Mamadou Kaba

27070179

ELEC 372 CN-X

23/07/2024

30/07/2024

Mahshid Rahimifard

Objective

The objectives of this lab are to investigate methods for enhancing system performance using PD, PI, and PID control techniques. We aim to study the implementation and effects of these control strategies on system stability and transient response. Additionally, we will examine how to eliminate steady-state errors caused by disturbance inputs and optimize control parameters to achieve desired performance. This lab will provide hands-on experience with tuning control systems and understanding the theoretical underpinnings of these adjustments.

Theory

In control systems, PID (Proportional-Integral-Derivative) control is a widely used method for improving system performance. The PID controller adjusts the control input based on the error signal, which is the difference between the desired and actual outputs. Proportional control (P) adjusts the control effort in direct proportion to the error signal. Integral control (I) accumulates the error over time, aiming to eliminate steady-state error by increasing the control effort when there is a persistent error. Derivative control (D) predicts future error based on its rate of change, providing a damping effect that enhances system stability and reduces overshoot.

Proportional-Integral (PI) control combines the proportional and integral actions to improve both transient and steady-state performance. By adding an integral term, the system's type number increases, reducing steady-state error. The transfer function of a PI controller introduces a pole at the origin and a zero, which can be tuned to minimize its impact on transient response while improving steady-state accuracy.

Proportional-Integral-Derivative (PID) control further enhances system performance by incorporating a derivative term. This addition helps counteract the increase in system order due to integral control, thereby stabilizing the system and reducing overshoot. The transfer function of a PID controller includes terms for proportional, integral, and derivative actions, making it highly versatile and effective for a wide range of control applications.

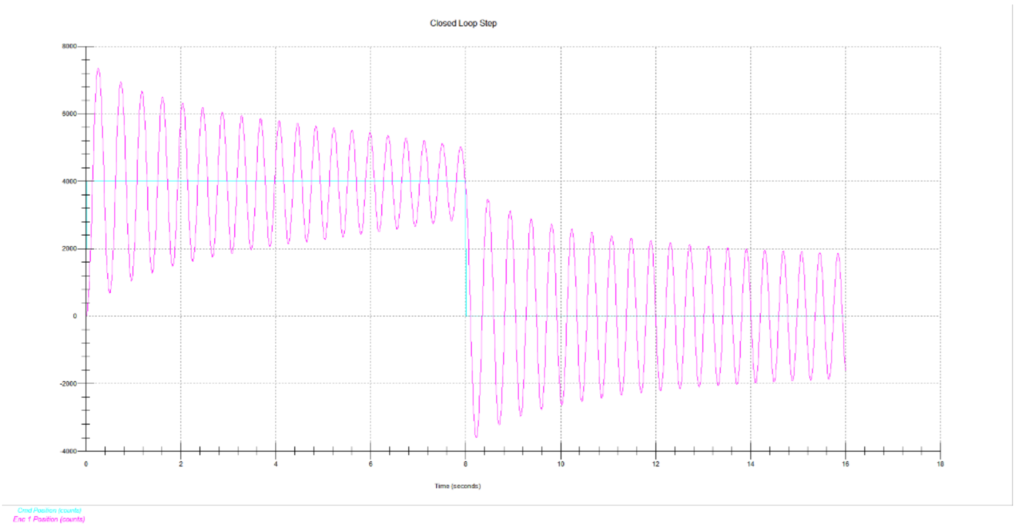
In this lab, we will implement and compare PD, PI, and PID controllers. We will examine their effects on system stability, transient response, and steady-state error. By tuning the control parameters, we aim to achieve optimal performance and understand the trade-offs involved in each control strategy.

Tasks/Results/Discussion

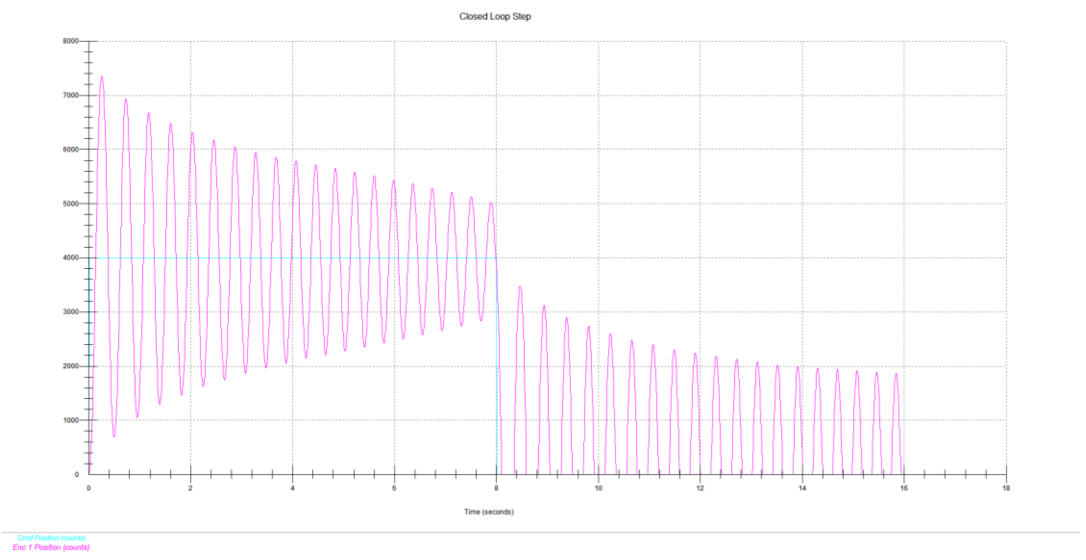
Task 5.3.1: PI Control

1. **Reset the Controller:** The controller was reset from the Utility menu. The PID configuration was implemented with Ki=0, Kd=0, and Kp=0.2. A closed-loop step response input was set up using a step size of 4000 Counts, a dwell time of 8000 ms, and 1 repetition. The step test was executed, and values for the overshoot and the damped natural frequency ωd were determined using axis scaling**.**
2. **Disturbance Configuration:** From the Command menu, 'Disturbance configuration' was selected, and Viscous Friction was set to 4 volt-sec/radian. The "Include Viscous friction" box was checked, and the step test was executed again. The offset error in the resulting step response was measured.
3. **Incremental Integral Feedback:** The controller was re-implemented with Ki set to successive values from 0.02 to 0.1 in steps of 0.02. The step test was executed for each value, and the Ki value that minimized the steady-state error to within 5% was selected and noted. Observations for each test were recorded.

Results and Observations



Plot 1: Full closed-loop Plot of step 5.3.1.1(PI -Control)

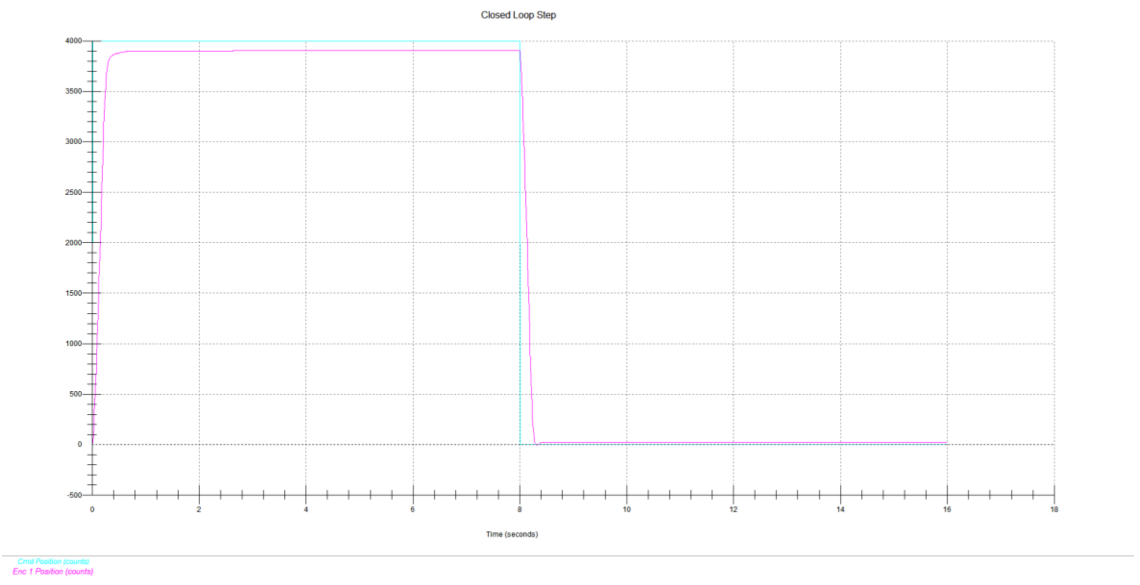


Plot 2: Positive Overshoot Plot of step 5.3.1.1

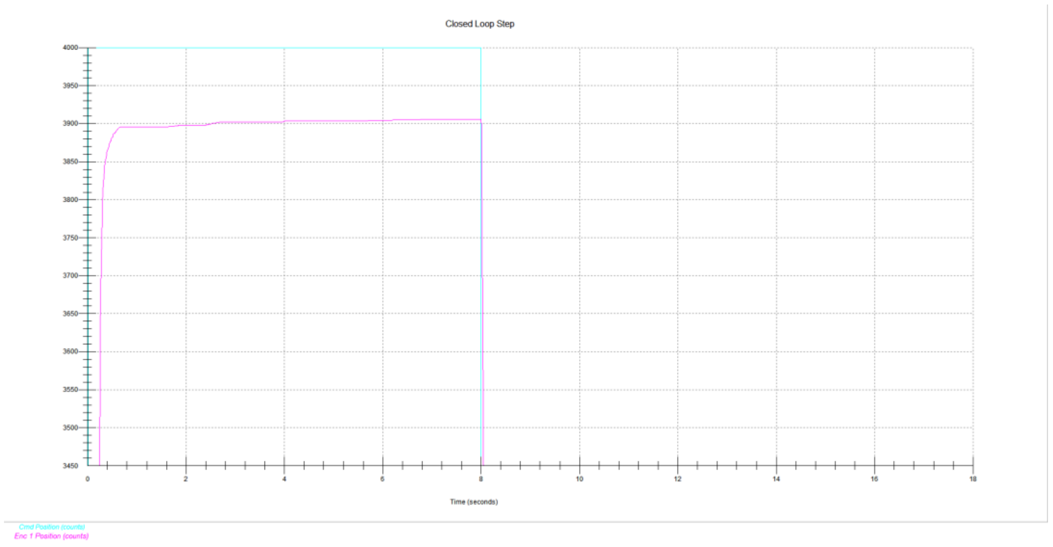
From the positive overshoot plot: $\%OS = \frac{7000-4000}{4000} \times 100 = 75\%$

