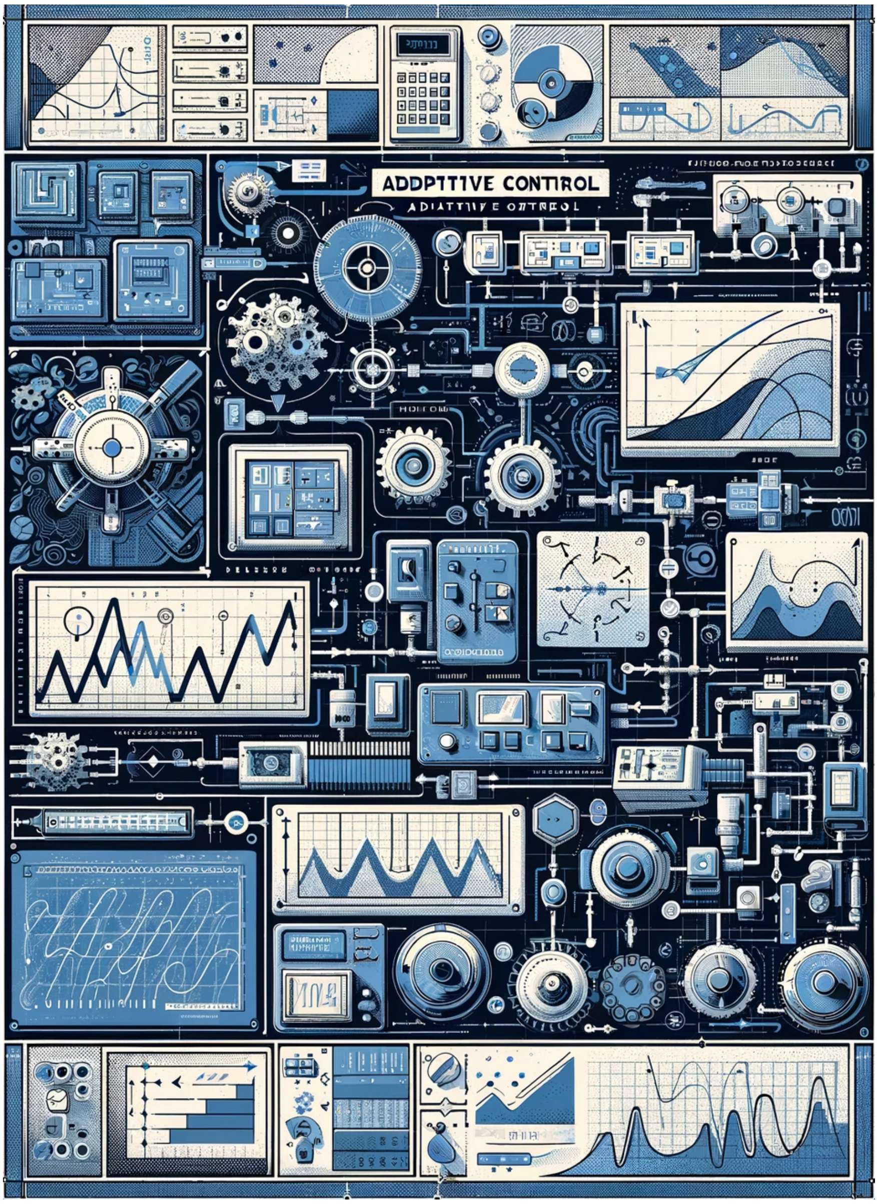
Simulation 4 Adaptive Control

Student Name: Amirhossein Ehsani
Student ID: 810602159
Instructor: Dr. Ayati



June 29, 2024

Contents

List of Figures

List of Tables

1 Dynamic System

The system is give n as follows:

$$\begin{cases} A(q) = (q - 0.1)(q - 0.65) \\ B(q) = (q - 0.46) \\ d = 2 \text{ Delay} \end{cases}$$

The system is both stable and minimum phase.

2 Pole Placement Controller Design

We will take the specifications bellow for the desired system in order to design the controller:

```matlab
%% Desired system specifications
overshootPercentage = 5;
riseTime = 1;

% Calculate damping ratio and natural frequency
zeta = cos(atan2(pi,-log(overshootPercentage/100)));
wn = 1.8/(riseTime);

% Desired pole and zeros
desiredZero = -25;
desiredPole1 = -35;
desiredPole2 = -40;
```

We will use the minimum degree pole placement (MDPP) method without pole-zero cancellation.

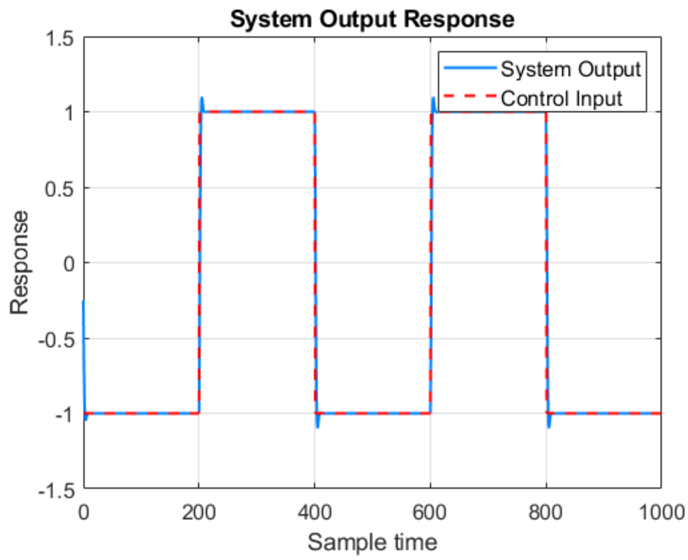


Figure 1: System Response Vs. system input (MDPP without pole-zero cancellation).

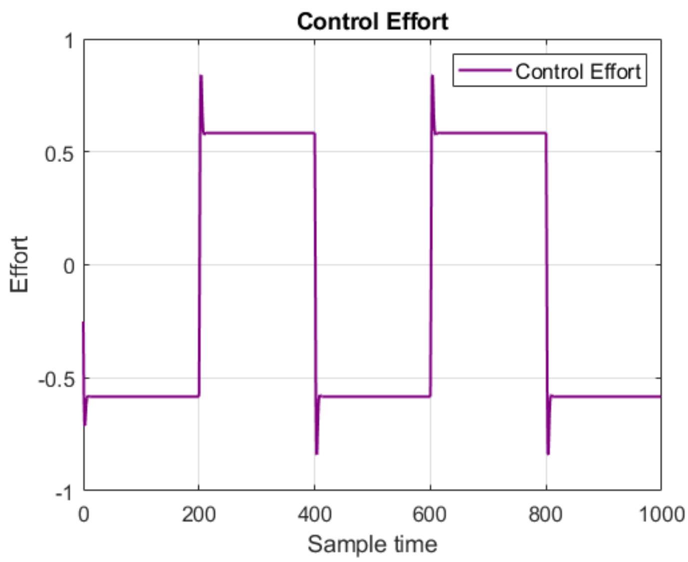


Figure 2: System control effort signal (MDPP without pole-zero cancellation).

As it is shown in the figures above when the control signal is fed into the plant there is delay until it affects on the system's output.

3 Model Predictive Controller (Non-Adaptive)

In this section we will implement the model predictive controllers with no system identification process is not performed.

3.1 One Step Ahead Method

To derive the control relationship and predict the system's output at future time steps, it is necessary to calculate G and F from the following Diophantine equation:

$$1 = A^*(q^{-1})F_d^*(q^{-1}) + q^{-d}G_d^*(q^{-1})$$

To solve this equation, we first write it in terms of the q operator and use the functions developed in Simulink to solve the equation for the given system.

The above equations account for the system's delay. It should be noted that the primary control codes are estimated and analyzed in all cases using these equations.

Thus, after calculating G and F, the polynomials α and β must be computed using them to predict future inputs and design the control input. The relationship is given by:

$$\beta(q^{-1})u(t) = y^*(t+d) - \alpha(q^{-1})y(t)$$

Input noise and disturbances must be defined. It should be noted that this code is very similar in all defined models, where the input and control output are calculated using a function named `filtering`. In this function, the u and y values are filtered and adjusted. In reality, a multiplying polynomial is used to compute u and y values from the following functions.

We will obtain the results with and without white noise with the mean zero and variance of 0.001 and also add disturbance and check robustness of the controller.

3.1.1 Results Without Noise (One Step Ahead)

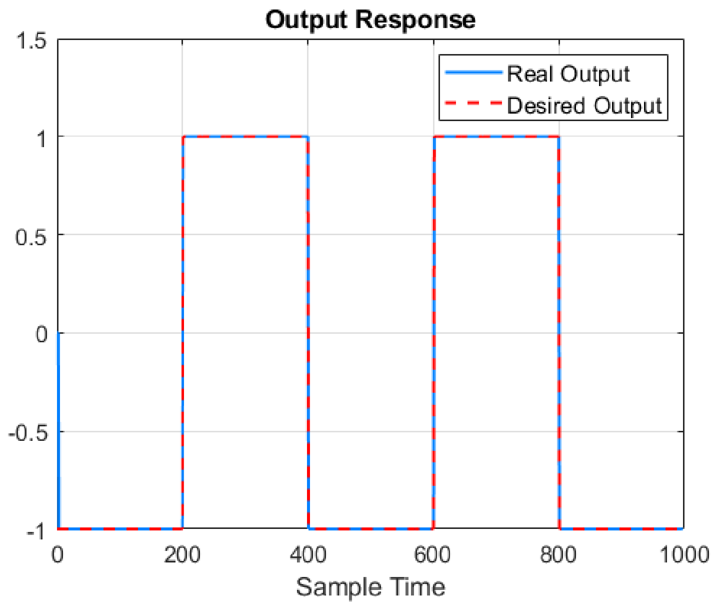


Figure 3: System Response Vs. system input (One step ahead, without noise).

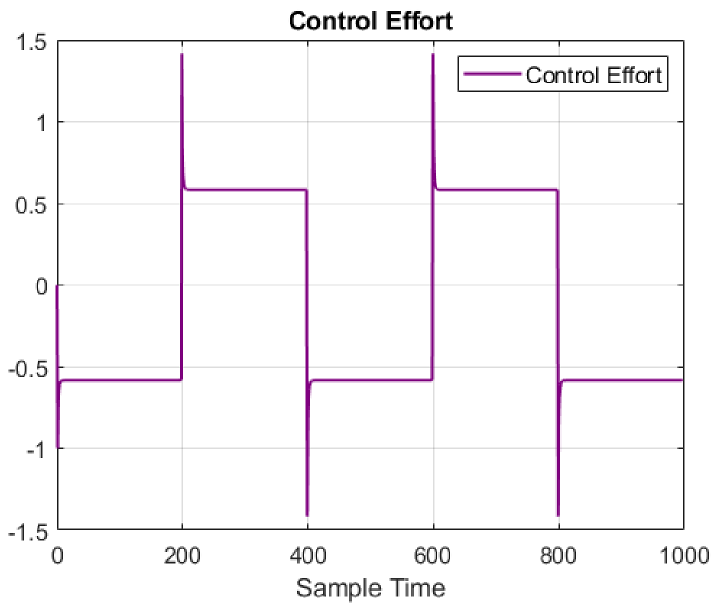


Figure 4: System control effort signal (One step ahead, without noise).

