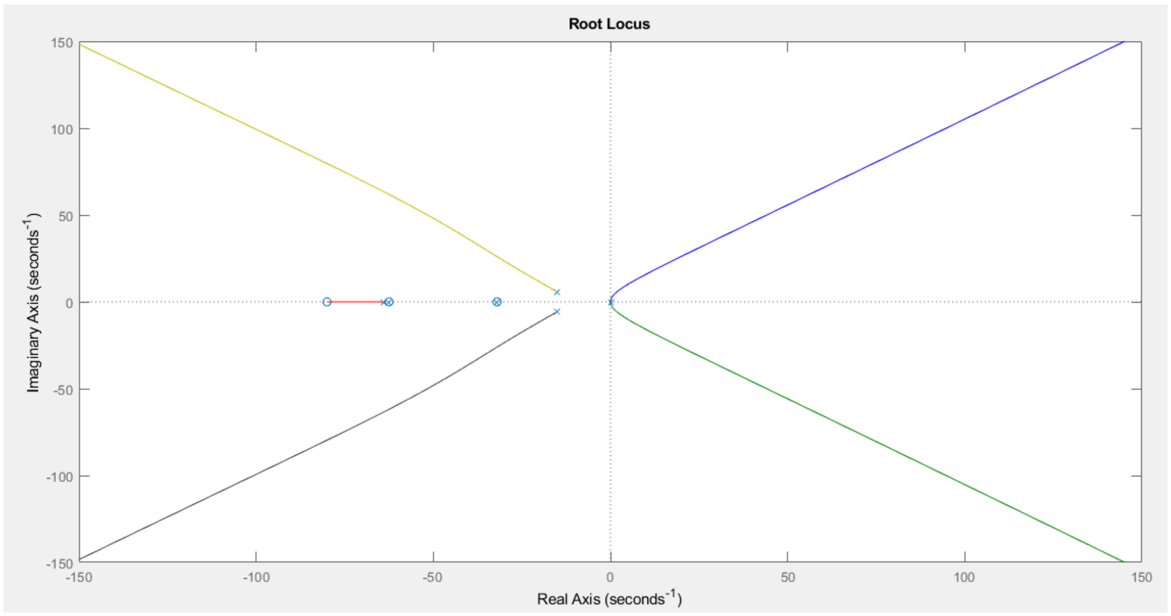


Figure 1. Root locus of unsimplified overall plant

There are some obvious cancellations here, so the transfer function then becomes

$$P_{ov} = -68.224\frac{(s+80)}{s^2(s+64.14)(s^2+30.36s+265.1)}$$

In order to simplify the controller design. There are further simplifications that can be made. Since the pole at s = -64.14 and the zero at s = -80 are so far left, they have very little effect on the overall system since they are so fast. The transfer function can then be approximated as