Period (T) = 3.8 -1.6 = 2.2 sec

$$\omega d = \frac{2\pi}{T} = \frac{2\pi}{2.2} \approx 2.86\, rad/s$$



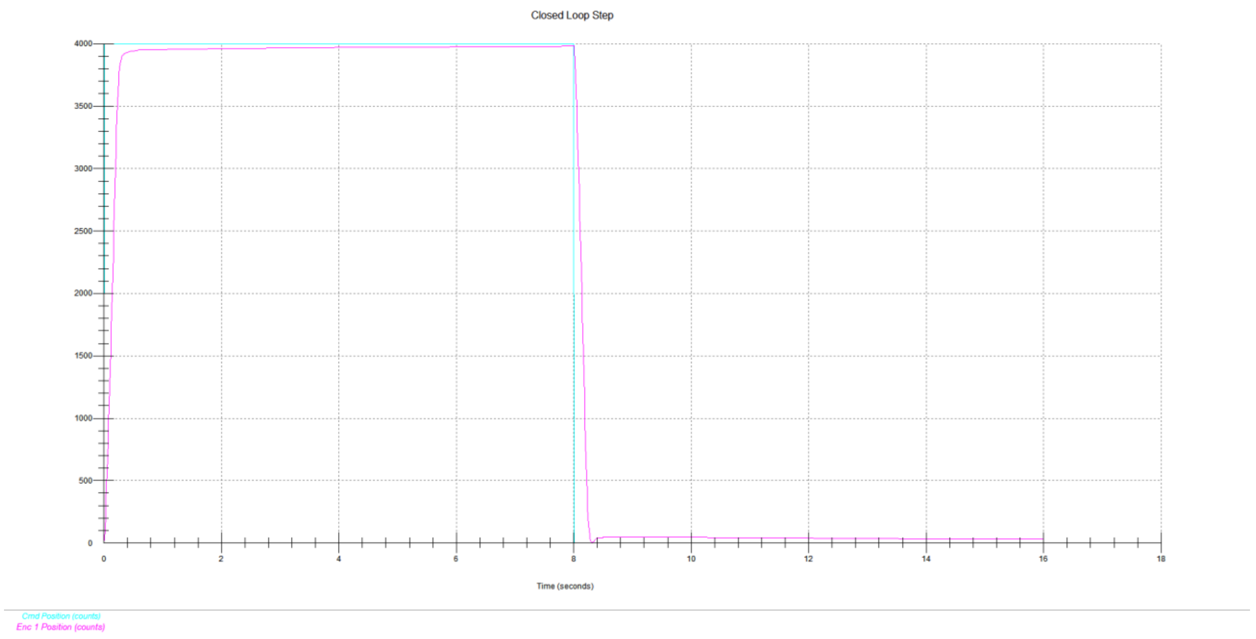
Plot 3: Plot with Viscous Friction included of step 5.3.1.2



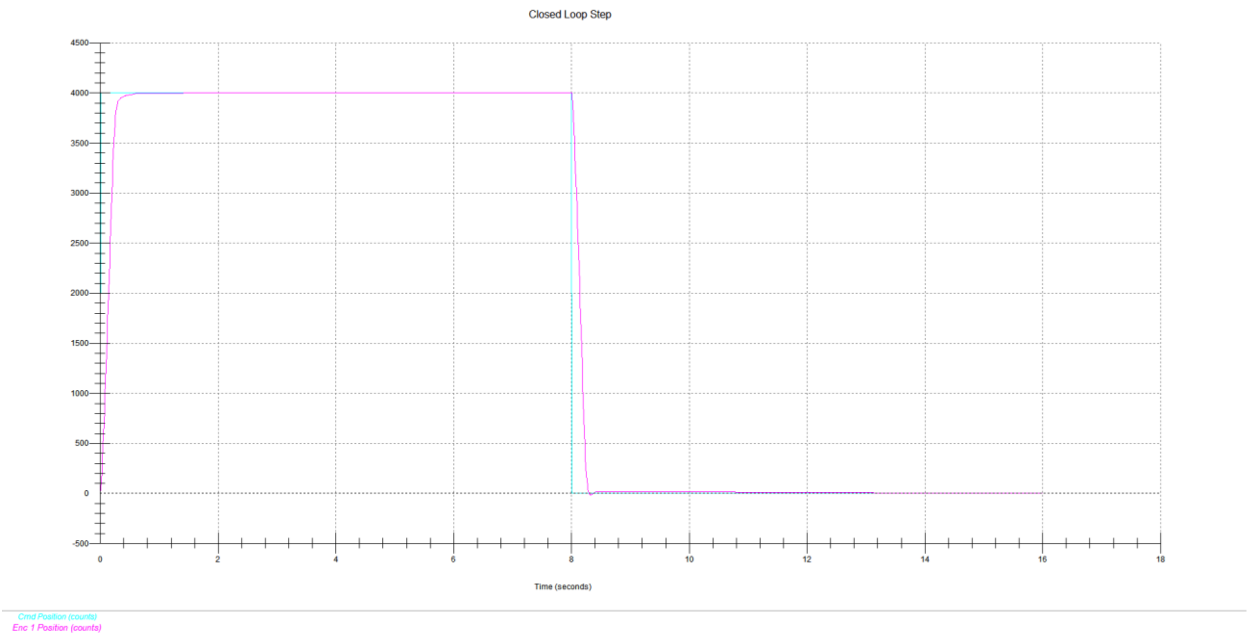
Plot 4: Zoomed in plot of step 5.3.1.2

Offset Error (e_{ss}) = Commanded Position - Steady State Value

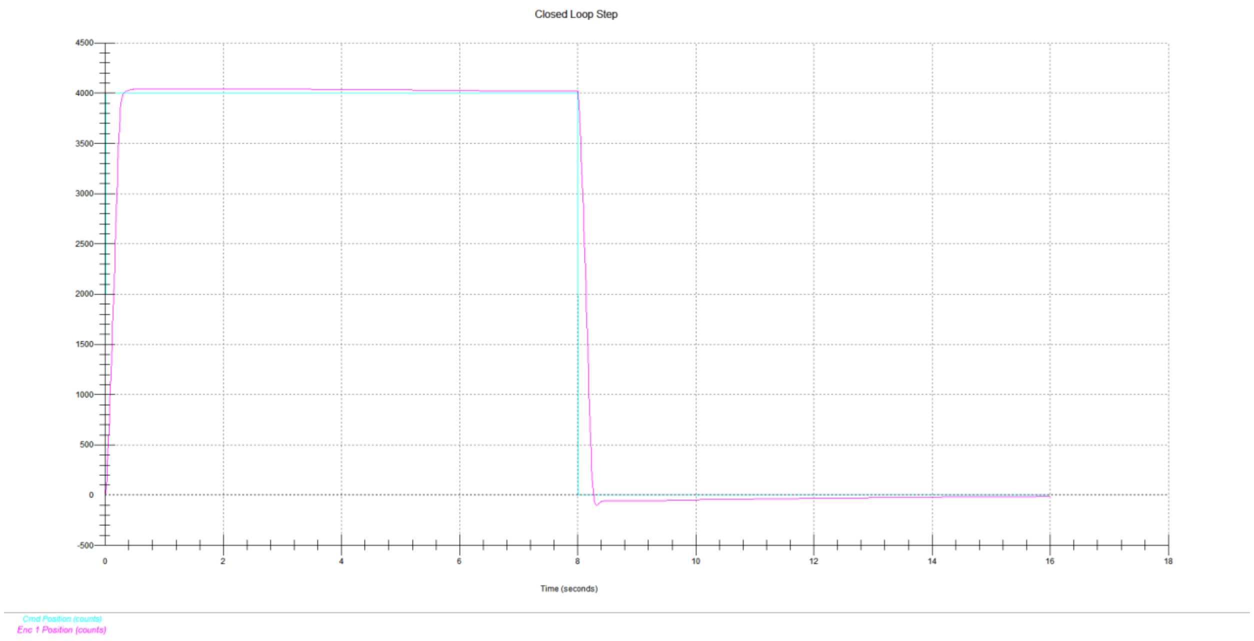
$e_{ss} = 4000 - 3870 = 130$ counts



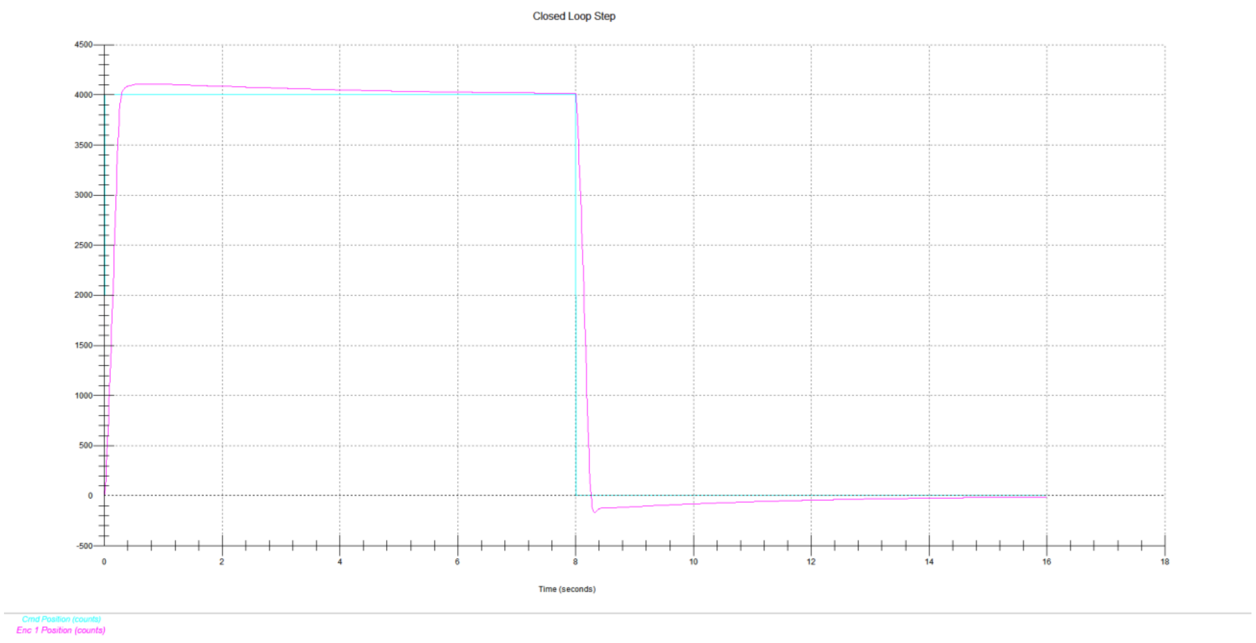
Plot 5: Controller with Ki set to 0.02 for step 5.3.1.3