3.1.2 Results With White Noise $\sigma^2 = 0.001$ (One Step Ahead)

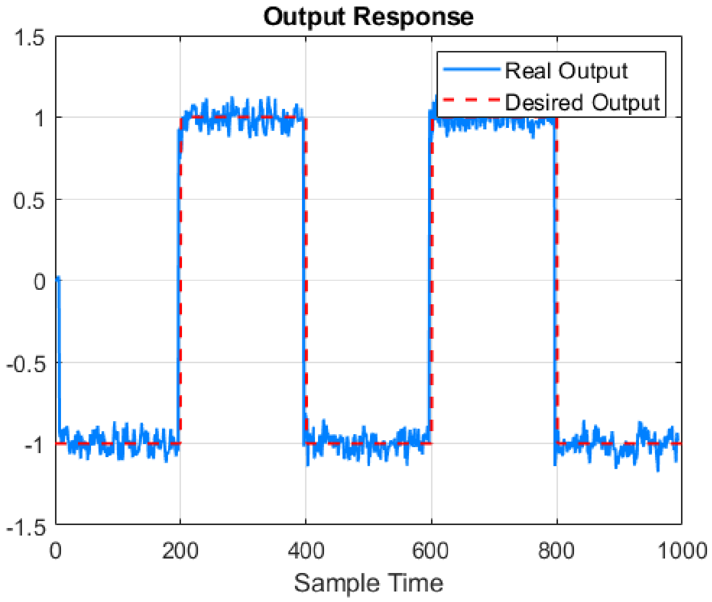


Figure 5: System Response Vs. system input (One step ahead, White Noise $\sigma^2 = 0.001$).

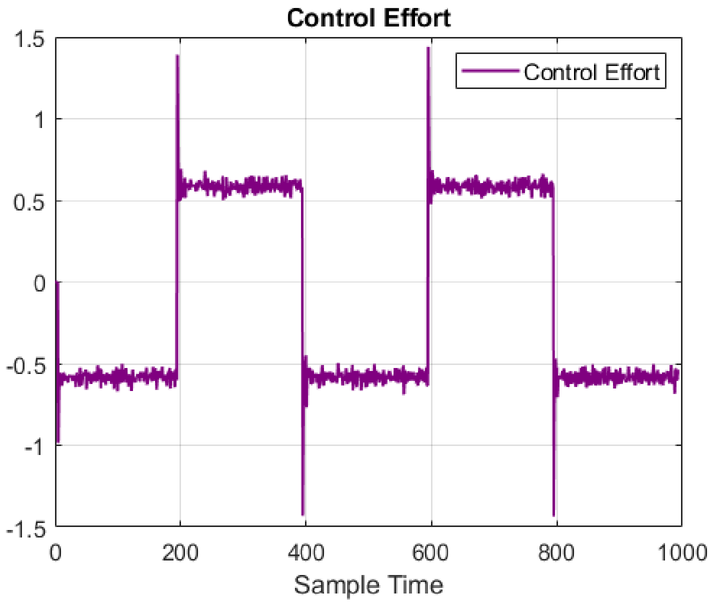


Figure 6: System control effort signal (One step ahead, White Noise $\sigma^2 = 0.001$).

3.1.3 Results With White Noise And Disturbance (One Step Ahead)

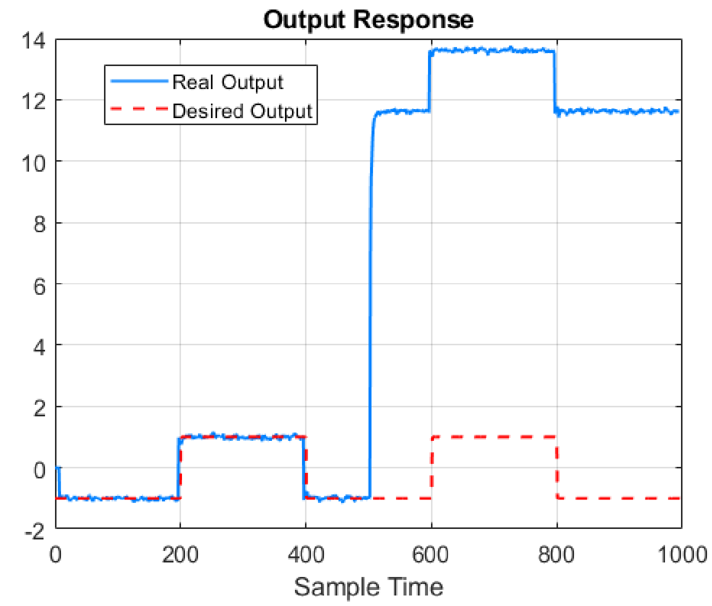


Figure 7: System Response Vs. system input (One step ahead, White Noise And Disturbance).

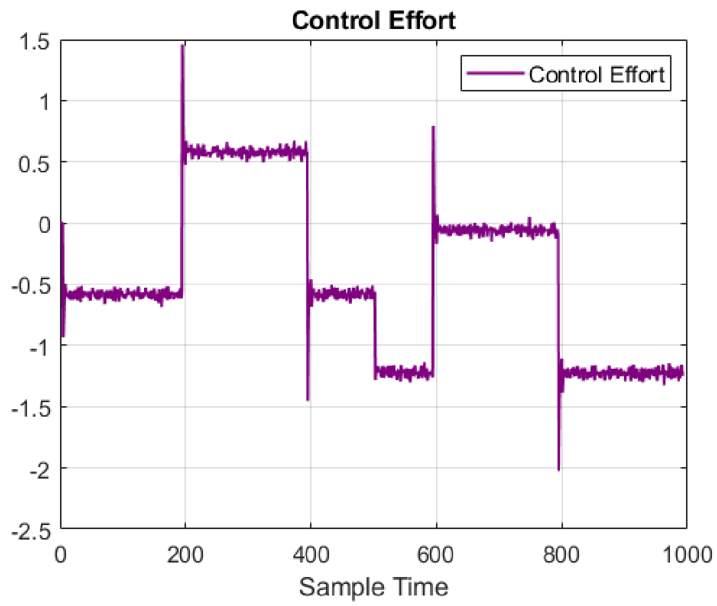


Figure 8: System control effort signal (One step ahead, White Noise And Disturbance).

3.1.4 Results For Changing Delay Of The System Without Noise (One Step Ahead)

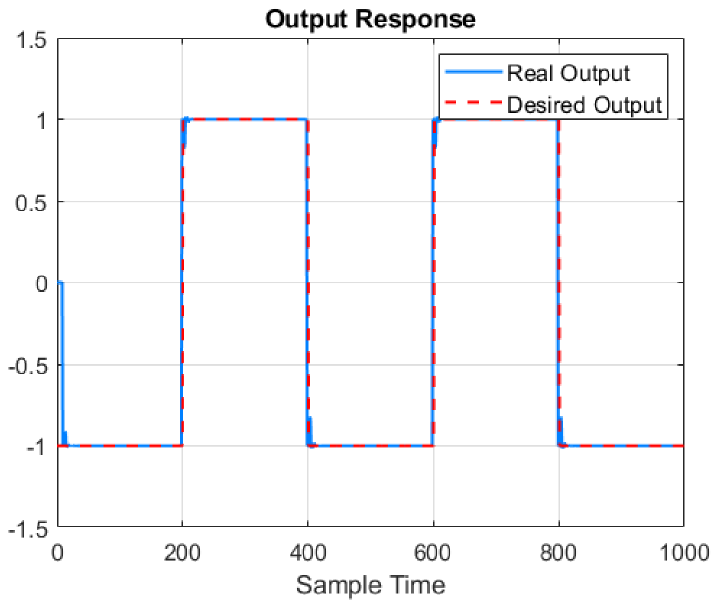


Figure 9: System Response Vs. system input (One step ahead, Changing Delay).

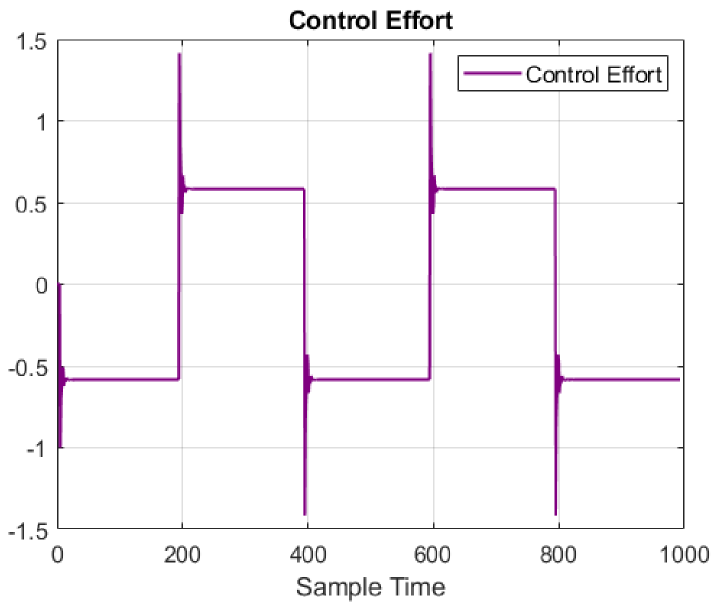


Figure 10: System control effort signal (One step ahead, Changing Delay).

3.1.5 Conclusion On The Results (One Step Ahead)

Examining the figures, we can infer that the controller performed well in all cases except when disturbances were present. In noiseless scenarios, we achieved perfect tracking.

Simultaneously changing the delay in both the controller and the main system does not impact the system's performance; it merely causes the output to follow the input with a greater delay. Therefore, the focus of the question appears to be on examining the effects of a mismatch in the delays considered for the controller and the system. For this analysis, we set the delay in the main system to 2, while the controller's delay was set to 6. Consequently, the dimensions of F and G were adjusted accordingly.

As observed, in this scenario, the system fails to exhibit satisfactory transient behavior. Under such conditions, the influence of a weighted controller becomes highly significant, and this will be explored further.

3.2 Weighted One Step Ahead Method

The main difference in this controller is that a value, λ, is added to the controller, while the rest of the calculations remain the same as before. To calculate the value of λ, the root locus plot has been used, which is shown in the figure below.

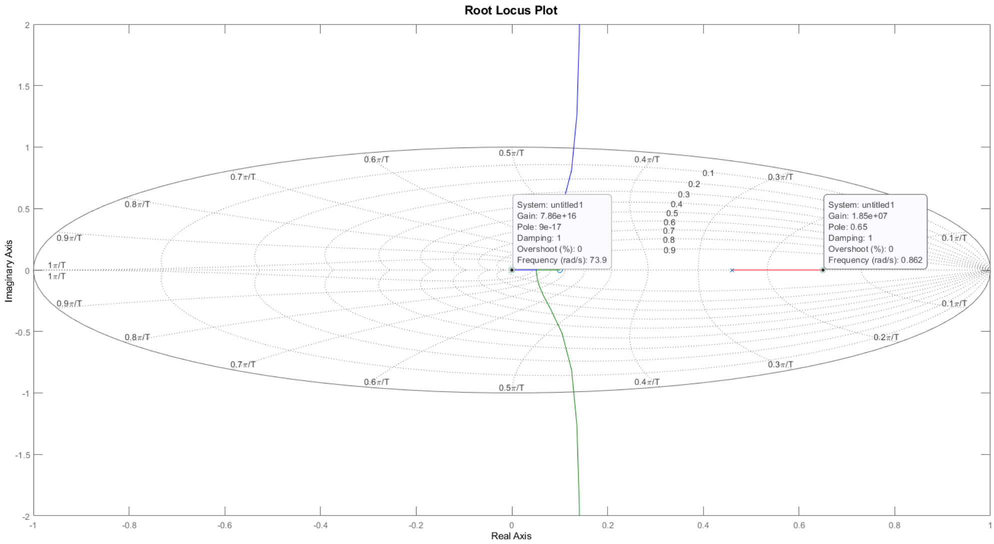


Figure 11: System root locus plot.