$$P_{ov} \approx \frac{-68.224}{s^2(s^2+30.36s+265.1)}$$

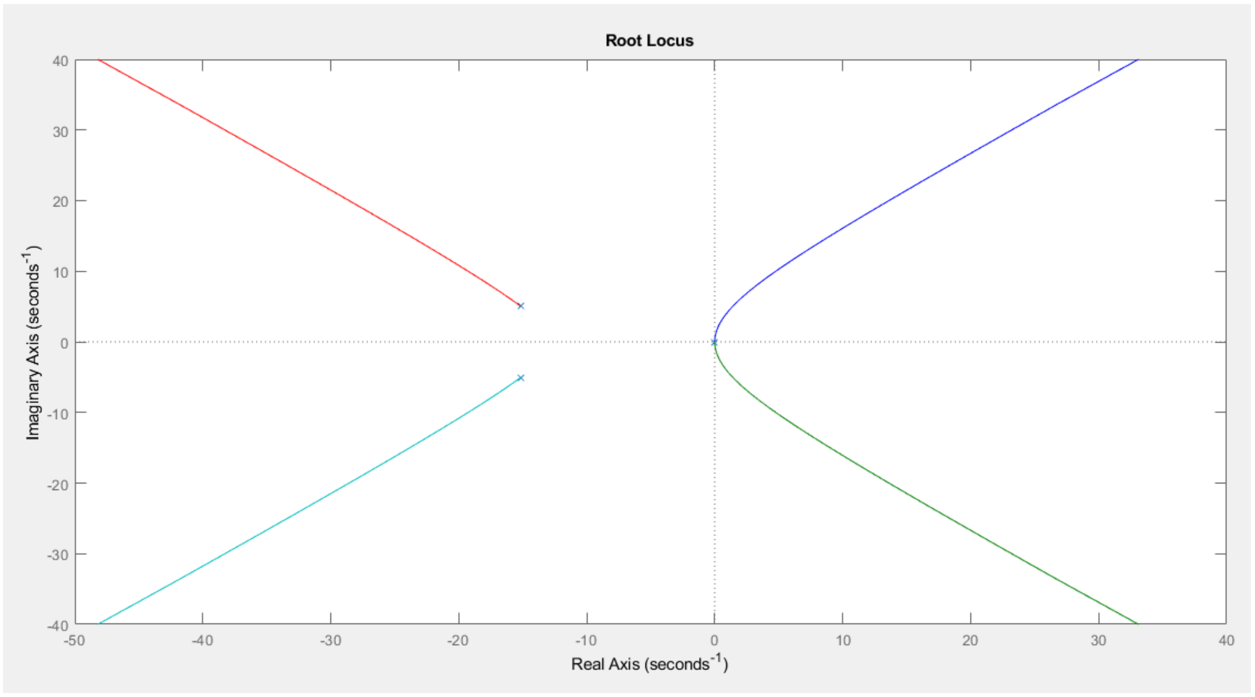


Figure 2. Root locus of simplified overall plant

Clearly this transfer function is unstable for all gains.

Continuous Time Controller Design

Since this plant is unstable for all gains, a proportional controller is immediately ruled out.

In order to design the controller, I kept the following guidelines in mind:

1. Any type of derivative control will cause the system to be unstable because it introduces a zero at s = 0 creating an unstable pole-zero cancellation
2. At least two zeros need to be added in the open left half plane in order to pull the unstable poles in to the left
3. If M zeros are added, at least M poles must be added to make the controller causal. These should be placed far to the left so that they are fast and do not greatly change the transient response
4. Integral control is unnecessary due to the two poles at s = 0 which already provide the integral action necessary for perfect steady state tracking of step signals
5. In order to minimize overshoot, the system complex poles should be kept relatively close to the real axis
6. To make the system settle fast enough, the system poles should be moved as far left as possible without causing too much overshoot and while keeping the system in the stable region

With these guidelines in mind, I picked zeros at s = 1.8 and s = 14.5 to pull in the unstable poles. I noticed the system was stable for a gain of roughly 0.7 and gave a reasonable compromise between overshoot and settling time. With this configuration, I found the gain crossover frequency to estimate the bandwidth of the system and found it to be ~3.62 rad/s. I then placed the poles at s = 40 to satisfy the rule of thumb that the time constant should be less then 1/(10*BW) where BW is the bandwidth of the system. However, I found that these poles still were not fast enough to leave the rest of the system unaffected. I ended up choosing to place the poles at s = 53 to ensure they did not interfere.

As a sanity check, with those poles in place, the bandwidth of the system is still roughly 3.6 rad/s. With poles at s = 40, the bandwidth was ~5.8 rad/s meaning those poles affected the system transient response in an undesired way and were too slow.

This led to the following controller.

$$C_2(s) = -2000\frac{(s+1.8)(s+14.5)}{(s+53)^2}$$

The root locus plot for the controller and overall plant are shown in figure 3.