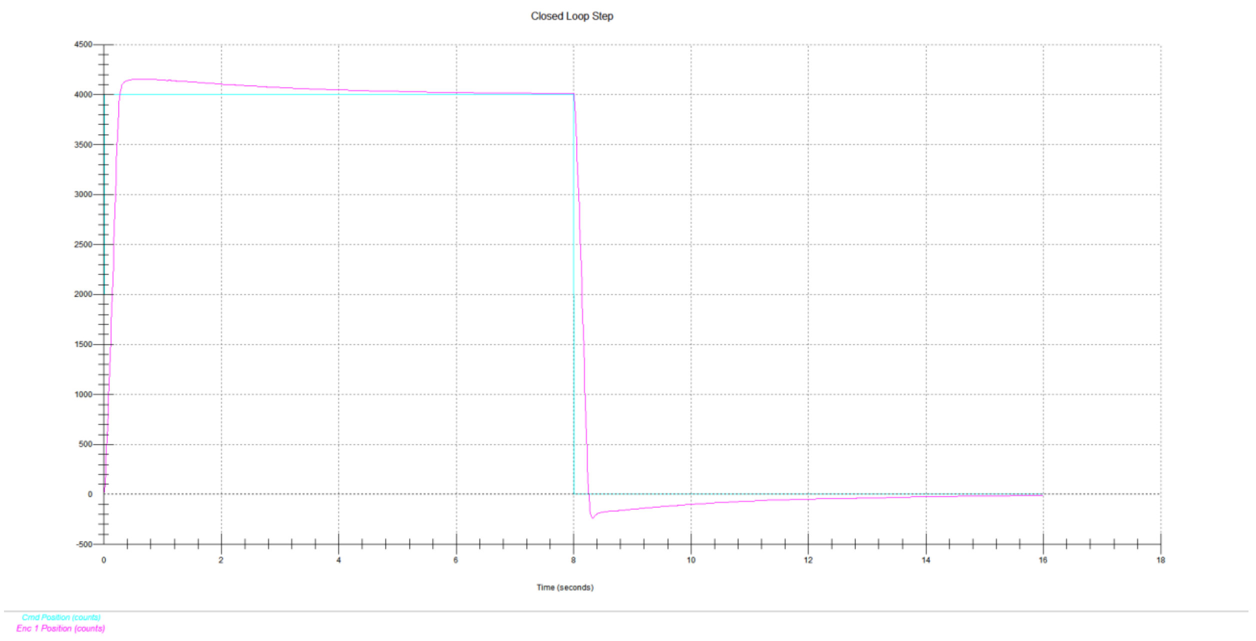
Plot 6: Controller with Ki set to 0.04 for step 5.3.1.3



Plot 7: Controller with Ki set to 0.06 for step 5.3.1.3



Plot 8: Controller with Ki set to 0.08 for step 5.3.1.3



Plot 9: Controller with Ki set to 0.1 for step 5.3.1.3

- Ki = **0.02**: The steady-state error is still noticeable.
- Ki = **0.04**: The steady-state error is reduced but still present.
- Ki = **0.06**: The steady-state error is further reduced.
- Ki = **0.08**: The steady-state error is minimized to an acceptable value within 5%.
- Ki = **0.10**: The steady-state error is minimized and stable.

The value of Ki = 0.08 provides the best reduction in steady-state error to within an acceptable value of 5%, with the system remaining stable. This is considered the optimal Ki setting for minimizing steady-state error in this experiment.

1) Effect of Increase in Ki on Offset Error and Overshoot

Offset Error:

- Increasing Ki helps to reduce the steady-state error or offset error. This is because the integral action of the PID controller accumulates the error over time and applies a correction based on the accumulated error. As Ki increases, the system becomes more aggressive in correcting any deviation from the setpoint, which results in a smaller steady-state error.

Overshoot:

- However, increasing Ki can also lead to an increase in overshoot. The integral action can cause the system to become more oscillatory and less stable if Ki is too high. This is because the correction applied by the integral action can cause the system to exceed the setpoint, resulting in overshoot. The system may oscillate around the setpoint before settling, especially if the proportional and derivative gains are not appropriately tuned to compensate for the increased integral action.

2) Derive the Closed-Loop Transfer Function (CLTF) for the PID System and Obtain the Relation Between Coefficients Required for Stability

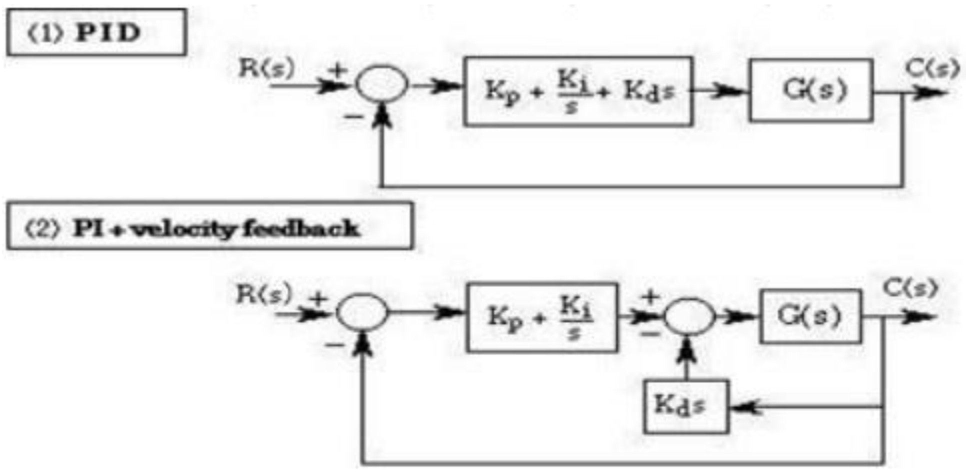


Figure 5.1: Closed-loop Configuration

Given the standard form of a PID controller:

$Gc(s) = Kp + Ki/s + Kd * s$

And the plant transfer function:

$G(s) = 1/(Js^2 + Bs)$

The open-loop transfer function is:

$G\_OL(s) = Gc(s) * G(s) = (Kp + Ki/s + Kd * s) * 1/(Js^2 + Bs)$

The closed-loop transfer function is:

$G\_CL(s) = G\_OL(s) / (1 + G\_OL(s))$

We substitute G_OL(s):

$G\_CL(s) = (Kp * s^2 + Ki + Kd * s^3) / ((J + Kd) * s^3 + (B + Kp) * s^2 + Ki)$

For the system to be stable, the characteristic equation D(s) of the denominator should not have any roots with positive real parts. The characteristic equation is: $D(s) = (J + Kd) * s^3 + (B + Kp) * s^2 + Ki = 0$

According to the given stability criterion for a cubic equation $a * s^3 + b * s^2 + c * s + d = 0$, no root will have a positive real part if the condition $b * c > a * d$ is satisfied. Here:

- $a = J + Kd$
- $b = B + Kp$
- $c = Ki$
- $d = 0$

So the condition becomes: $(b * c > a * d)$

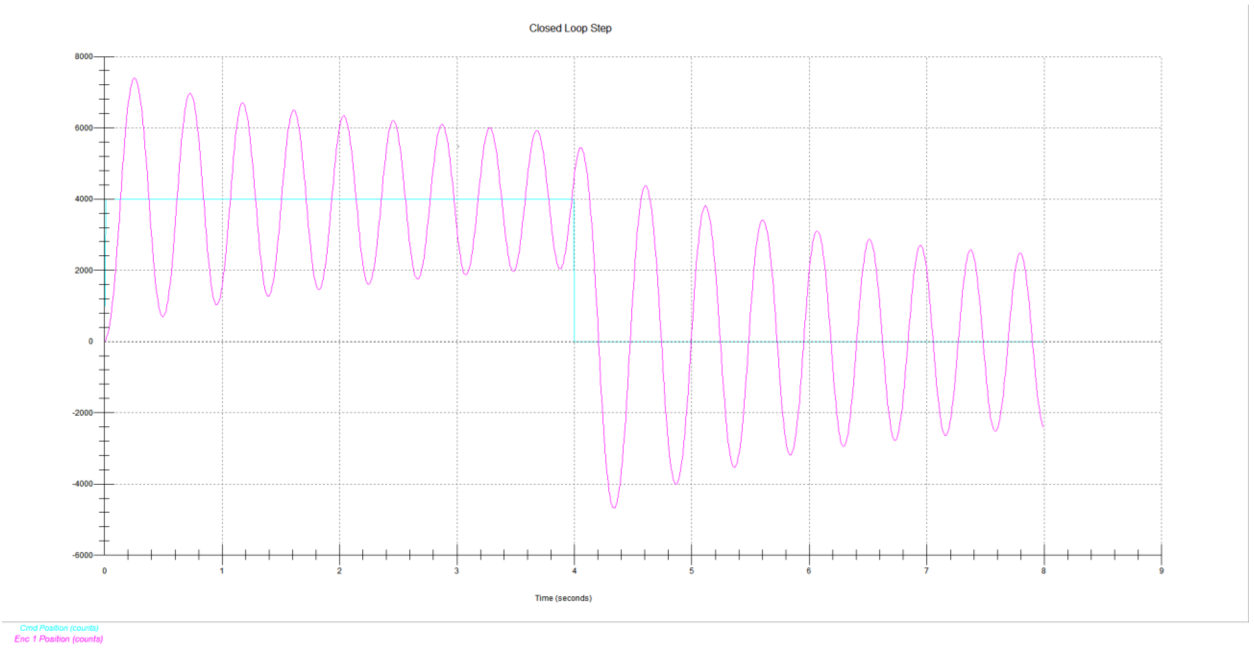
Substituting the values: $(B + Kp) * Ki > (J + Kd) * 0$

Since $d = 0$, the condition is always satisfied as long as $B + Kp$ and Ki are positive.