As observed, values greater than approximately 0.65 for λ ensure system stability. Therefore, λ has been set to 2. By considering a positive λ, we impose a constraint on u. This approach reduces the control input value, but it also increases the system error. The method for calculating the control input is as follows:

```
control_u(i) = beta_0 * (desired_output(i+d) - filter_signal(alpha, output_y, i) - filter_signal(beta_prime, control_u, i-1)) / (beta_0^2 + landa);
```

3.2.1 Results Without Noise (Weighted One Step Ahead)

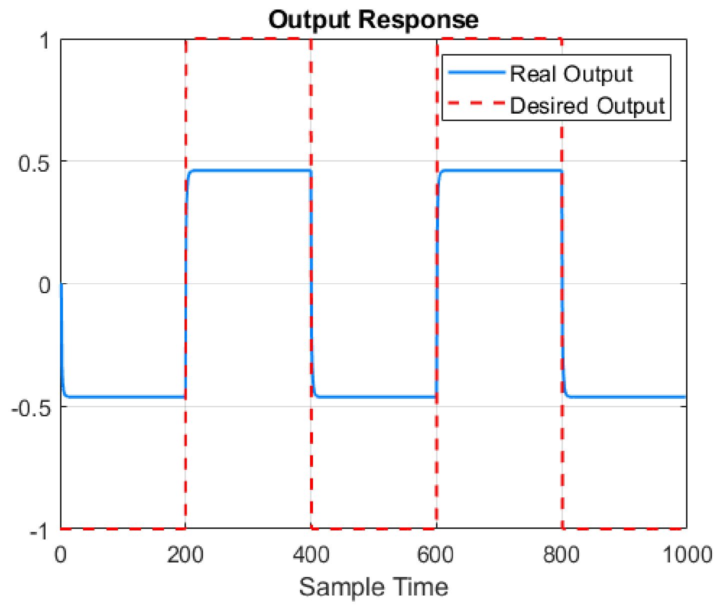


Figure 12: System Response Vs. system input (Weighted One step ahead, without noise).

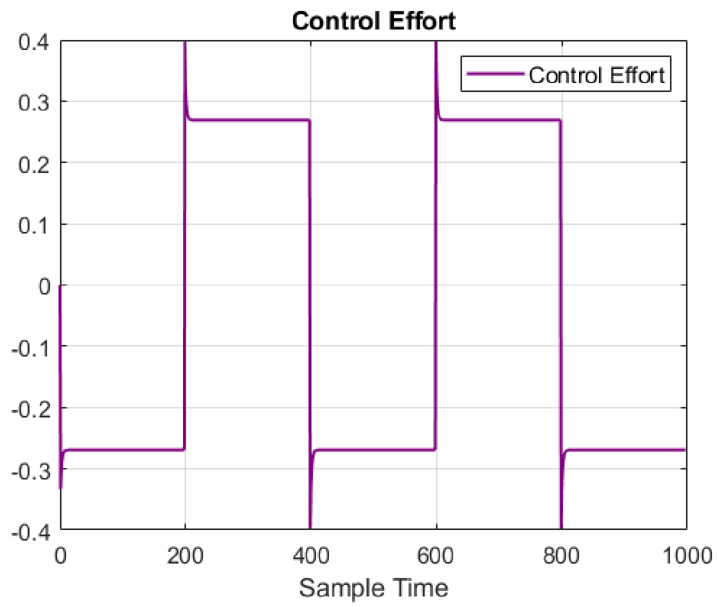


Figure 13: System control effort signal (Weighted One step ahead, without noise).

3.2.2 Results With White Noise $\sigma^2 = 0.001$ (Weighted One Step Ahead)

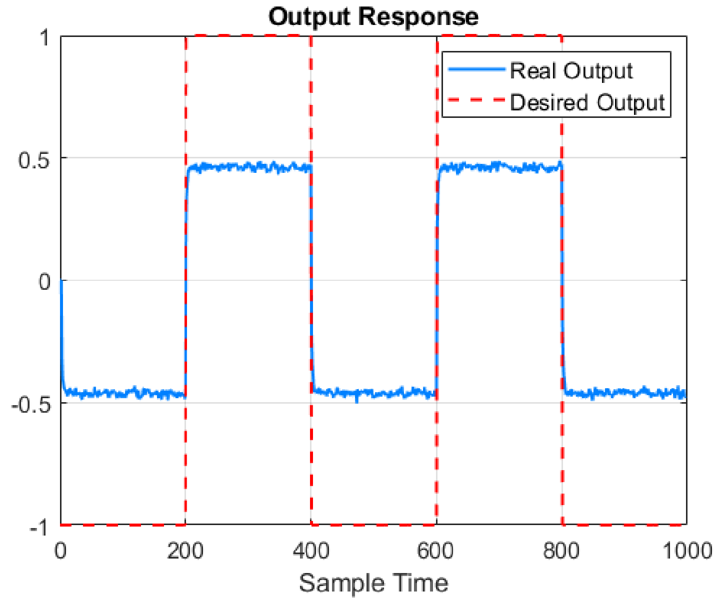


Figure 14: System Response Vs. system input (Weighted One step ahead, White Noise $\sigma^2 = 0.001$).

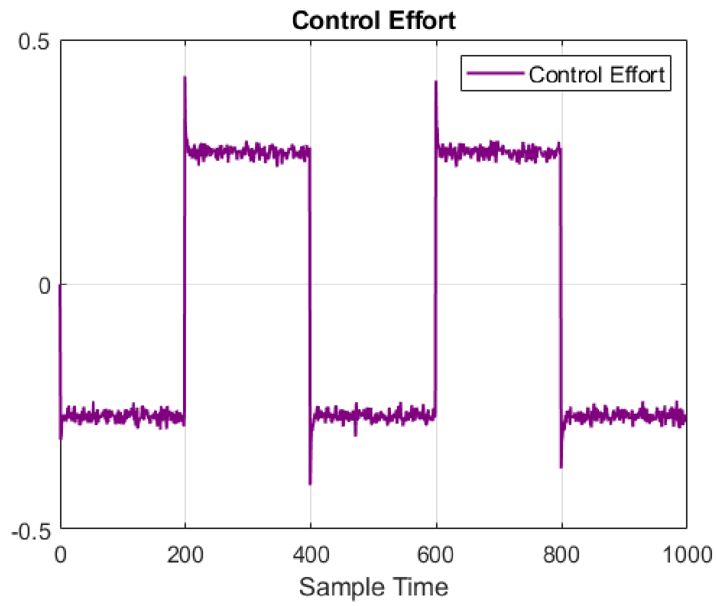


Figure 15: System control effort signal (Weighted One step ahead, White Noise $\sigma^2 = 0.001$).

3.2.3 Results With White Noise And Disturbance (Weighted One Step Ahead)

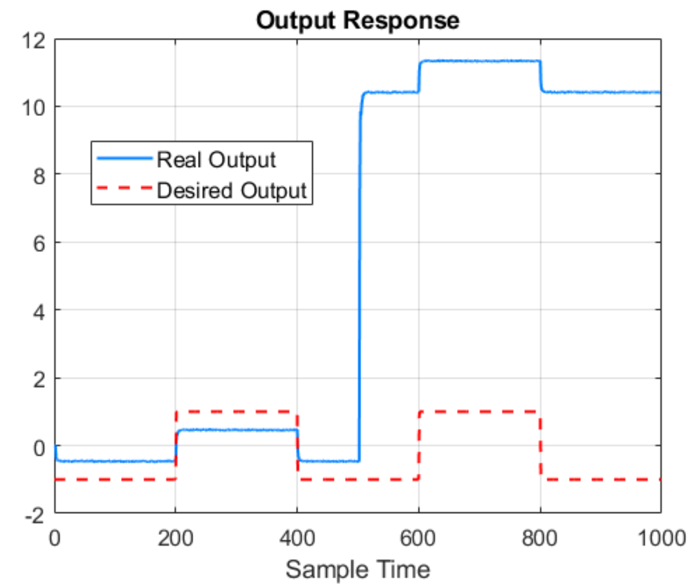


Figure 16: System Response Vs. system input (Weighted One step ahead, White Noise And Disturbance).

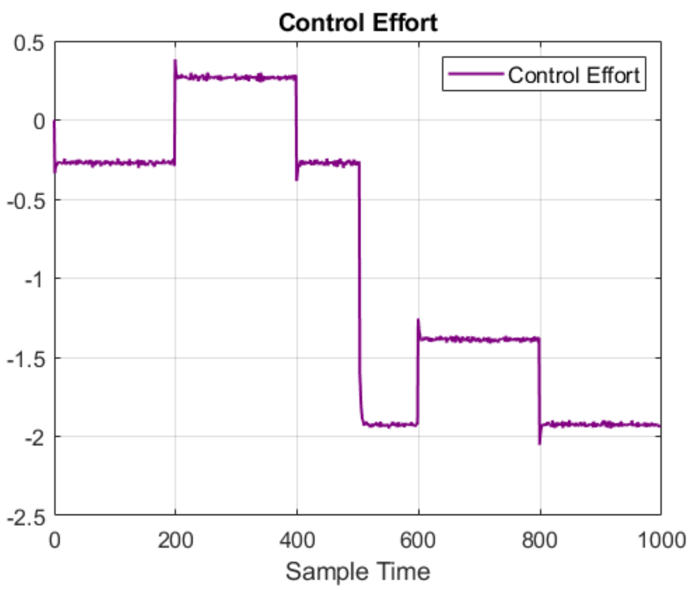


Figure 17: System control effort signal (Weighted One step ahead, White Noise And Disturbance).

3.2.4 Results For Changing Delay Of The System Without Noise (Weighted One Step Ahead)

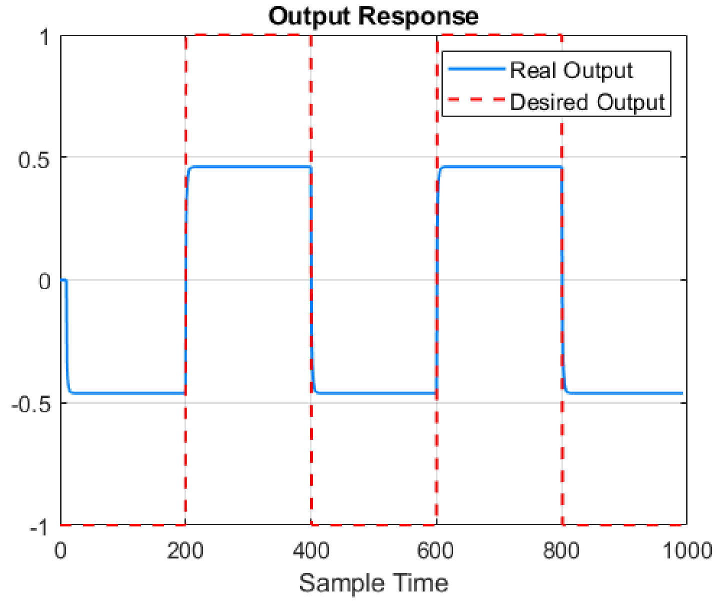


Figure 18: System Response Vs. system input (Weighted One step ahead, Changing Delay).

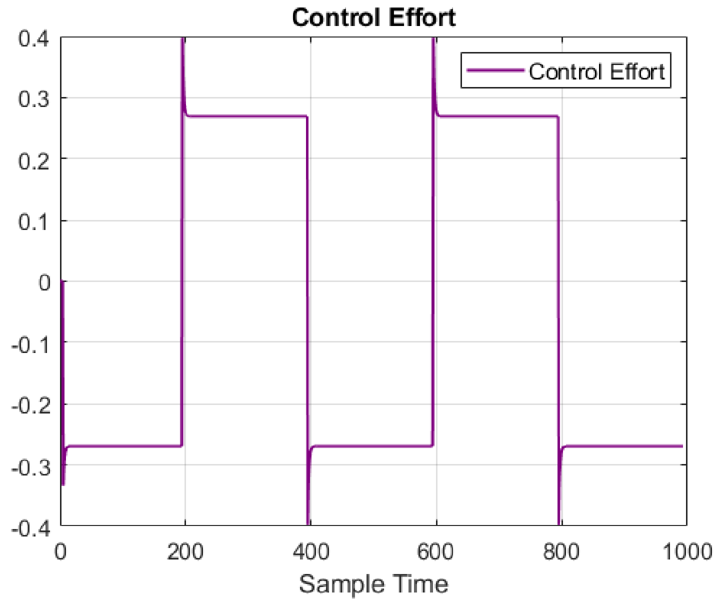


Figure 19: System control effort signal (Weighted One step ahead, Changing Delay).