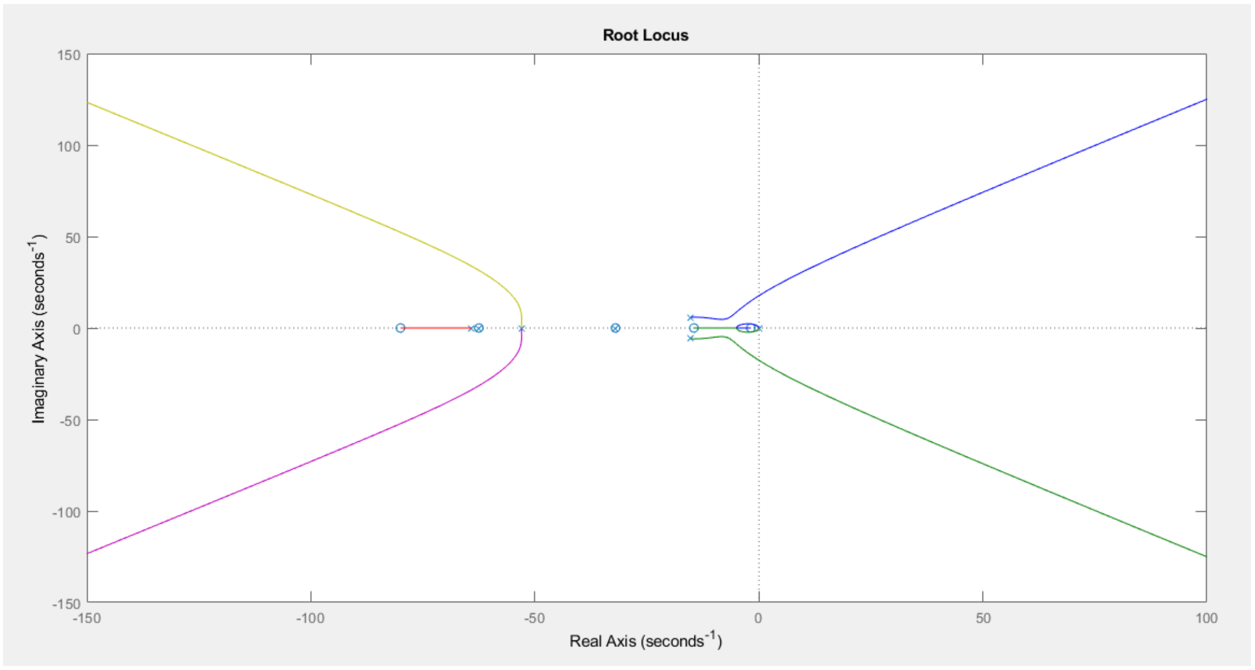


Figure 3. Root locus of system with continuous time controller

The important section is close to s = 0 since it shows the previously unstable regions and at the gain I have chosen, this area determines the overshoot and settling time for this system. This region is shown in figure 4.