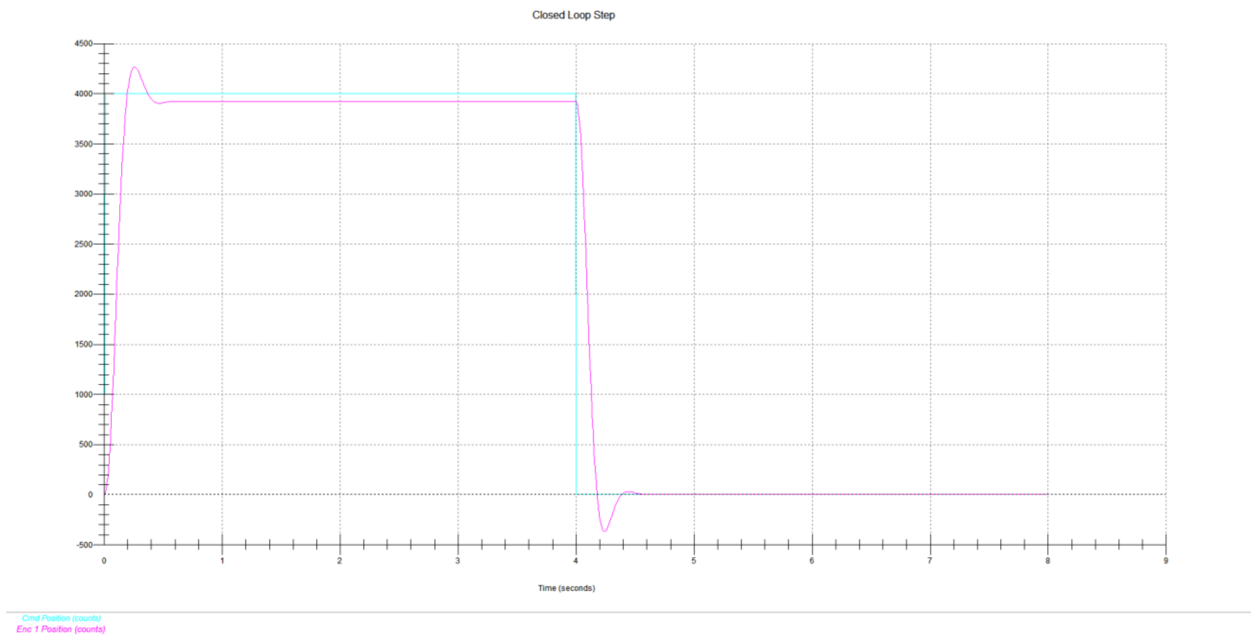
Task 5.3.2: PID Control

1. **Reset the Controller:** The controller was reset from the Utility menu. From the Command menu, viscous friction and the disturbance input were removed. The PID controller was implemented with p=0.2, Ki=0, and Kd=0. A closed-loop step response was obtained using a step size of 4000 Counts, a dwell time of 4000 ms, and 1 repetition. The response was displayed within the dwell period and the display was reduced for later comparison.
2. **Tuning Derivative Control:** The value of Kd was successively increased in small steps (e.g., 0.002) to achieve a desirable overshoot. The best value of Kd was determined by increasing it up to 0.015, aiming for overshoot values in the 5-10% range. The chosen value of Kd, the overshoot (OS), and the steady-state error were recorded.
3. **Tuning Integral Control:** The value of Ki was successively increased in steps of 0.01 until the offset in the step response was almost reduced to zero. The chosen value of Ki was recorded.

Results and Observations



Plot 10: Plot of the closed-loop PID Controller of step 5.3.2.1

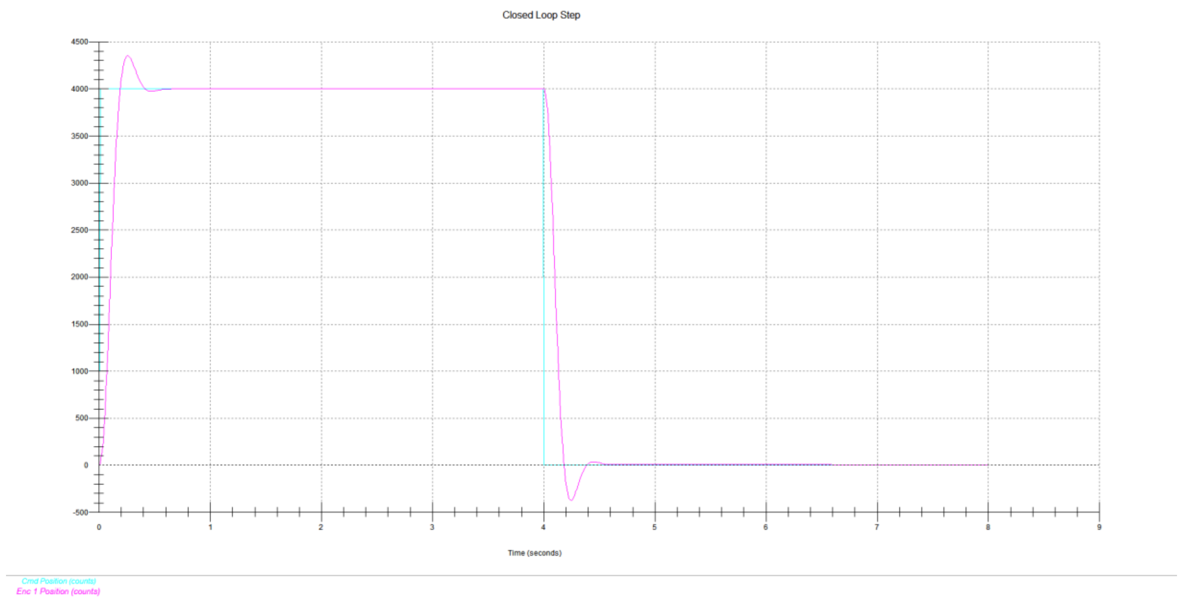


Plot 11: Closed-loop PID Controller of step 5.3.2.2 with a Kd=0.014

From the plot: $\%OS = \frac{4300-4000}{4000} \times 100 = 7.5\%$

Steady-State Error = Reference Value − Steady-State Value

Steady-State Error = 4000 – 3900 = 100 counts

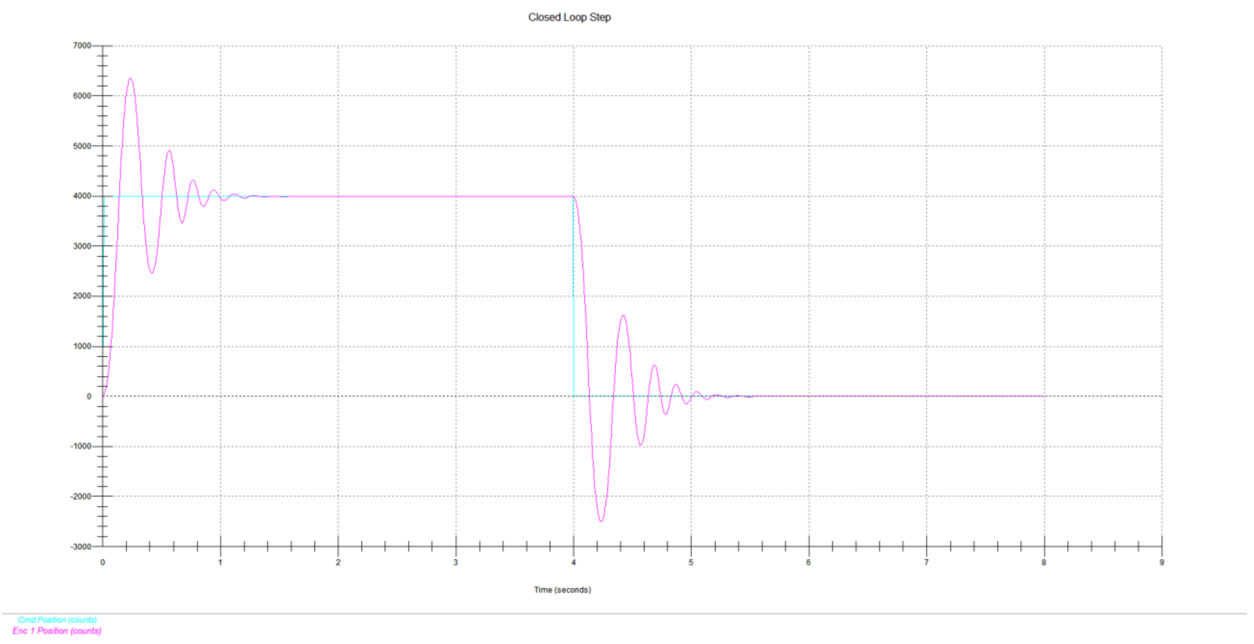


Plot 12: Closed-loop PID Controller of step 5.3.2.3 with Kd=0.014 and Ki=0.04 (step response 0)