3.2.5 Conclusion On The Results (Weighted One Step Ahead)

As observed, using the weighted one-step-ahead approach with the added λ significantly improves control effort. However, this comes at the cost of worsened tracking performance, introducing an offset error between the system and the desired response.

In the previous scenario, without the inclusion of λ, the system did not exhibit desirable transient behavior. With the addition of λ, the system's performance has improved. Increasing the value of λ can further enhance the response.

3.3 MPC J2 Loss Function (Non Minimum Phase System)

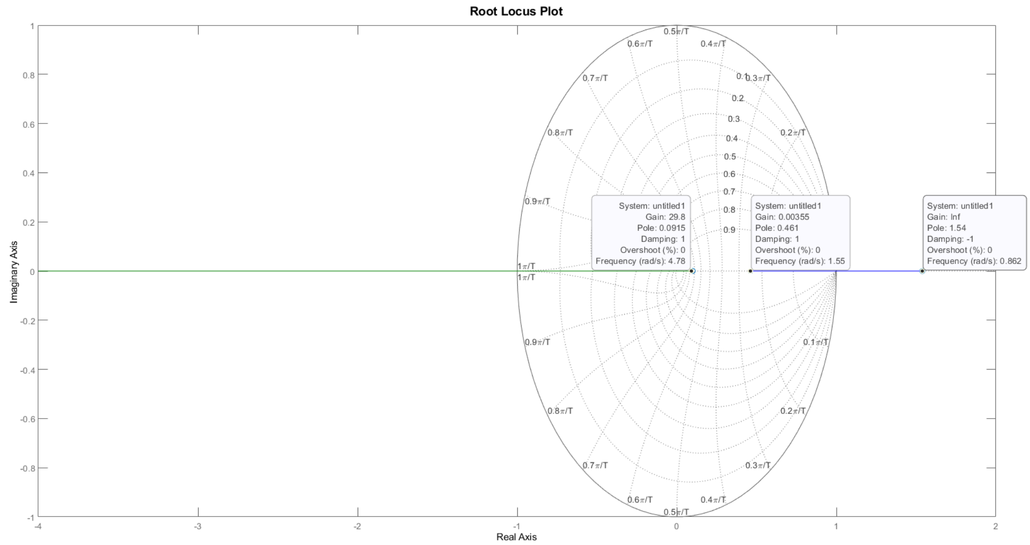


Figure 20: System root locus plot.

The weighted controller J2 is similar to the previous section. According to the root locus plot, the system is stable for all values of λ. Therefore, λ is set to 2.

3.3.1 Results Without Noise (MPC J2 Loss Function)

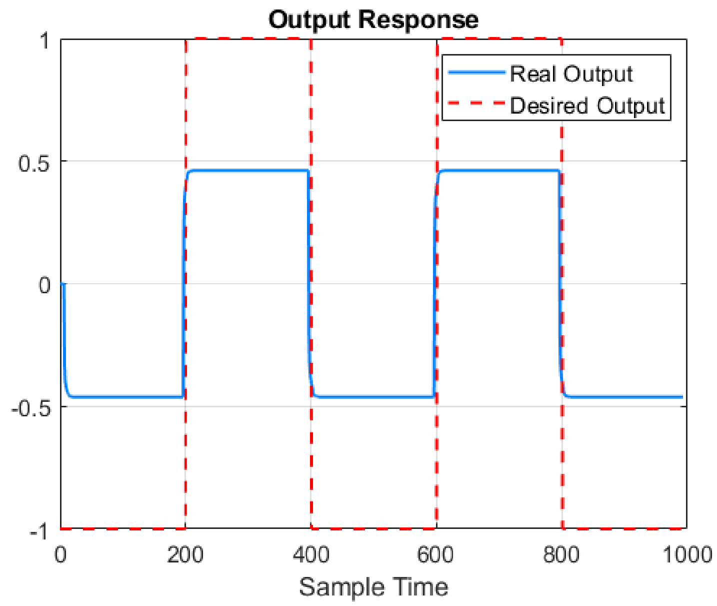


Figure 21: System Response Vs. system input (MPC J2 Loss Function, without noise).

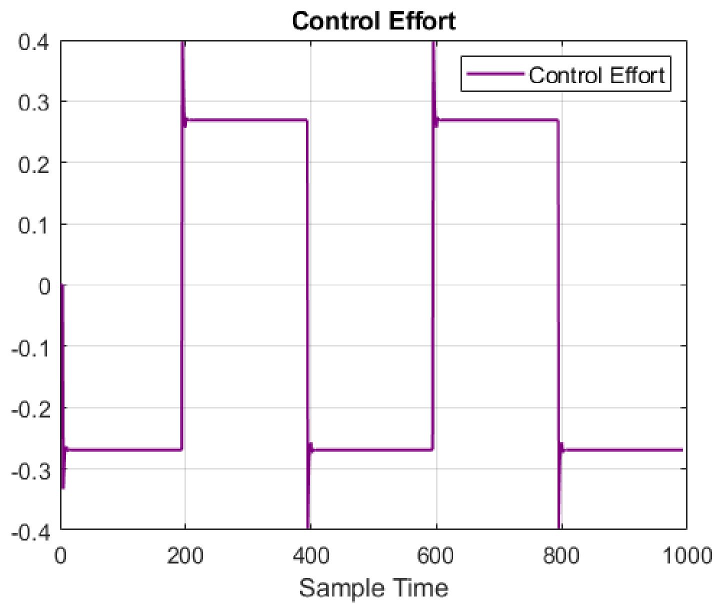


Figure 22: System control effort signal (MPC J2 Loss Function, without noise).

3.3.2 Results With White Noise $\sigma^2 = 0.001$ (MPC J2 Loss Function)

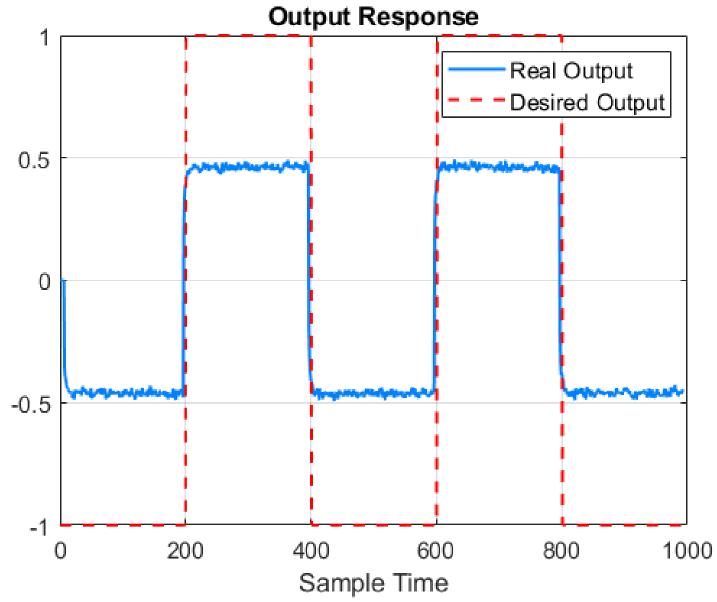


Figure 23: System Response Vs. system input (MPC J2 Loss Function, White Noise $\sigma^2 = 0.001$).

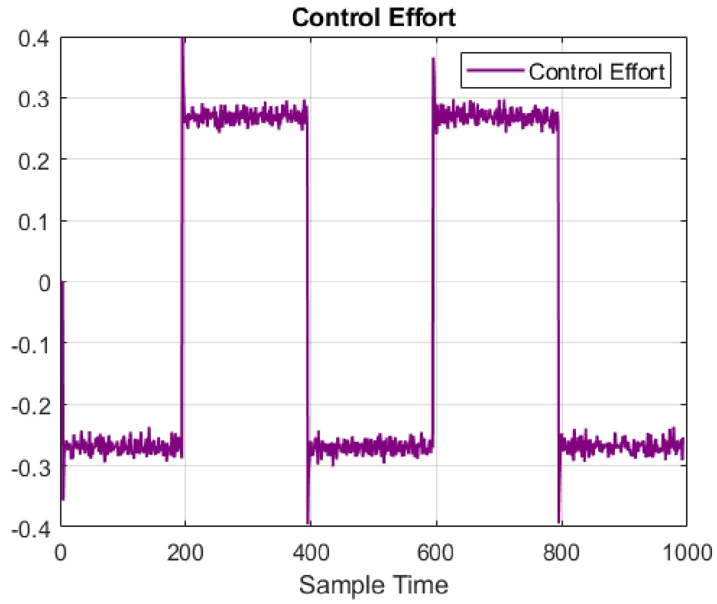


Figure 24: System control effort signal (MPC J2 Loss Function, White Noise $\sigma^2 = 0.001$).

3.3.3 Results With White Noise And Disturbance (MPC J2 Loss Function)

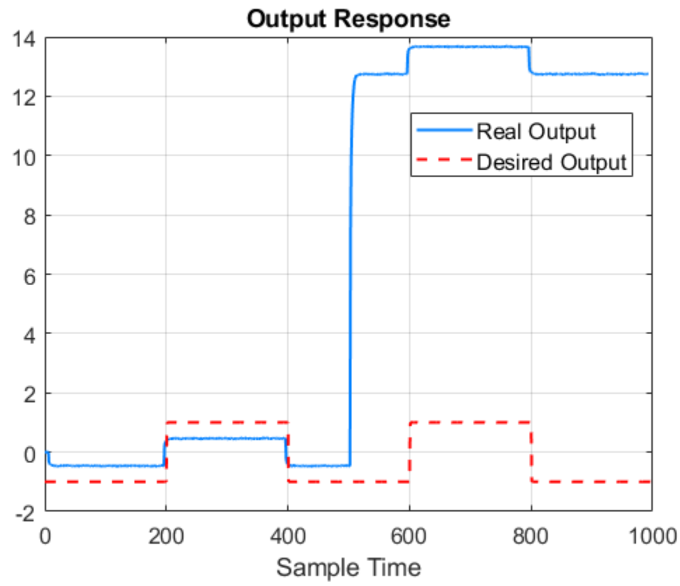


Figure 25: System Response Vs. system input (MPC J2 Loss Function, White Noise And Disturbance).

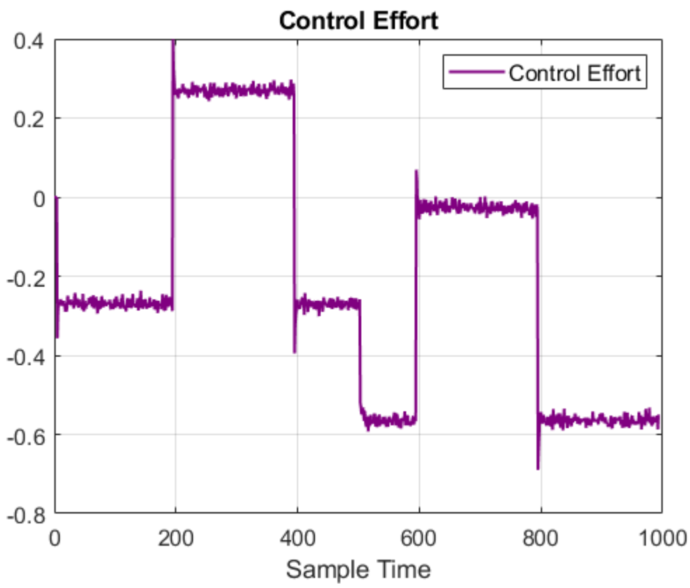


Figure 26: System control effort signal (MPC J2 Loss Function, White Noise And Disturbance).