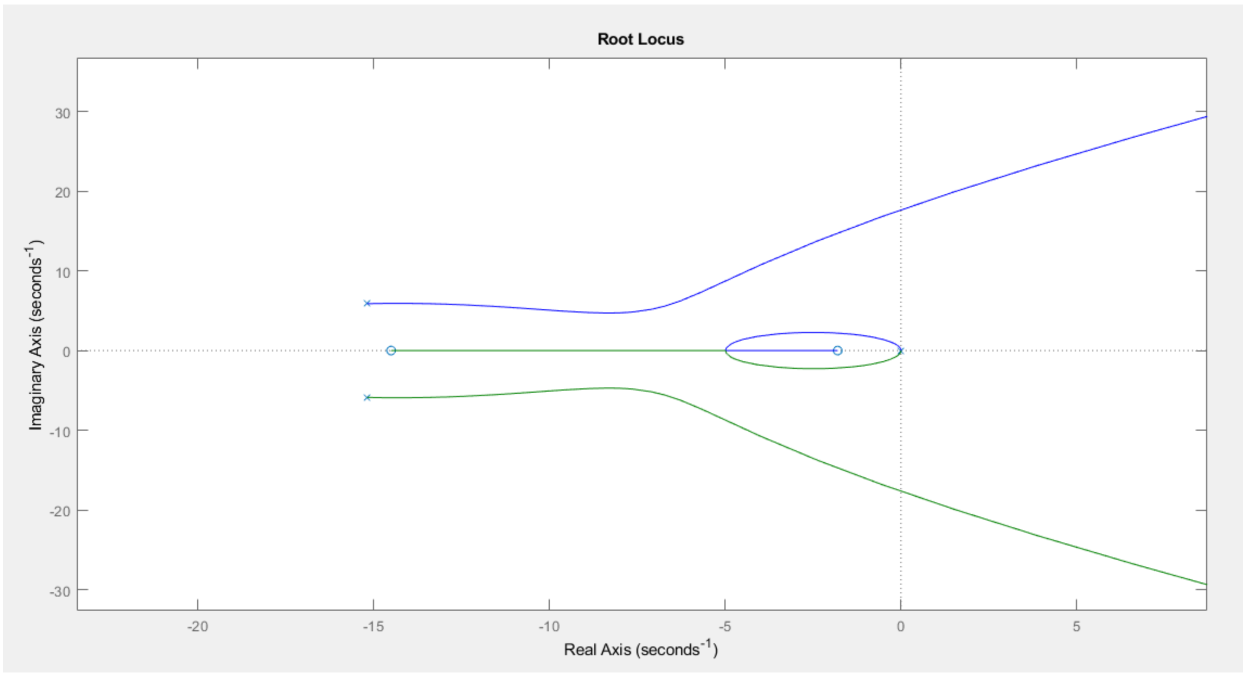


Figure 4. Root locus of system with continuous time controller (zoomed in)

This controller also performs quite well, as shown in figure 5. The settling time is well below 6 seconds and there is very little overshoot. The actual values are 5.07% overshoot and 2.704 second settling time. Thus, meeting the design requirements.

One thing to note is this controller may not work in reality because of the sharp spike in theta. This would mean that the motor has to move very fast which it might not be able to do in reality. Also when deriving the transfer function from ϕ to y, it was assumed that ϕ does not change so rapidly. With a rapidly changing Θ and thus a rapidly changing ϕ, the ball may bounce around and give undesired behaviour. However, if the motor cannot move this fast, it would not cause the ball to bounce around as much which may be ok. Without the actual system to test on, it is difficult to solve the issue so the controller will be left as is.

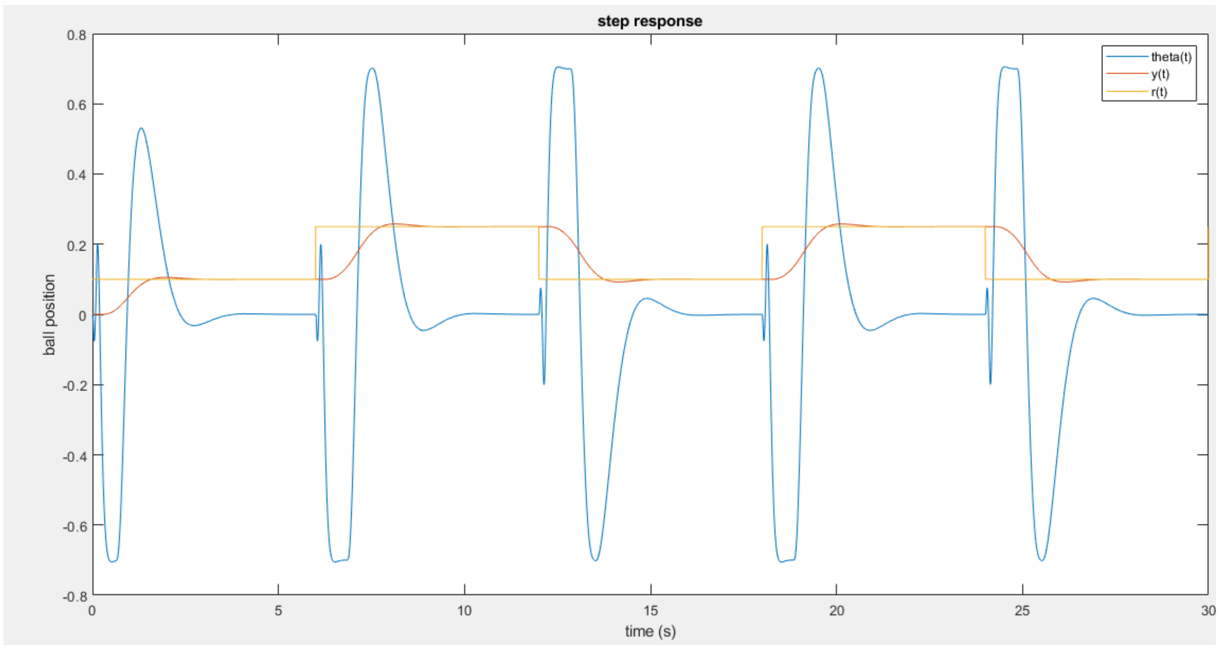


Figure 5. Continuous time controller performance

Discrete Time Controller Design

In order to design the discrete time controller. I ensured that the controller worked well with the T/2 delay in the plant. I used T = 0.01 since this was the time period I used for the inner loop controller. It also should work well as a sampling frequency since the system bandwidth is roughly 3.6 rad/s (roughly 0.573 Hz) and the sampling frequency (100 Hz)is much higher than this, meaning it is more than sufficient. With this value for T, the controller seemed to stabilize the system with the T/2 delay and provide similar performance so I discretized my controller at this sampling period using the SCH rule. However, I found that while this system worked, it resulted in a slower response and more unwanted oscillations in θ.