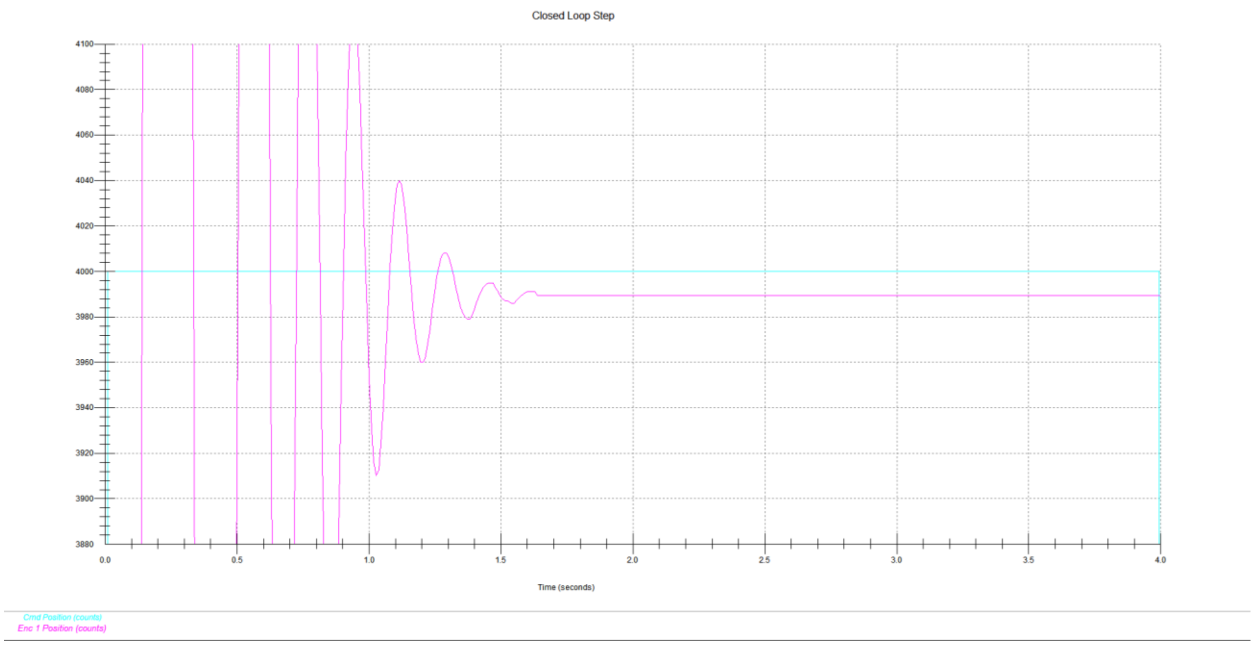
Task 5.4.1: Use Step Signal with PD

1. **Reset the Controller:** The controller was reset from the Utility menu. The "PI + velocity feedback" control algorithm was implemented with the following settings: Ts = 0.00442s, Kp=1, Ki=0, and Kd=0.01, using the Continuous Time type.
2. **Closed-Loop Step Test:** A closed-loop step test was performed using a Step Size of 4000 Counts, a Dwell Time of 4000 ms, and 1 repetition.
3. **Plot Waveform:** The waveform was viewed by clicking on Plot. Using axis scaling, the steady-state step error (tracking error) in counts was obtained from the increasing part of the response.
4. **Repeat Steps with Different Kp Values**: The above steps were repeated with Kp set to 0.5 and 0.2, keeping Ki and Kd unchanged. The system became stable, resulting in an underdamped step response. The percent overshoot (PO) and the damped natural frequency ωd were obtained using suitable axis scaling. Kd was then increased to 0.1, and the effect on the step response was observed.

Results and Observations



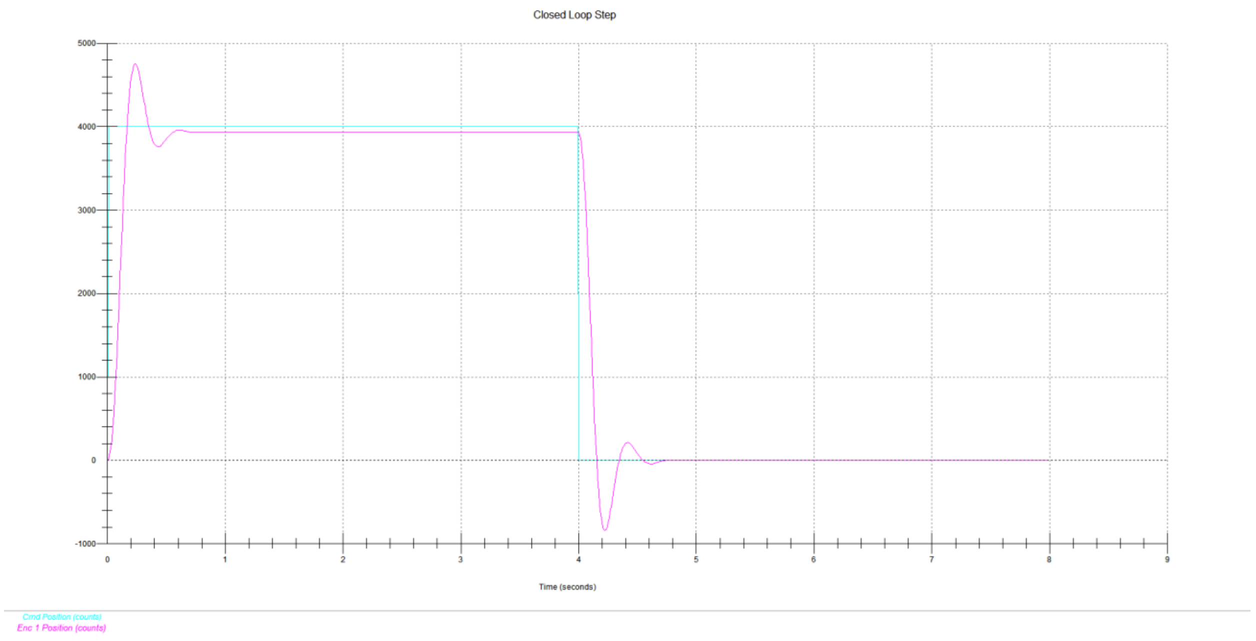
Plot 13: Step Signal with PD Controller of step 5.4.1.1