3.3.4 Results For Changing Delay Of The System Without Noise (MPC J2 Loss Function)

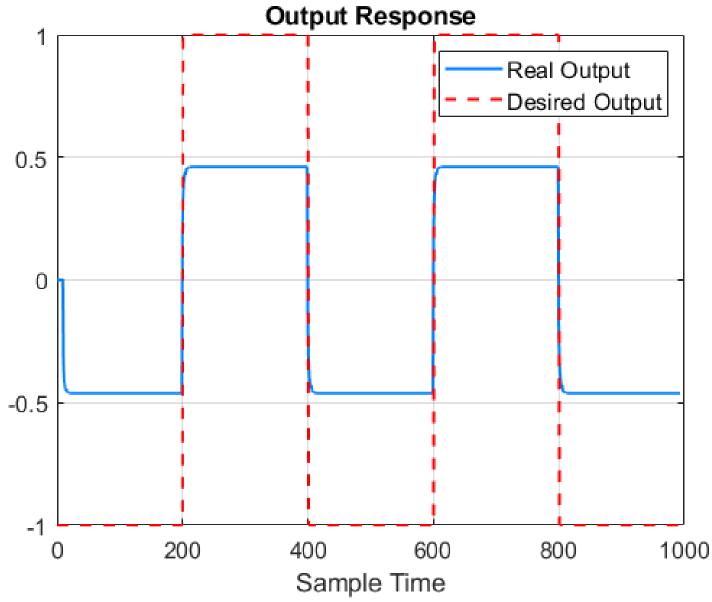


Figure 27: System Response Vs. system input (MPC J2 Loss Function, Changing Delay).

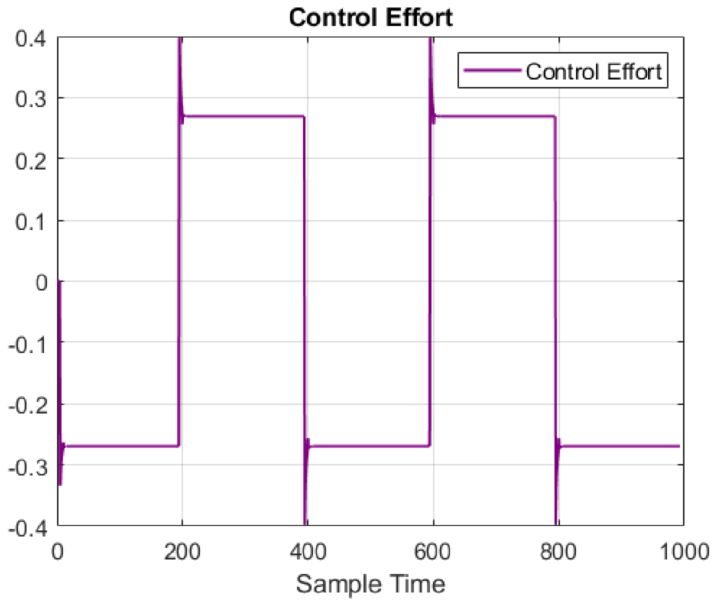


Figure 28: System control effort signal (MPC J2 Loss Function, Changing Delay).

3.3.5 Conclusion On The Results (MPC J2 Loss Function)

As observed, using the weighted one-step-ahead approach with the added λ significantly improves control effort. However, this comes at the cost of worsened tracking performance, introducing an offset error between the system and the desired response.

In the previous scenario, without the inclusion of λ, the system did not exhibit desirable transient behavior. With the addition of λ, the system's performance has improved. Increasing the value of λ can further enhance the response. As observed, the system has stabilized and the output is close to the desired value. Compared to previous cases, the system is more affected by disturbances due to the larger value of λ considered, which amplifies the disturbance's impact on the system.

Interestingly, with a larger value of λ in a non-minimum phase system, the output remained stable even without delay. The system's output in such a case is shown below, with d set to 6. There are two instances of mismatch, causing λ to reach the desired value at incorrect times. Increasing λ will result in a higher steady-state error, but overall, the system will exhibit better behavior.

3.4 MPC J3 Loss Function (Non Minimum Phase System)

In this case, the control command can be obtained using the following code snippet:

```
% Filtering
P = [1];
R = [1, -1];
pr_transfer = tf(R, P, Ts);

control_u(i) = (beta_0 * (desired_output(i+d) - filter_signal(alpha, output_y, i) - filter_signal(beta_prime, control_u, i-1)) + landa * filter_signal(P(2:end), u_bar, i-1) - landa * filter_signal(R(2:end), control_u, i-1)) / (beta_0^2 + landa);

u_bar(i) = (filter_signal(R, control_u, i) - filter_signal(P(2:end), u_bar, i-1)) / P(1);
```

Here, we chose P and R in such a way that an integrator is added to the system. Under these conditions, the steady-state error of the system to a step input will be zero. The root locus plot of the augmented system is obtained as follows:

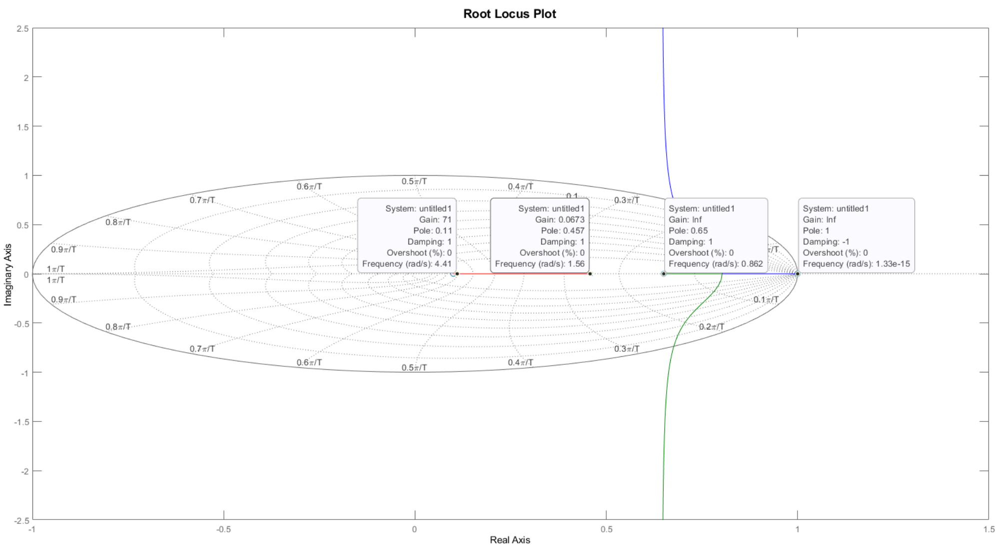


Figure 29: System root locus plot.

As observed, for values of λ greater than 1, the system will be stable. In this section, λ is set to 5.

3.4.1 Results Without Noise (MPC J3 Loss Function)

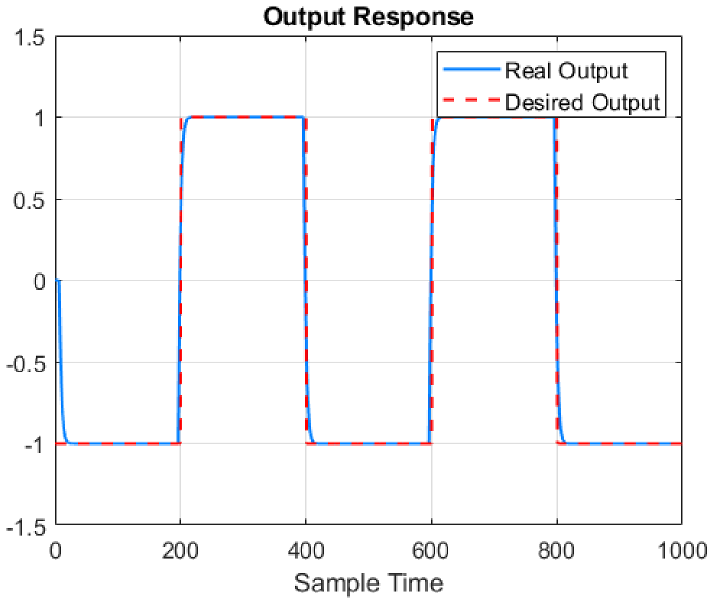


Figure 30: System Response Vs. system input (MPC J3 Loss Function, without noise).

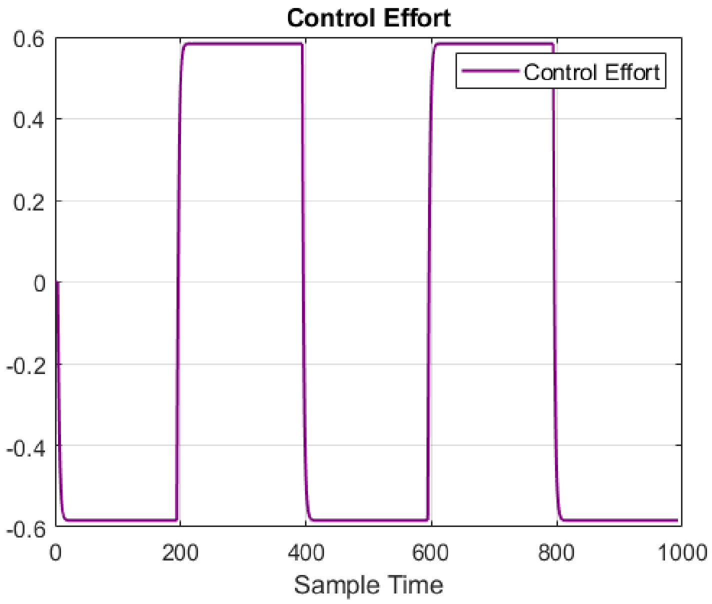


Figure 31: System control effort signal (MPC J3 Loss Function, without noise).

3.4.2 Results With White Noise $\sigma^2 = 0.001$ (MPC J3 Loss Function)

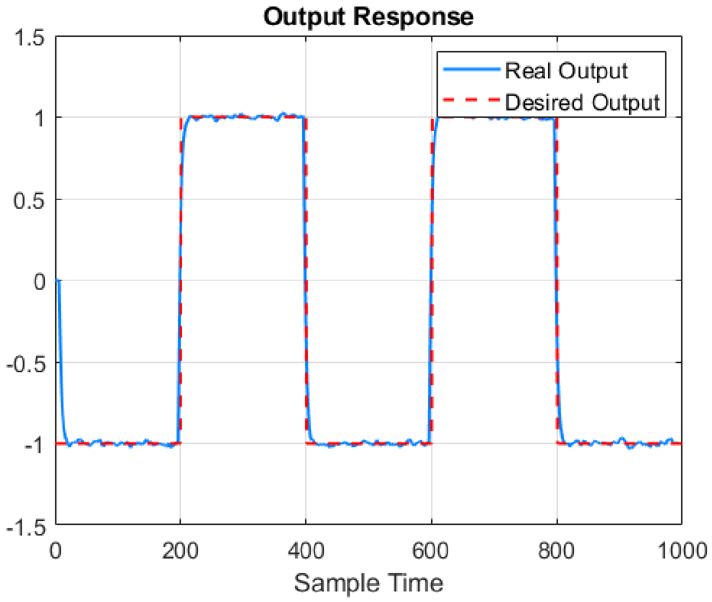


Figure 32: System Response Vs. system input (MPC J3 Loss Function, White Noise $\sigma^2 = 0.001$).