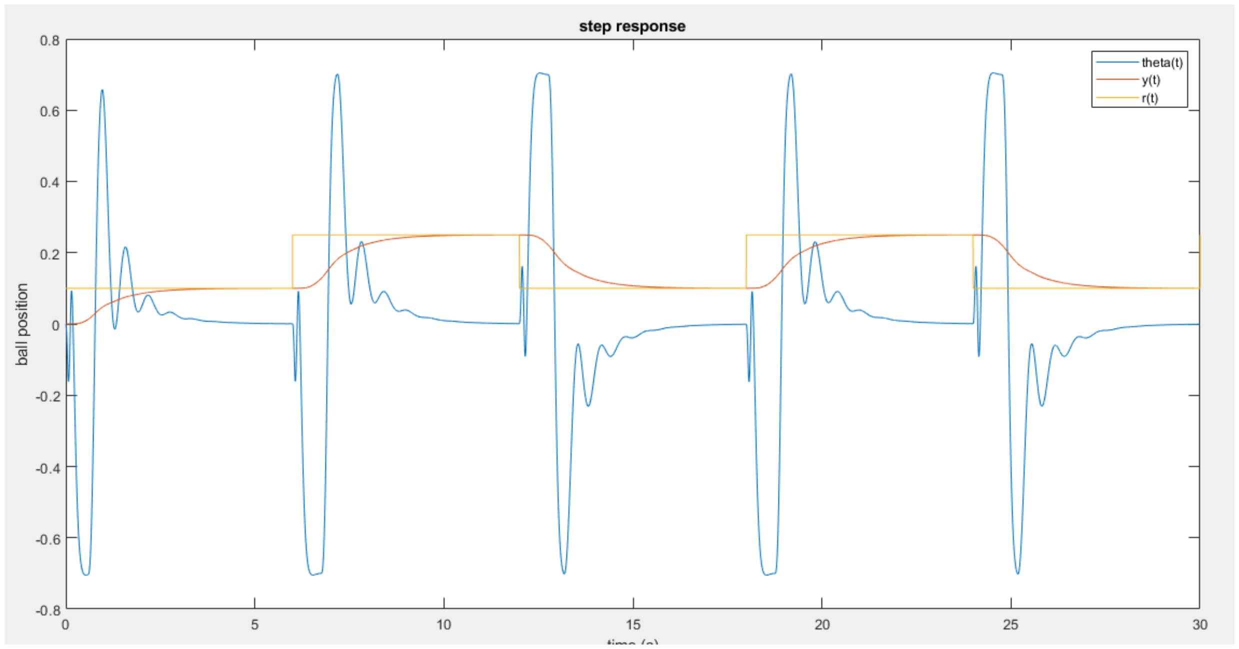


Figure 6. Discretized controller performance using SCH rule

I then tried changing the discretization method to the trapezoidal rule and received a much nicer response that was more similar to the continuous time performance.