Plot 14: closed-loop step test using a Step Size of 4000 Counts Zoomed in

Steady-State Error = Reference Value − Steady-State Value

Steady-State Error = 4000 – 3988 = 12 counts

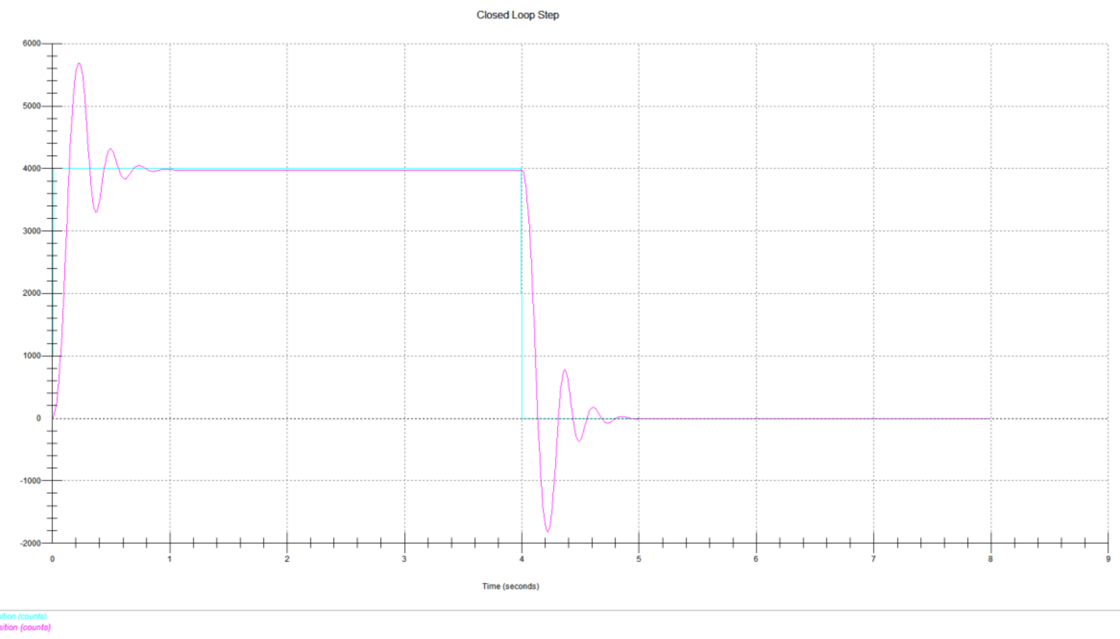


Plot 15: Step Signal with PD Controller with Kp = 0.2

From the plot: $\%OS = \frac{4800-4000}{4000} \times 100 = 20\%$

Period (T) = 0.30 – 0.12 = 0.18 sec

$$\omega d = \frac{2\pi}{T} = \frac{2\pi}{0.18} \approx 34.9\, rad/s$$

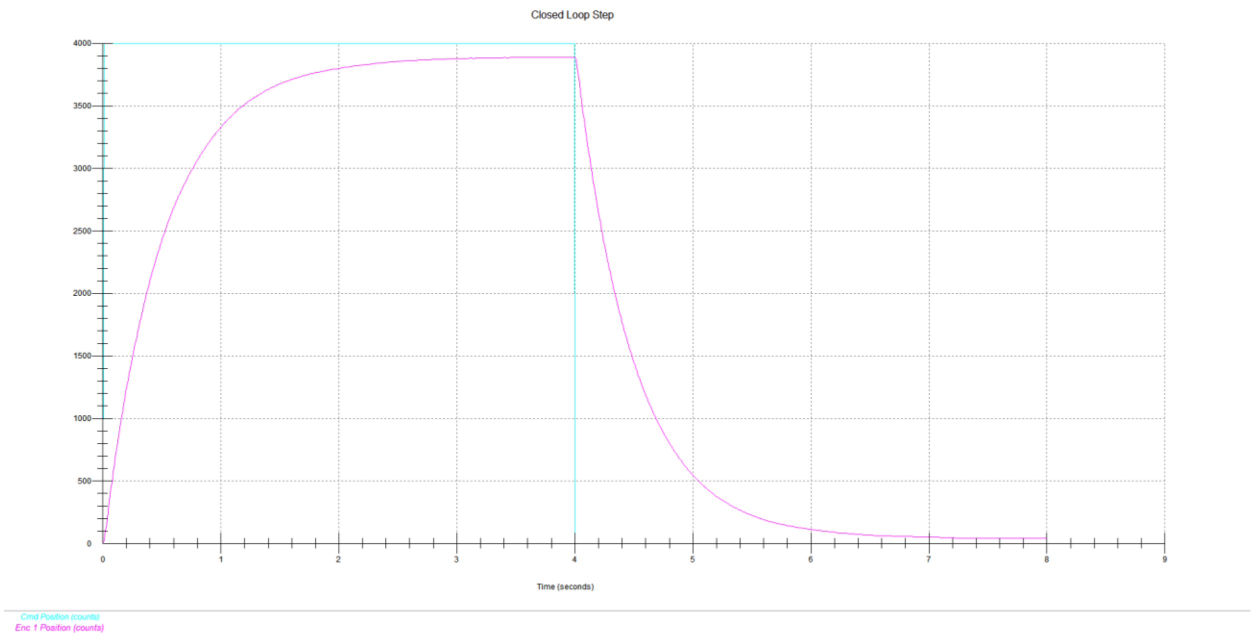


Plot 16: Step Signal with PD Controller with Kp = 0.5

From the plot: $\%OS = \frac{5600-4000}{4000} \times 100 = 40\%$

Period (T) = 0.40 – 0.10 = 0.30 sec

$$\omega d = \frac{2\pi}{T} = \frac{2\pi}{0.30} \approx 20.94\, rad/s$$



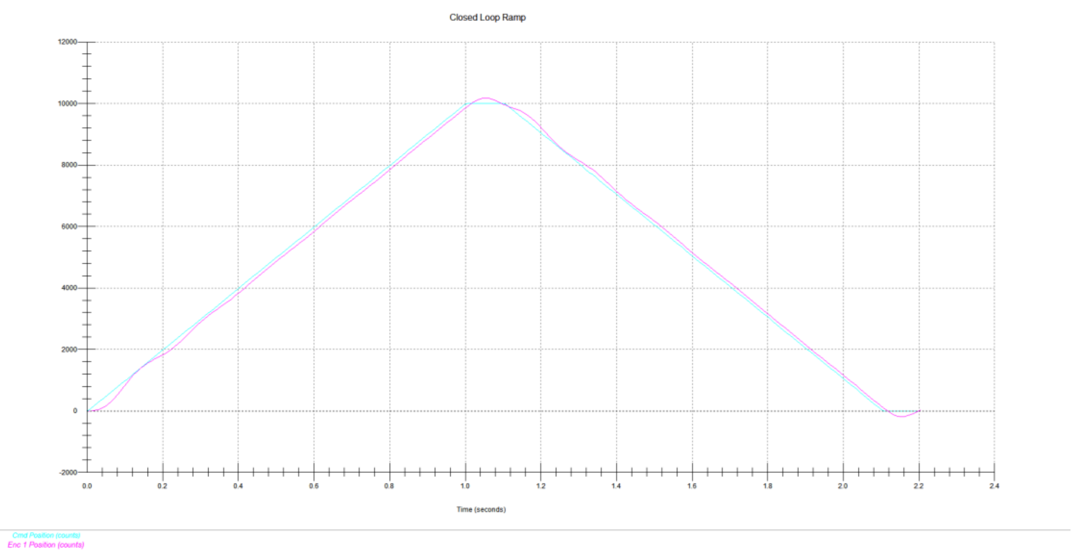
Plot 17: Step Signal with PD Controller with Kp = 0.2 with Kd increase to 0.1

The system response is slow with a long settling time (close to 7 seconds) and minimal oscillations, indicating a highly damped system with no overshoot compared to the other plots which has overshoot of 20% and 40% respectively and a shorter settling time of approximatively 5 seconds. Increasing Kd reduces the overshoot and increases the damping, leading to a more stable but slower response.

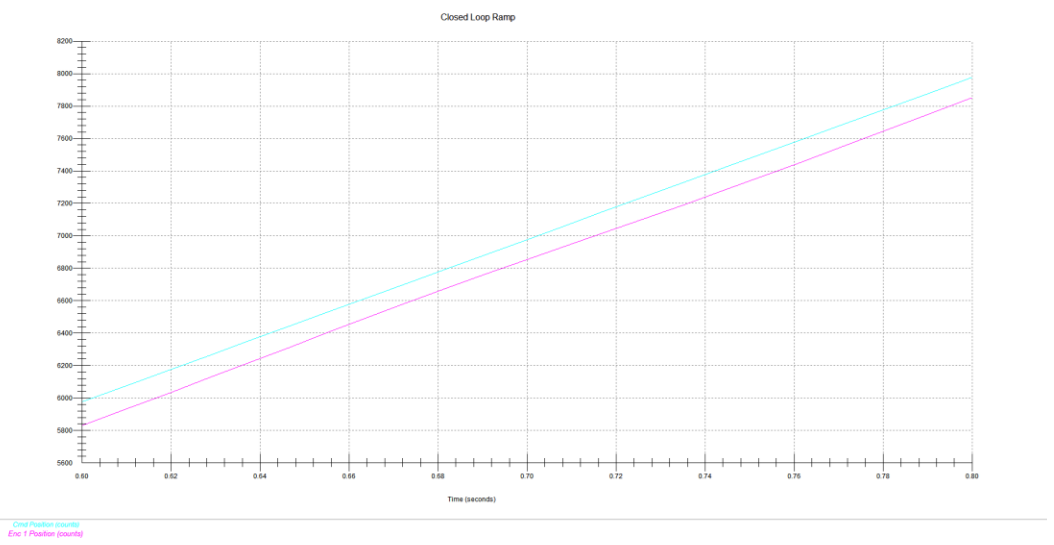
Task 5.4.2: Use Ramp Signal

1. **Reset the Controller**: The controller was reset from the Utility menu. The “PI + velocity feedback” control algorithm was implemented with the following settings: Ts = 0.00442s, Kp=1, Ki=0, and Kd=0, using the Continuous Time type.
2. **Perform Ramp Test**: A RAMP test was set up with a Ramp Size of 10000 counts, Velocity of 10000 Counts/sec, Dwell Time of 100 ms, and 1 repetition. The test was performed.
3. **Repeat with PID Controller**: Steps 1-2 were repeated using the same gains (Kp=1, Ki=0, and Kd=0.01) but with the “PID” controller implemented instead of the “PI + velocity feedback” configuration. Note that in this case, there was no minor loop.

Results and Observations



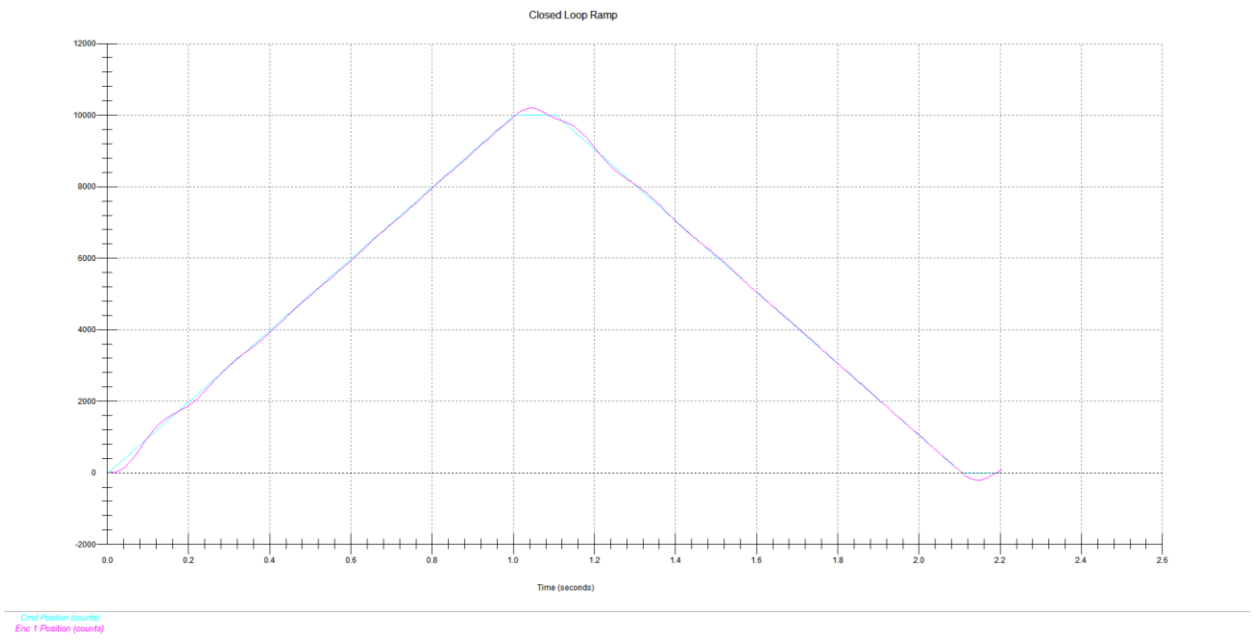
Plot 18: Ramp Signal with PI + velocity feedback of step 5.4.2.1



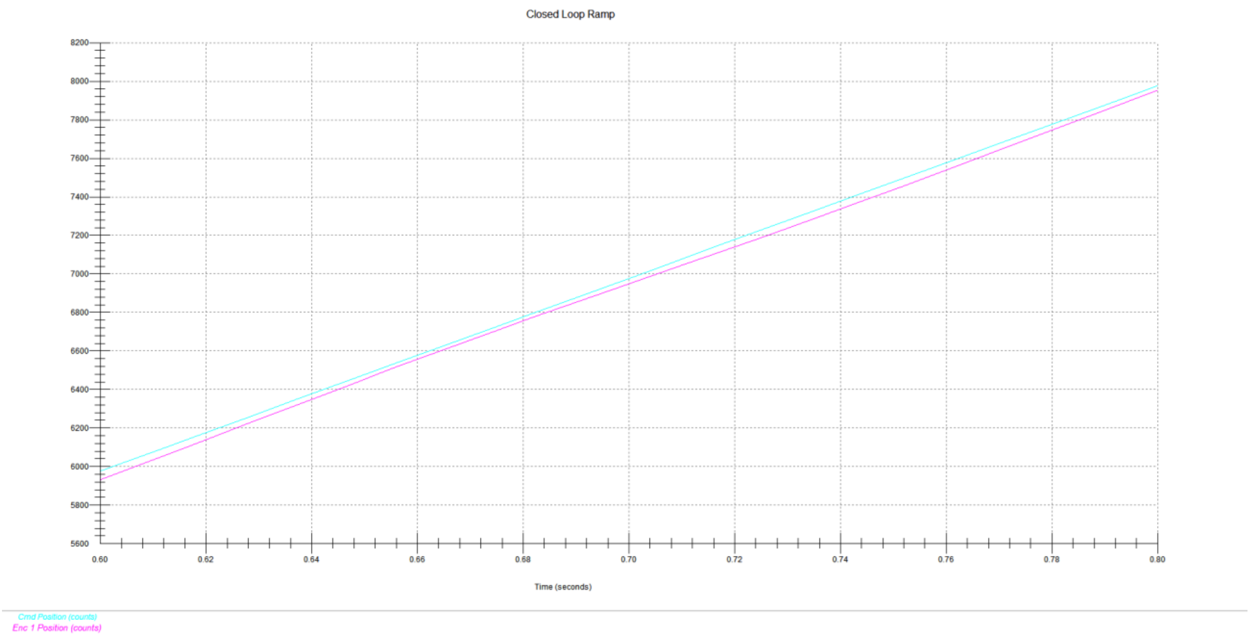
Plot 19: Zoomed in and increasing part of plot 18 to calculate the steady state Step error