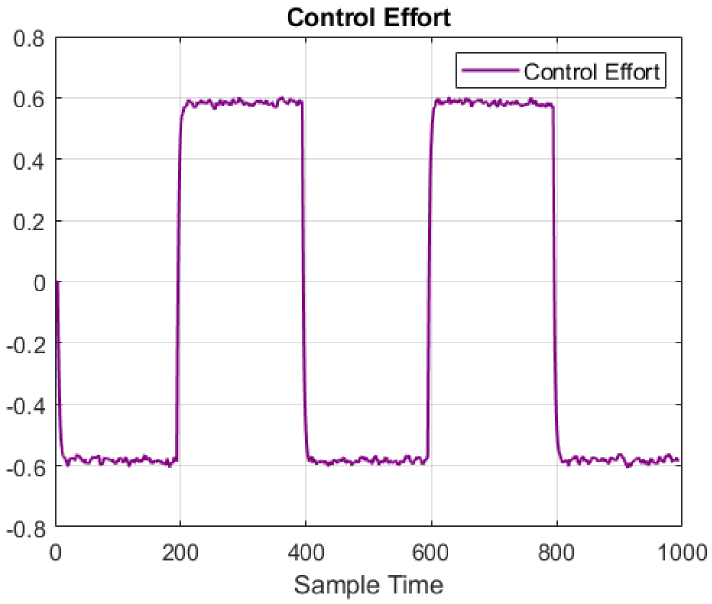


Figure 33: System control effort signal (MPC J3 Loss Function, White Noise $\sigma^2 = 0.001$).

3.4.3 Results With White Noise And Disturbance (MPC J3 Loss Function)

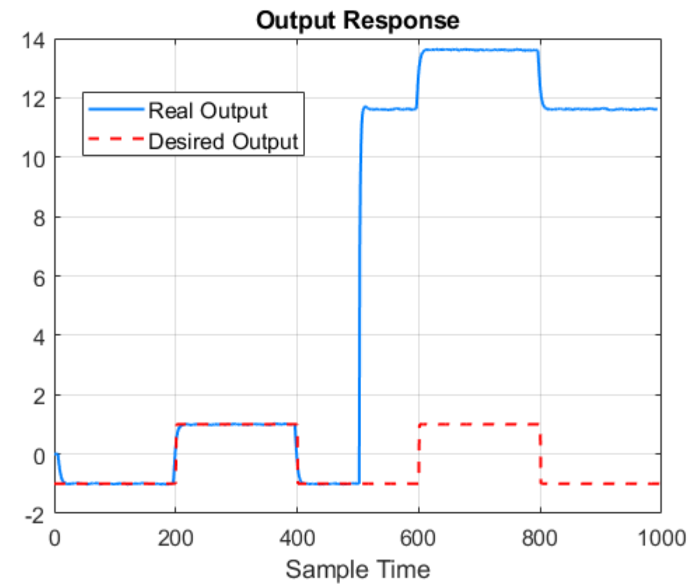


Figure 34: System Response Vs. system input (MPC J3 Loss Function, White Noise And Disturbance).

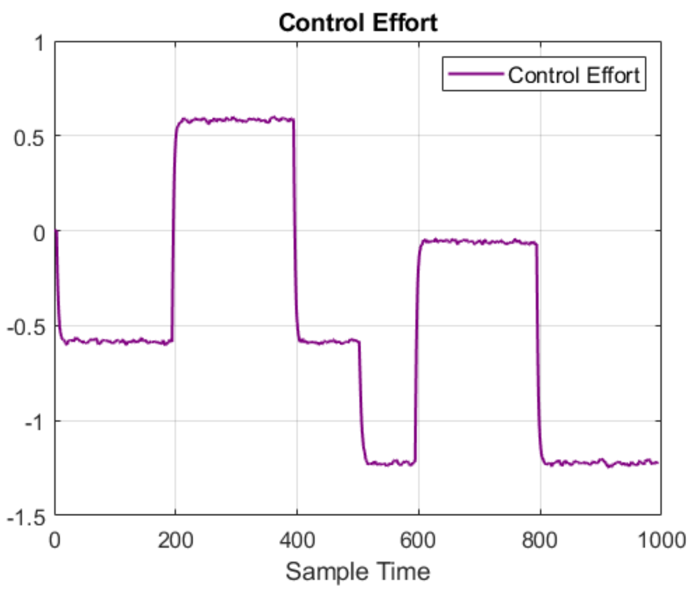


Figure 35: System control effort signal (MPC J3 Loss Function, White Noise And Disturbance).

3.4.4 Results For Changing Delay Of The System Without Noise (MPC J3 Loss Function)

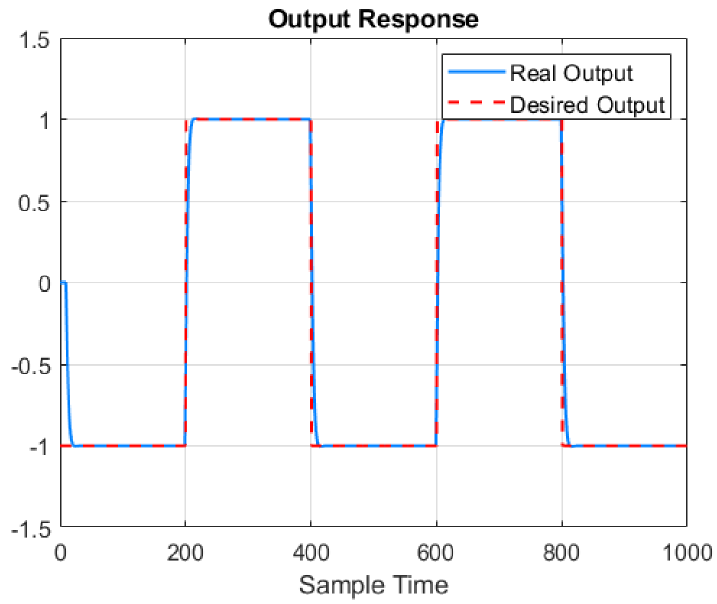


Figure 36: System Response Vs. system input (MPC J3 Loss Function, Changing Delay).

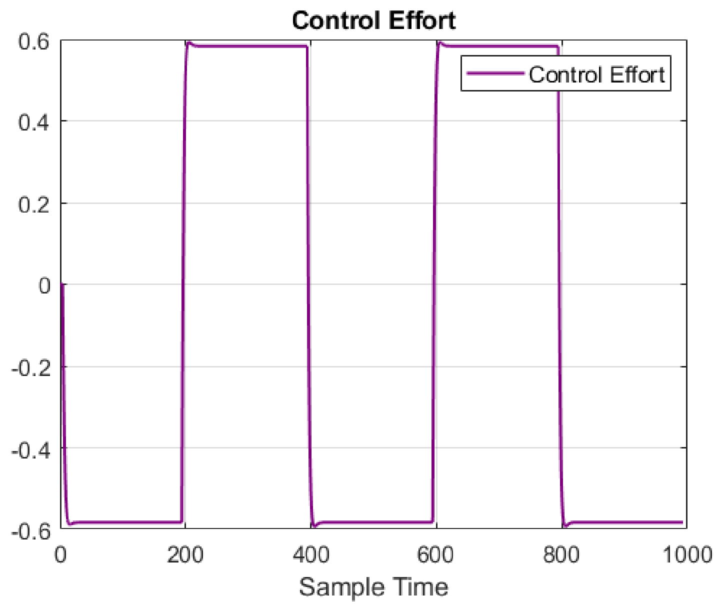


Figure 37: System control effort signal (MPC J3 Loss Function, Changing Delay).

3.4.5 Conclusion On The Results (MPC J3 Loss Function)

The addition of an integrator has increased the impact of noise because the integrator accumulates error. Since the controller cannot mitigate noise, this error also gets amplified. Therefore, it is better to adjust P and R to eliminate the effect of noise as well.

As expected, the disturbance has not been mitigated here either, and the system has performed poorly.

Due to the incorrect consideration of delay in the controller, it started changing the control input 4 steps ahead.

3.5 Constant Future MPC

Here, the prediction horizon is considered equal to twice the system delay. The Diophantine equation for calculating G and F is the same as the previous equations. After calculating G and F, it is also necessary to calculate the values of R and $\bar{R}$ as shown below:

$$B^*(q^{-1})F_d^*(q^{-1}) = R_d^*(q^{-1}) + q^{-(d-d_0+1)}\bar{R}_d^*(q^{-1})$$

In this equation, the term $d - d_0 + 1$ is written separately from the left-hand side, and R and $\bar{R}$ are placed next to the $n-1$ term. Therefore, the degrees of these two polynomials are obtained as follows:

$$\deg\ R_d^* = d - d_0$$

$$\deg\ \bar{R}_d = n - 2$$

Finally, the control relation is obtained as follows:

$$u(t) = \frac{y_m(t+d) - G_d^*(q^{-1})y(t)}{R_d^*(q^{-1}) + \bar{R}_d^*(q^{-1})q^{-1}}$$

3.5.1 Results Without Noise (Constant Future MPC)

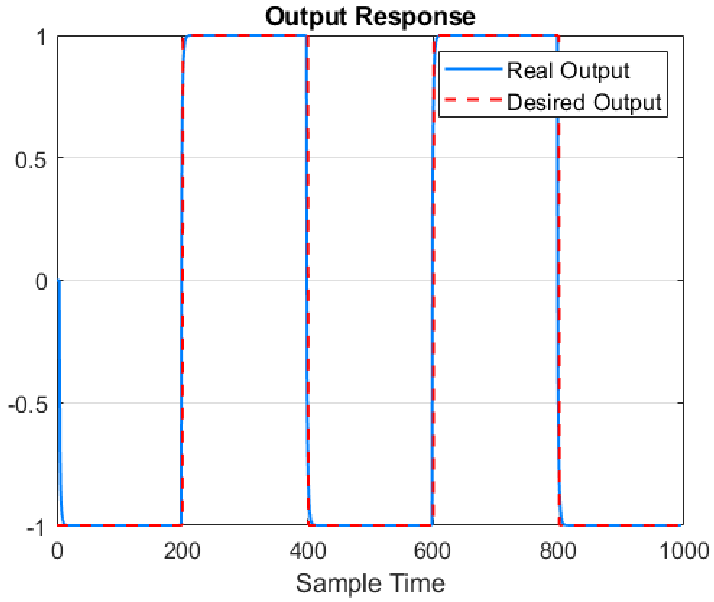


Figure 38: System Response Vs. system input (Constant Future MPC, without noise).

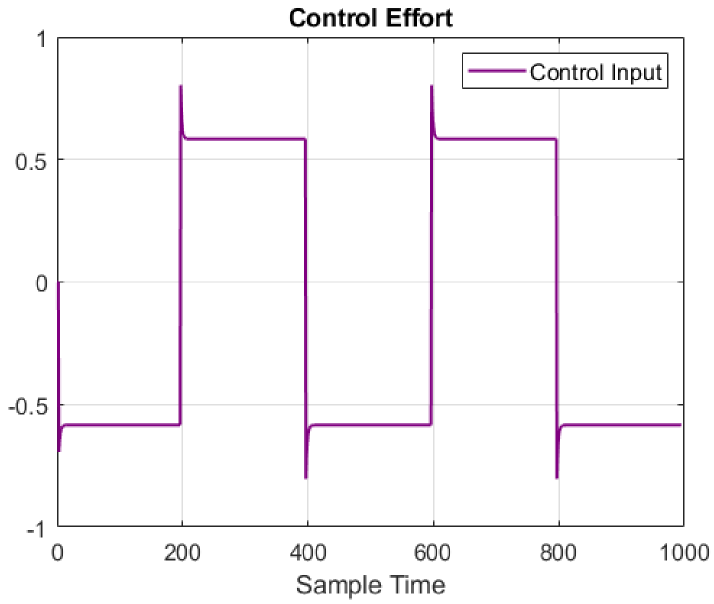


Figure 39: System control effort signal (Constant Future MPC, without noise).