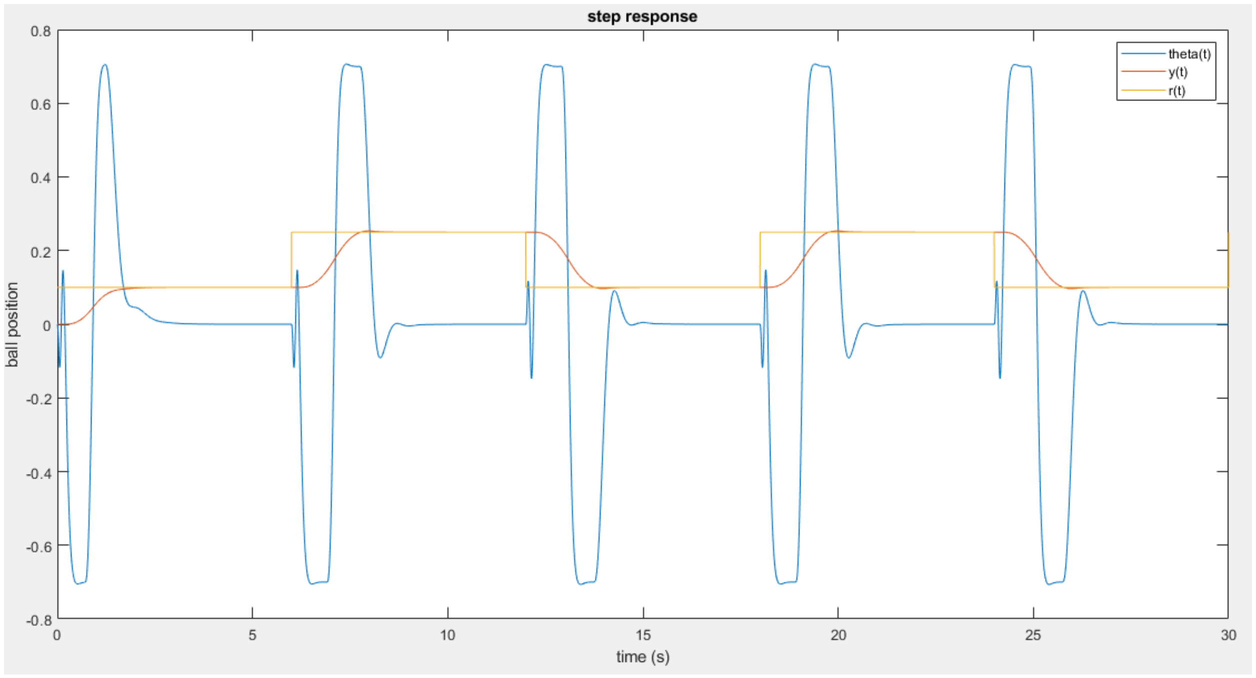


Figure 7. Discretized controller performance using trapezoidal rule

With this controller, the overshoot is 2.2% and the settling time is 2.046 seconds which is an even better response than the continuous time controller.

The final discretized controller used is shown below.

$$D_2[z] = -1503\frac{(z-0.9822)(z-0.8648)}{(z-0.6667)^2}$$

Expanded out, this is equivalent to

$$D_2[z] = -\frac{1503z^2 - 2776z + 1277}{z^2 - 1.3333z + 0.4444}$$

This form can be used to write out the time domain expression for this controller which would be used to implement the controller in the real experiment.

I will define y_{ref} to be the reference position signal and θ_{ref} to be the output of the controller. The transfer function of the controller is simply $\Theta_{ref}[z]/Y_{ref}[z]$.

$$\Theta_{ref}[z] * (z^2 - 1.3333z + 0.4444) = -Y_{ref}[z] * (1503z^2 - 2776z + 1277)$$

Taking the inverse z transform:

$$\theta_{ref}[k+2] - 1.3333\theta_{ref}[k+1] + 0.4444\theta_{ref}[k] = -1503y_{ref}[k+2] + 2776y_{ref}[k+1] - 1277y_{ref}[k]$$

$$\theta_{ref}[k] = 1.3333\theta_{ref}[k-1] - 0.4444\theta_{ref}[k-2] - 1503y_{ref}[k] + 2776y_{ref}[k-1] - 1277y_{ref}[k-2]$$

Overall System Block Diagram

The block diagram for the entire closed loop system is shown in figure 8. It includes the inner loop to control the motor angle, and the outer loop to control the ball position. There is a relay in the inner loop for stiction compensation and a saturator after the outer loop controller to limit the reference motor angle between -0.7 and 0.7.