Steady-state Error (e_{ss}) = Commanded Position – Encoder 1 Position

$e_{ss} = 5980 - 5840 = 140$ counts



Plot 20: Ramp Signal with PI + velocity feedback of step 5.4.2.3 (Kp= 1, Ki = 0, and Kd= 0.01)



Plot 21: Zoomed in and increasing part of plot 20 to calculate the steady state Step error

Steady-state Error (e_{ss}) = Commanded Position – Encoder 1 Position

e_{ss} = 5980 − 5930 = 50 counts

Tabulated Results:

Plot Number	Controller Type	Kp	Ki	Kd	Steady-State Error (counts)	Overshoot (%)	Damped Natural Frequency (rad/s)
Plot 14	PD	0.2	0	0.01	12	20%	34.9
Plot 15	PD	0.2	0	0.01	N/A	N/A	N/A
Plot 16	PD	0.5	0	0.01	N/A	40%	20.94
Plot 17	PD	0.2	0	0.10	N/A	N/A	N/A
Plot 18	PI + V	1	0	0.01	140	N/A	N/A
Plot 20	PI + V	1	0	0.01	50	N/A	N/A

The step response analysis reveals that increasing the proportional gain (Kp) in the PD controller significantly impacts the overshoot and damping characteristics of the system. Specifically, raising Kp from 0.2 to 0.5 results in a notable increase in overshoot from 20% to 40%, as seen in Plots 14 and 16. Additionally, adjusting the derivative gain (Kd) to 0.1 while keeping Kp at 0.2 (Plot 17) leads to a much smoother response with reduced overshoot. This indicates that higher Kd values enhance system stability and damping. In contrast, the ramp response analysis shows the effects of the PI + velocity feedback controller's tuning on steady-state error. With Kp set at 1 and Ki at 0, increasing Kd from 0 to 0.01 substantially decreases the steady-state error from 140 counts to 50 counts, as observed in Plots 18 and 20. This reduction in error demonstrates the improved accuracy in tracking the commanded position, highlighting the importance of derivative gain in achieving better performance in control systems.

Derivation and Calculation:

Transfer Functions and System Type:

1. **PID Controller with Ki = 0**:
   - The open-loop transfer function (OLTF) for a PID controller with Ki = 0 is given by: G(s)H(s) = Kp(s + Kd) / s(Bs + K)
   - The closed-loop transfer function (CLTF) can be derived as: CLTF = (G(s)H(s)) / (1 + G(s)H(s)) = Kp(s + Kd) / (s(Bs + K) + Kp(s + Kd))
   - Simplifying, we get: CLTF = Kp(s + Kd) / (s^2B + s(K + Kp) + KpKd)
   - For Ki = 0, the system type is 1, as there is one integrator (pole at origin).
2. **PI + Velocity Feedback (PI + V)**:
   - The open-loop transfer function (OLTF) for a PI + V controller is given by: G(s)H(s) = Kp / (Bs + K)
   - The closed-loop transfer function (CLTF) can be derived as: CLTF = (G(s)H(s)) / (1 + G(s)H(s)) = Kp / (Bs + K + Kp)

- Simplifying, we get: CLTF = Kp / (Bs + K + Kp)
- For Ki = 0, the system type is 1, as there is one integrator (pole at origin).

Velocity-Error Coefficients:

1. **PID Controller**:
   - The steady-state ramp error is given by: ess(ramp) = Ramp magnitude / Kvel
   - For a ramp input of 10,000 Counts/sec and ess = 50 counts: Kvel = 10000 / 50 = 200 Counts/sec
   - The velocity error-coefficient for the PID controller is: Kvel(PID) = KKp / B
2. **PI + Velocity Feedback (PI + V)**:
   - For a ramp input of 10,000 Counts/sec and ess = 140 counts: Kvel = 10000 / 140 ≈ 71.43 Counts/sec
   - The velocity error-coefficient for the PI + V controller is: Kvel(PI+V) = KKp / (B + KKd)

By deriving these transfer functions and calculating the velocity error coefficients, it is evident that the PID controller with Ki = 0 provides better performance in terms of minimizing the steady-state ramp error compared to the PI + V controller.

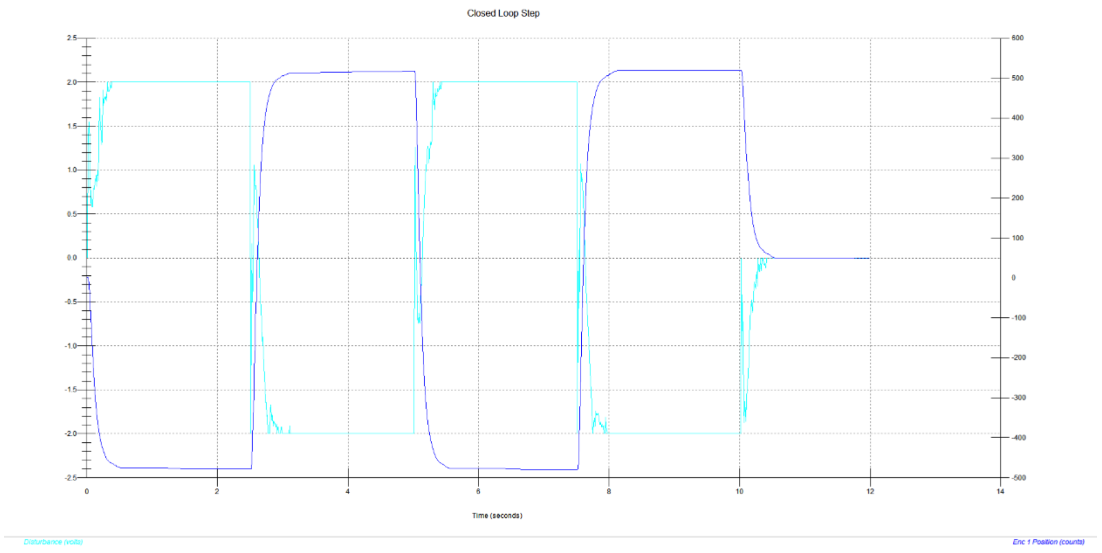
Task 5.5: Disturbance Attenuation

1. **Set up PID Control Scheme**: The PID control scheme was set up with Continuous Time type, Kp=0.2, Ki=0, and Kd=0, with viscous friction introduced through the disturbance motor set at 4 Volt-sec/radian.
2. **Input Step Trajectory**: From the Command menu, an input STEP trajectory was selected with a magnitude of zero counts and a 6000 ms dwell, with 1 repetition. This configured the system to acquire data over a period of 12 seconds.
3. **Disturbance Step Trajectory**: From the Command menu, a disturbance step trajectory was selected with 2 Volts magnitude, a 2500 ms dwell, and 2 repetitions. Encoder 1 Position was selected for the Right axis and Disturbance Effort for the Left axis in the Set Up Plot, ensuring both “Include Viscous friction” and “Include Disturbance” boxes were checked. The system was then run.
4. **Integral Action Implementation**: The algorithm was re-entered and Ki was set to 0.6. The controller was implemented, and Step 15 was repeated with the integral action included. The resulting plot showed the integral action attempting to force the error to

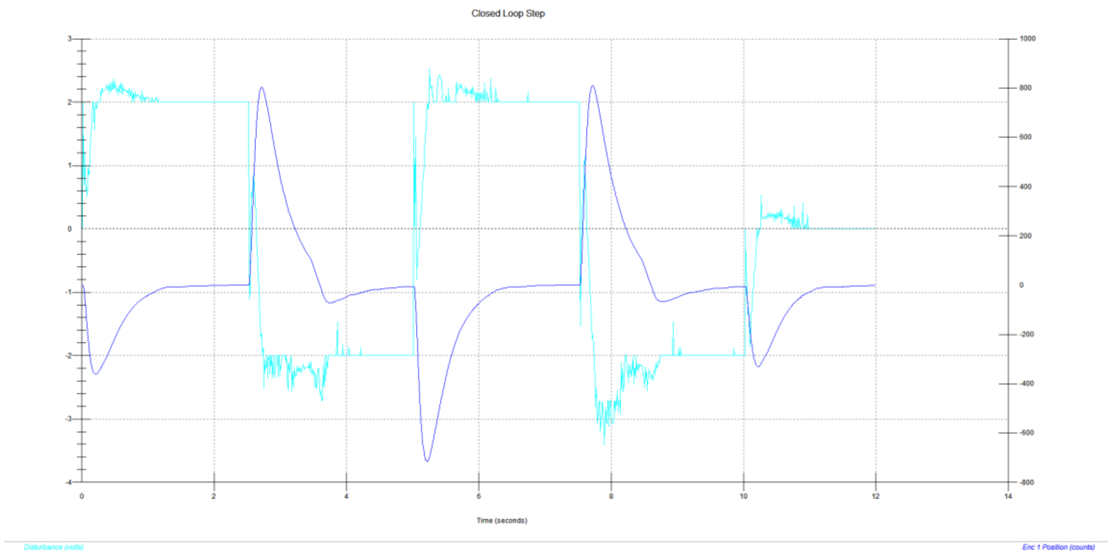
zero. The error-reducing force was also felt by hand by setting the disturbance to zero, implementing the system, and gently disturbing the load disk by a small angle.