3.5.2 Results With White Noise $\sigma^2 = 0.001$ (Constant Future MPC)

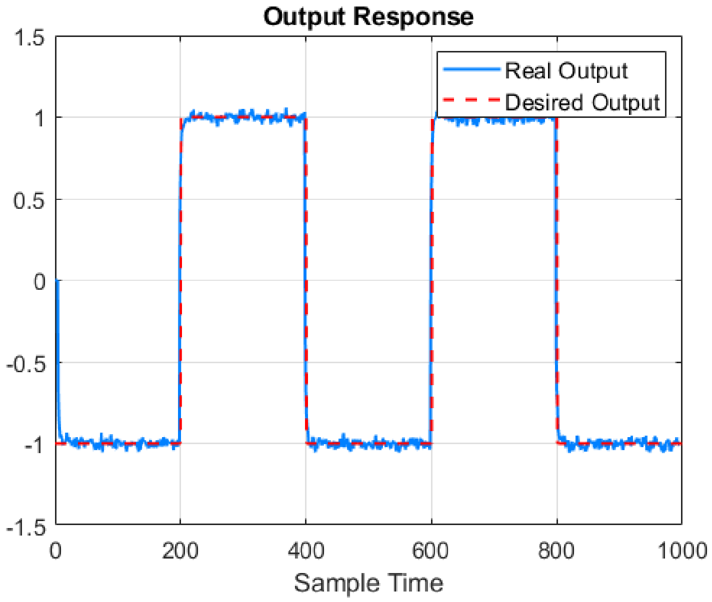


Figure 40: System Response Vs. system input (Constant Future MPC, White Noise $\sigma^2 = 0.001$).

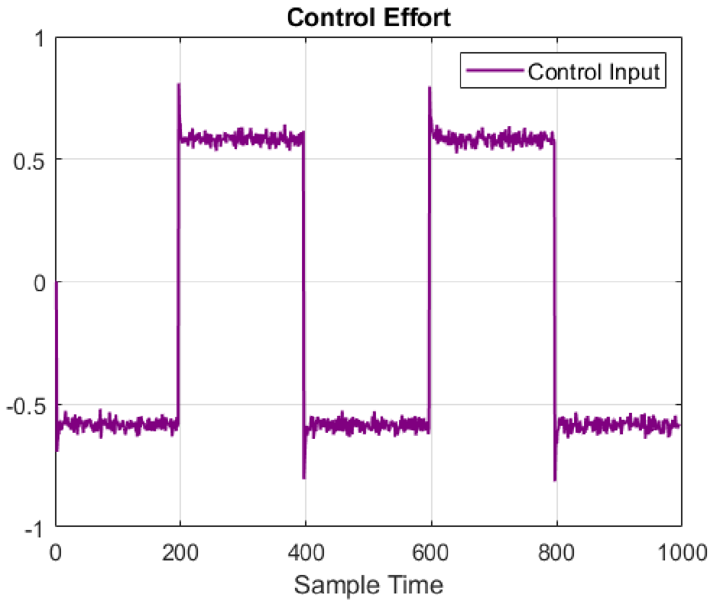


Figure 41: System control effort signal (Constant Future MPC, White Noise $\sigma^2 = 0.001$).

3.5.3 Results With White Noise And Disturbance (Constant Future MPC)

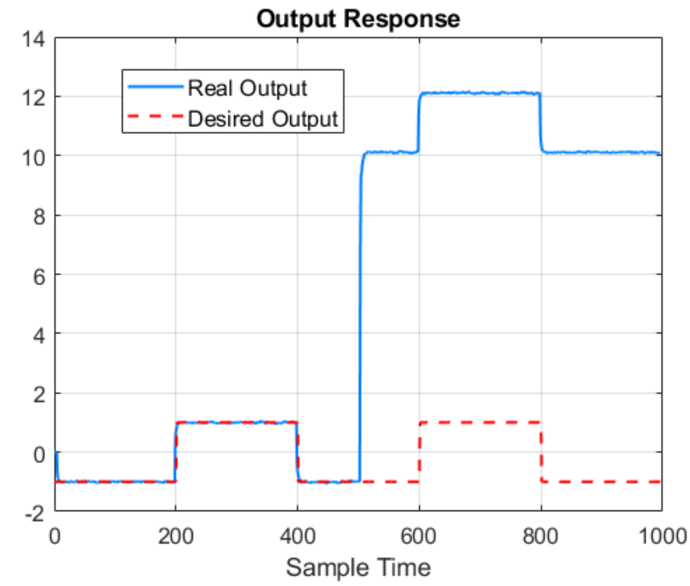


Figure 42: System Response Vs. system input (Constant Future MPC, White Noise And Disturbance).

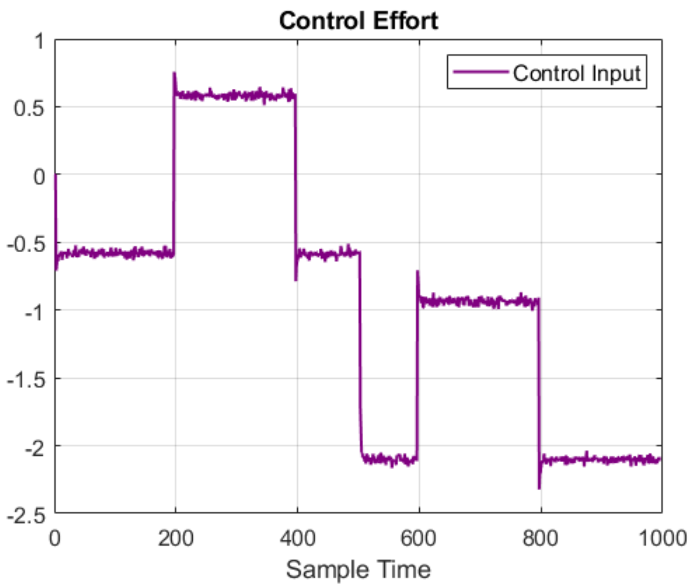


Figure 43: System control effort signal (Constant Future MPC, White Noise And Disturbance).

3.5.4 Conclusion On The Results (MPC J3 Loss Function)

As observed, the control system has effectively followed the desired input. Additionally, the control input is relatively small. However, this controller is also unable to mitigate system noise.

Regarding the fixed horizon controller, the issue of delay mismatch is meaningless, as this alignment is inherently absent. Therefore, in this case, the mismatch issue was not examined.

3.6 Conclusion On All MPC Methods (Non-Adaptive)

Method	Steady State Error	Noise Rejection	Disturbance Rejection	Unmatched Delay Sensitivity
One Step Ahead	No	No	No	Yes
Weighted One Step Ahead (J1)	Yes	No	No	No
Weighted One Step Ahead (J3)	No	No	No	No
Constant Future	No	No	No	No

Table 1: Comparing the MPC methods performances.

The table compares various Model Predictive Control (MPC) methods based on their handling of steady-state error, noise rejection, disturbance rejection, and sensitivity to unmatched delays.

- **One Step Ahead:** This method does not handle steady-state error, noise rejection, or disturbance rejection well and is sensitive to unmatched delays. It struggles with stability and performance under practical conditions involving noise and disturbances.

- **Weighted One Step Ahead (J1):** This approach eliminates steady-state error and is not sensitive to unmatched delays. However, it still fails to reject noise and disturbances, making it suitable in scenarios with consistent delays but not in noisy or disturbed environments.

- **Weighted One Step Ahead (J3):** Similar to the J1 method, it eliminates steady-state errors and is not sensitive to unmatched delays, but does not improve noise and disturbance rejection.

- **Constant Future:** This method shows poor performance across all metrics, indicating limited practical applicability.

In summary, while some methods effectively handle steady-state errors and delays, none adequately address noise and disturbance rejection, suggesting the need for further improvements.

4 Estimating Delay Of The System (Adaptive)

A common strategy for estimating the system delay is to perform an initial identification during the first few time steps and consider the initial elements that are zero as the system delay. However, this approach can cause errors.

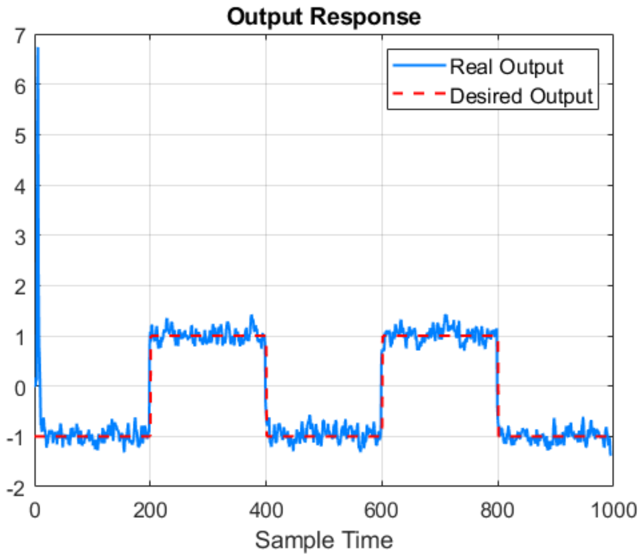


Figure 44: System Response Vs. system input (Estimating Delay)

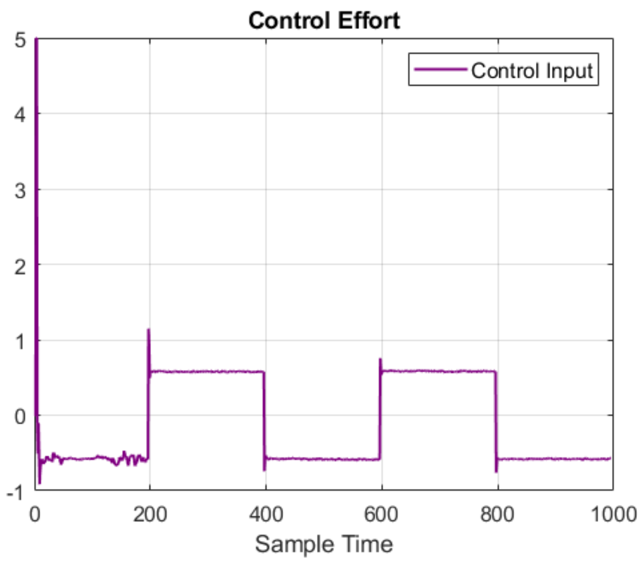


Figure 45: System control effort signal (Estimating Delay)