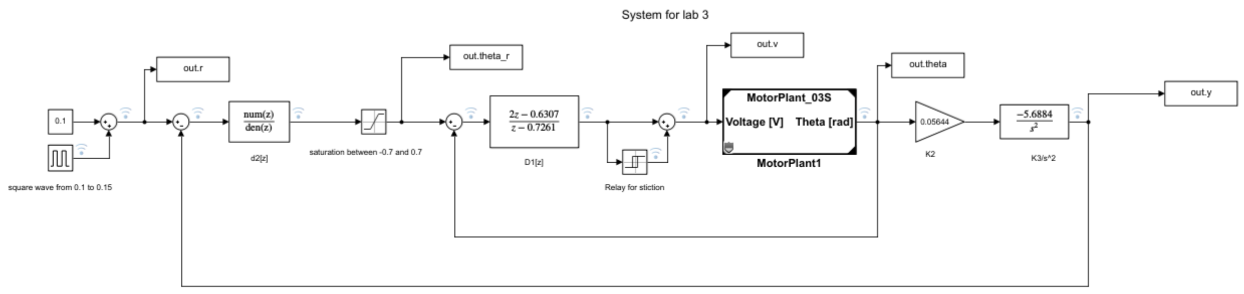
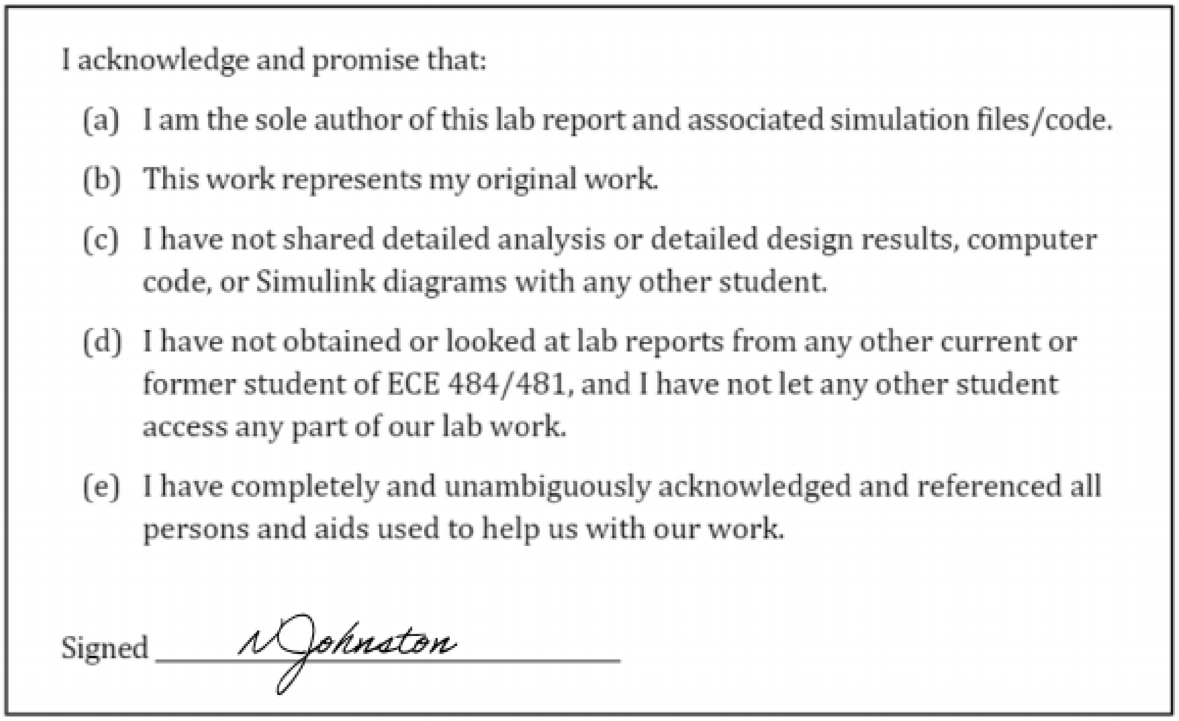


Figure 8. Overall system block diagram

Conclusion

In order to control the position of a ball on a beam, nested control loops in a sampled data system were used. The inner control loop was used to control the angle of the motor which moves the beam system. The outer loop is used to control the actual position of the ball. Using this configuration, controllers were designed to accurately place the ball in the desired position quickly and with minimal overshoot. These controllers were designed using the emulation method to first design a continuous controller and then discretize it.



I acknowledge and promise that:

(a) I am the sole author of this lab report and associated simulation files/code.

(b) This work represents my original work.

(c) I have not shared detailed analysis or detailed design results, computer code, or Simulink diagrams with any other student.

(d) I have not obtained or looked at lab reports from any other current or former student of ECE 484/481, and I have not let any other student access any part of our lab work.

(e) I have completely and unambiguously acknowledged and referenced all persons and aids used to help us with our work.

Signed N Johnston