5. **Incremental Integral Action**: Step 4 was repeated with lower and higher values of Ki, and observations were described. The results were used to evaluate the maximum value of Ki that could be used with the given settings.

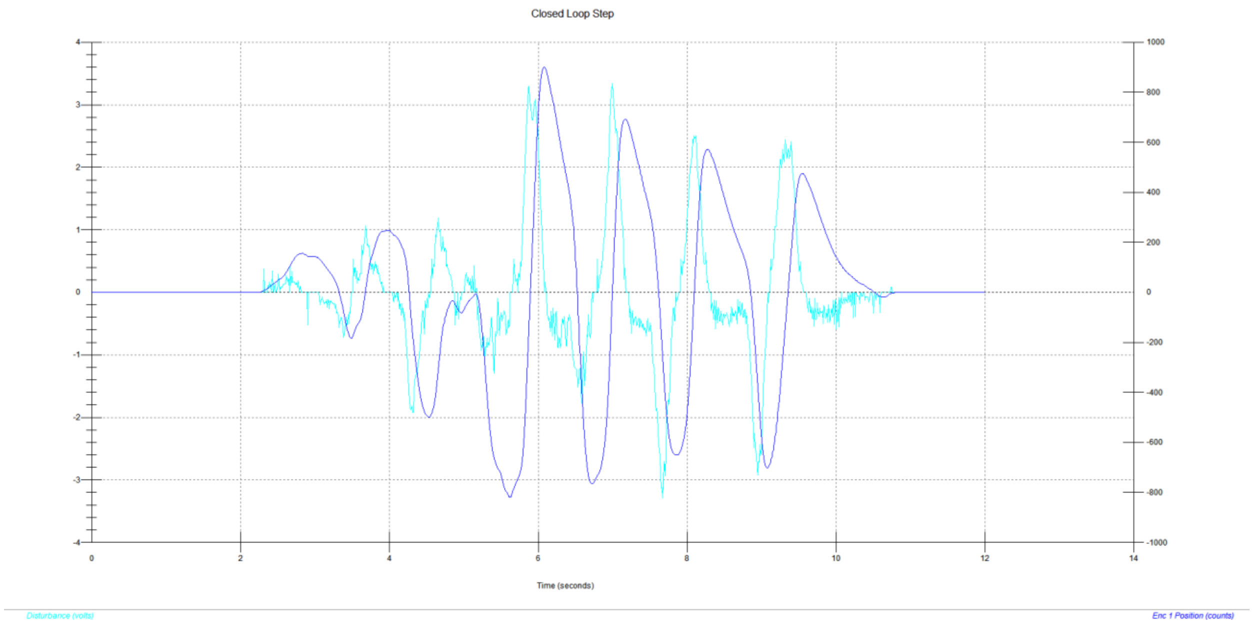
Results and Observations



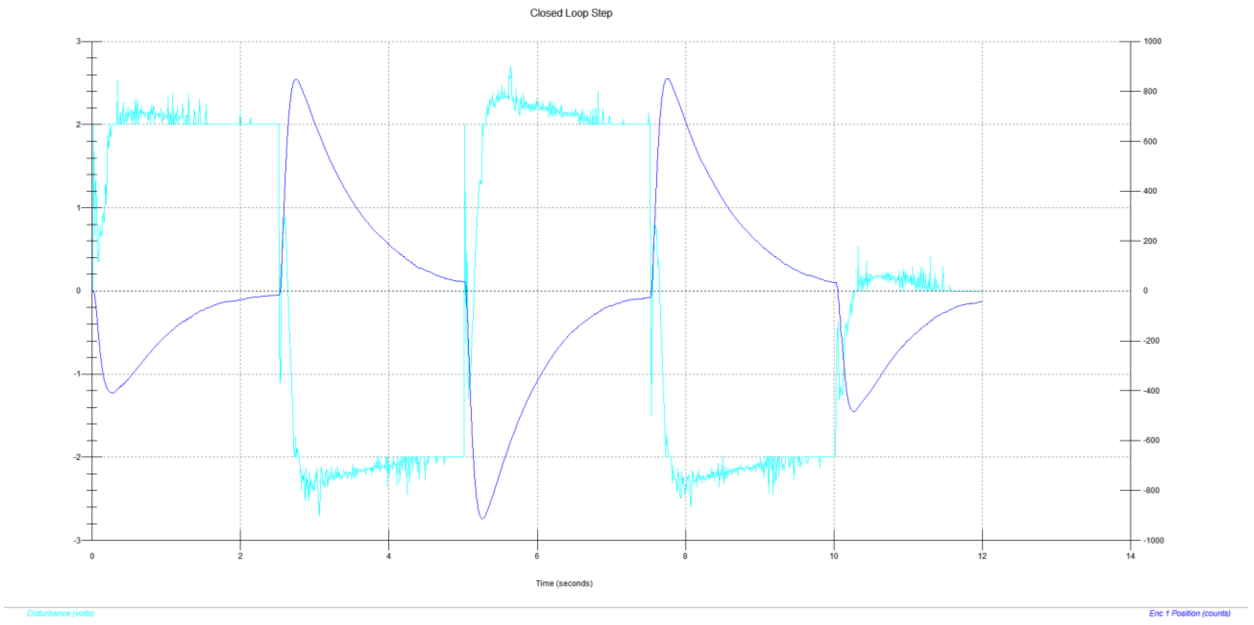
Plot 22: PID Controller with disturbance step (5.5.3)



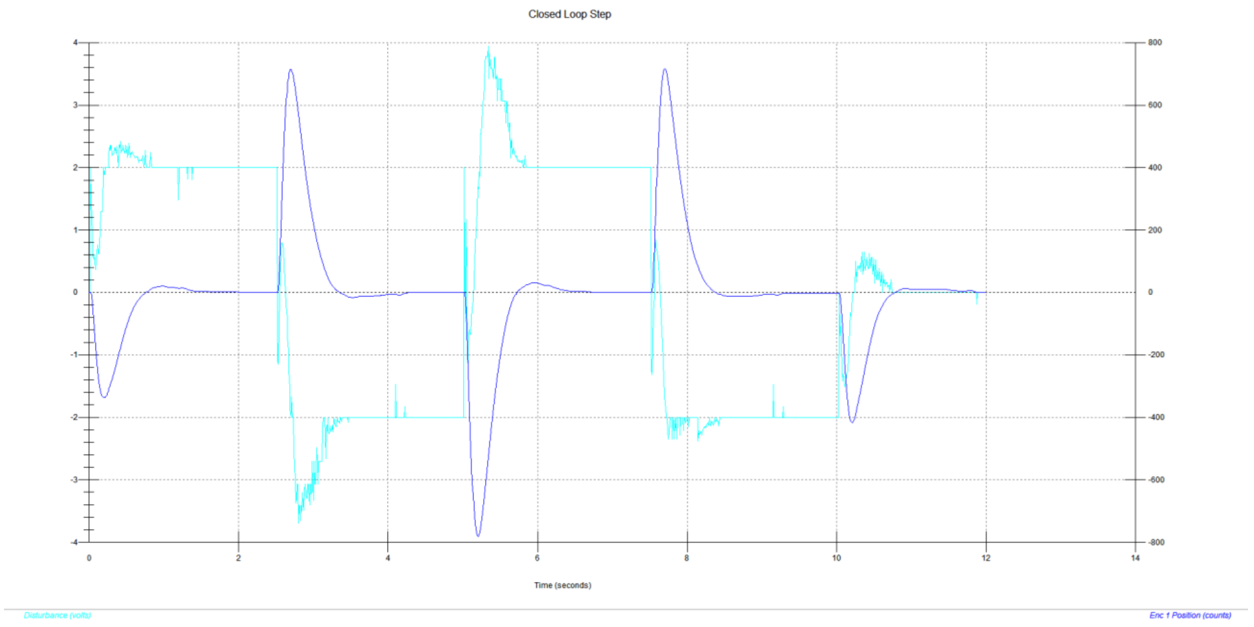
Plot 23: PID Controller with disturbance step with Ki = 0.6 (5.5.4)



Plot 24: PID Controller with manual disturbance



Plot 25: PID Controller with disturbance step with Ki = 0.3 (5.5.5)



Plot 26: PID Controller with disturbance step with Ki = 1.0 (5.5.5)

Based on the disturbance step responses, we can observe the effect of different integral gain (Ki) values on the system's behavior. When Ki is set to 0, the system oscillates significantly and does not settle down quickly, indicating high sensitivity to disturbances and an inability to correct steady-state errors efficiently. With Ki increased to 0.6, the system's ability to handle disturbances improves, showing fewer oscillations and a better tendency to settle down compared to no integral action. At Ki = 0.3, the system demonstrates a more controlled response with minimal oscillations, indicating an optimal range for Ki where the system effectively manages disturbances without excessive instability. However, with Ki set to 1.0, the system experiences significant oscillations and instability, struggling to settle down, which shows that an excessively high Ki deteriorates performance. Thus, based on these observations, the maximum value of Ki that maintains a stable and responsive system is around 0.3, as higher values introduce instability and excessive oscillations.

Given that T(s) is the closed-loop transfer function, we can express the equivalent open-loop transfer function Gufs(s) as:

$G_{ufs}(s) = T(s) / [1 - T(s)]$

Assuming the closed-loop transfer function T(s) is in the form:

$T(s) = K / (s^2 + as + K)$

where K is the gain and a is a constant representing system dynamics.

Using the given expression, we obtain Gufs(s):

$G_{ufs}(s) = (K / (s^2 + as + K)) / (1 - (K / (s^2 + as + K))) = K / (s^2 + as + K - K) = K / (s^2 + as)$

Conclusion

In conclusion, this report evaluated the performance of a PID controller under various configurations and disturbance conditions. By analyzing step responses and ramp responses with different Ki values, we observed the system's behavior in terms of overshoot, steady-state error, and damping. The findings indicate that while increasing Ki improves the steady-state error, it can also introduce significant oscillations and instability if not carefully tuned. Conversely, adjusting Kd can enhance damping and reduce overshoot, contributing to a more stable system response. The optimal balance of these parameters is essential for achieving robust control performance. This study underscores the importance of precise tuning in PID controllers to ensure effective and stable operation under varying conditions.