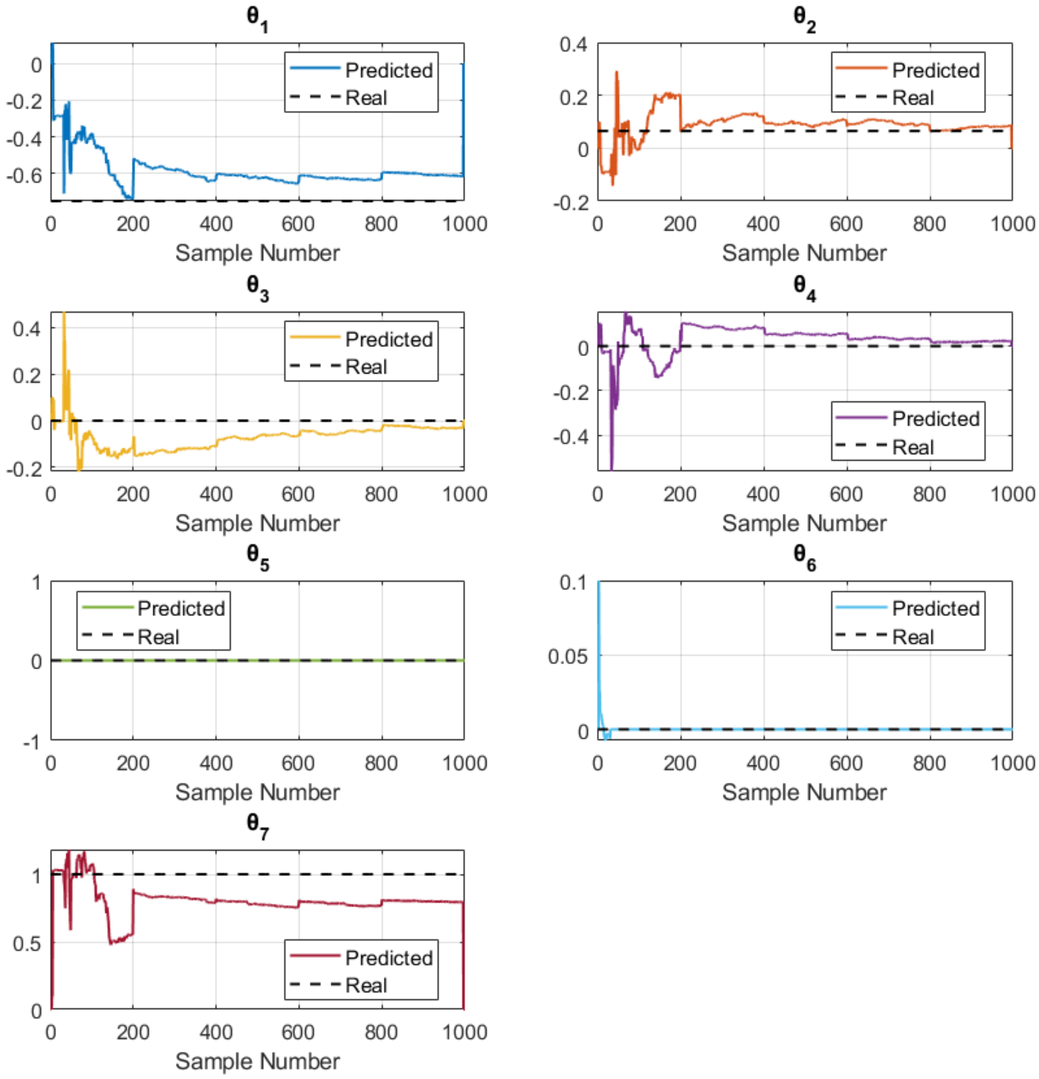


Figure 46: System parameters estimation (Estimating Delay)

5 Model Predictive Controller (Adaptive)

In this Section we will implement all the MPC methods that were implemented in the previous sections with system identification which means the algorithms are adaptive.

5.1 One Step Ahead Method (Adaptive)

For delay estimation, instead of separating the estimation and controller sections, these two processes occur simultaneously. A range is considered within which the estimated parameters B are examined. If the average value of the parameter is below a certain threshold within this range, this value is considered the system delay. Afterward, the identification system is updated, and one of its parameters is removed. Additionally, the controller's delay values are also adjusted.

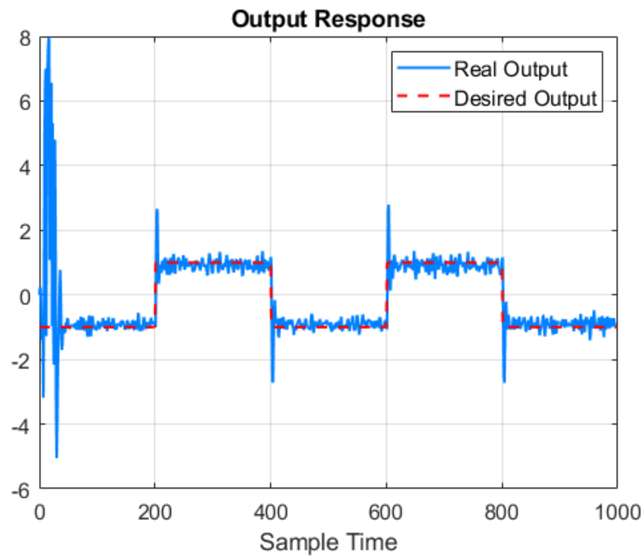


Figure 47: System Response Vs. system input (One Step Ahead, Adaptive)

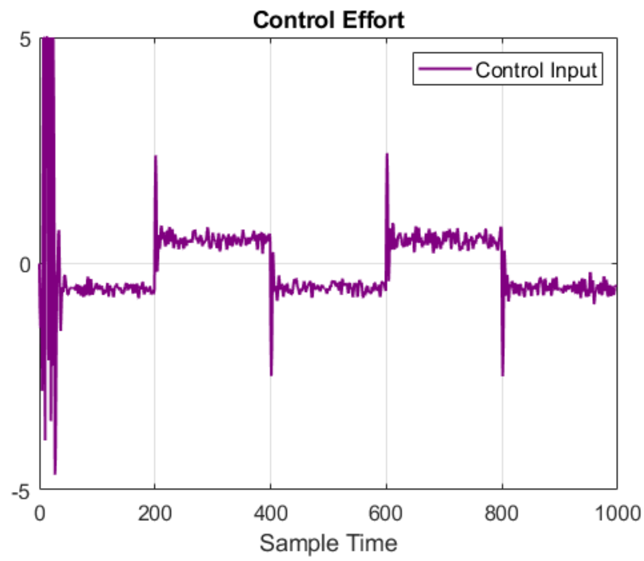


Figure 48: System control effort signal (One Step Ahead, Adaptive)

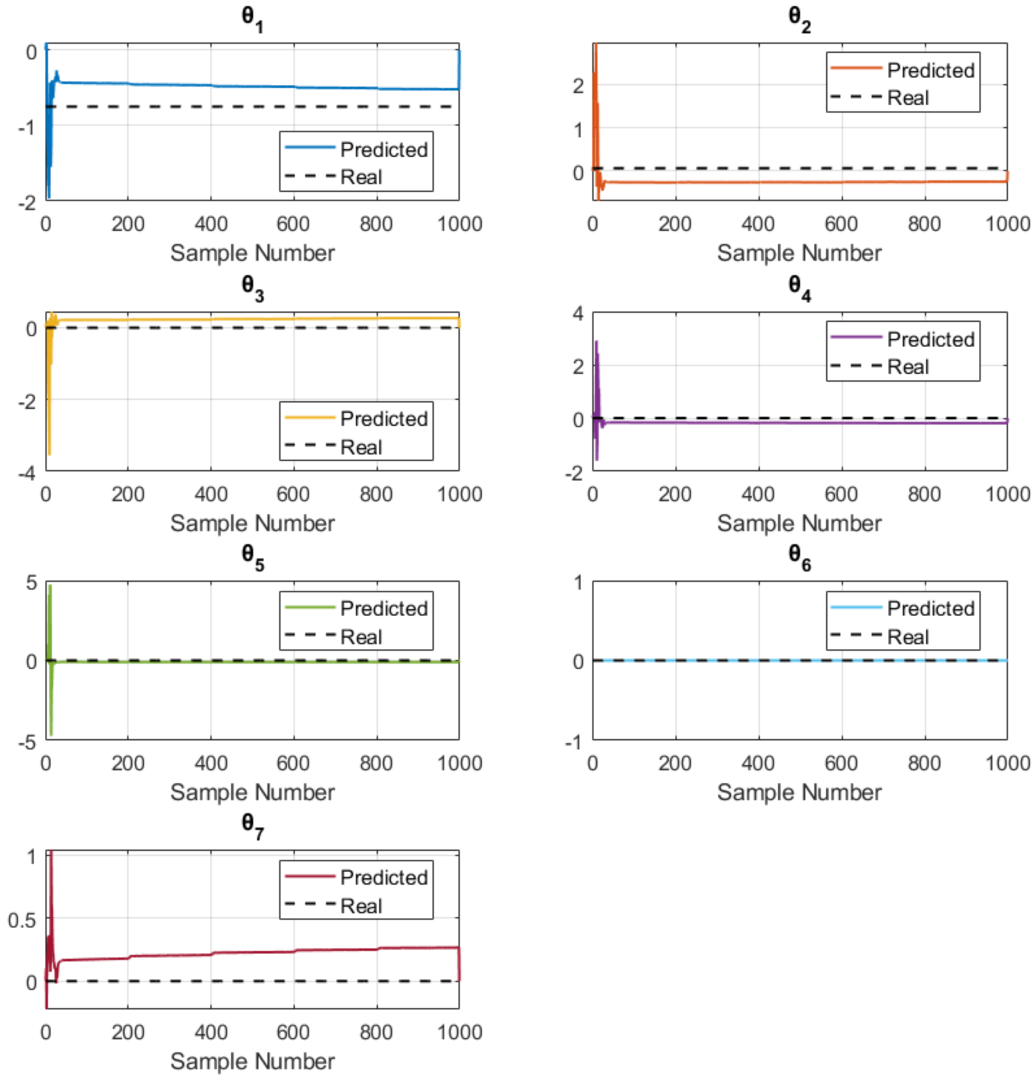


Figure 49: System parameters estimation (One Step Ahead, Adaptive)

5.2 Weighted One Step Ahead Method (Adaptive)

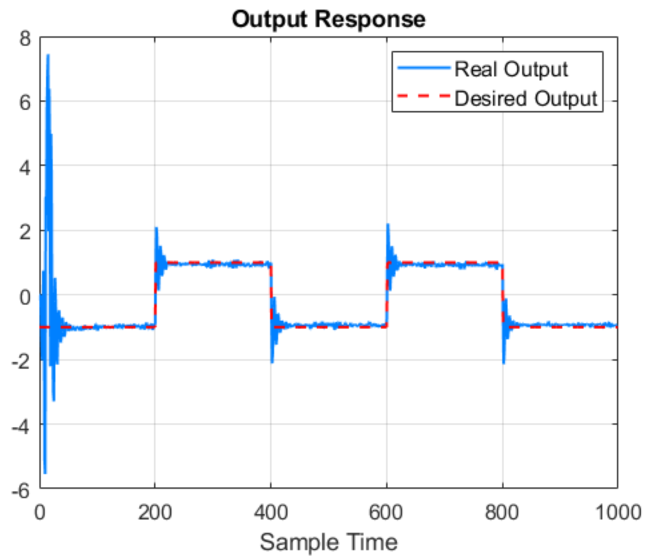


Figure 50: System Response Vs. system input (Weighted One Step Ahead, Adaptive)

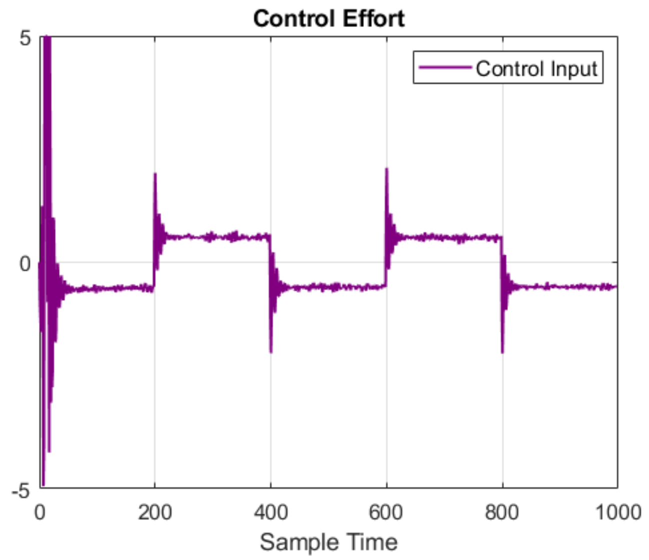


Figure 51: System control effort signal (Weighted One Step Ahead, Adaptive)

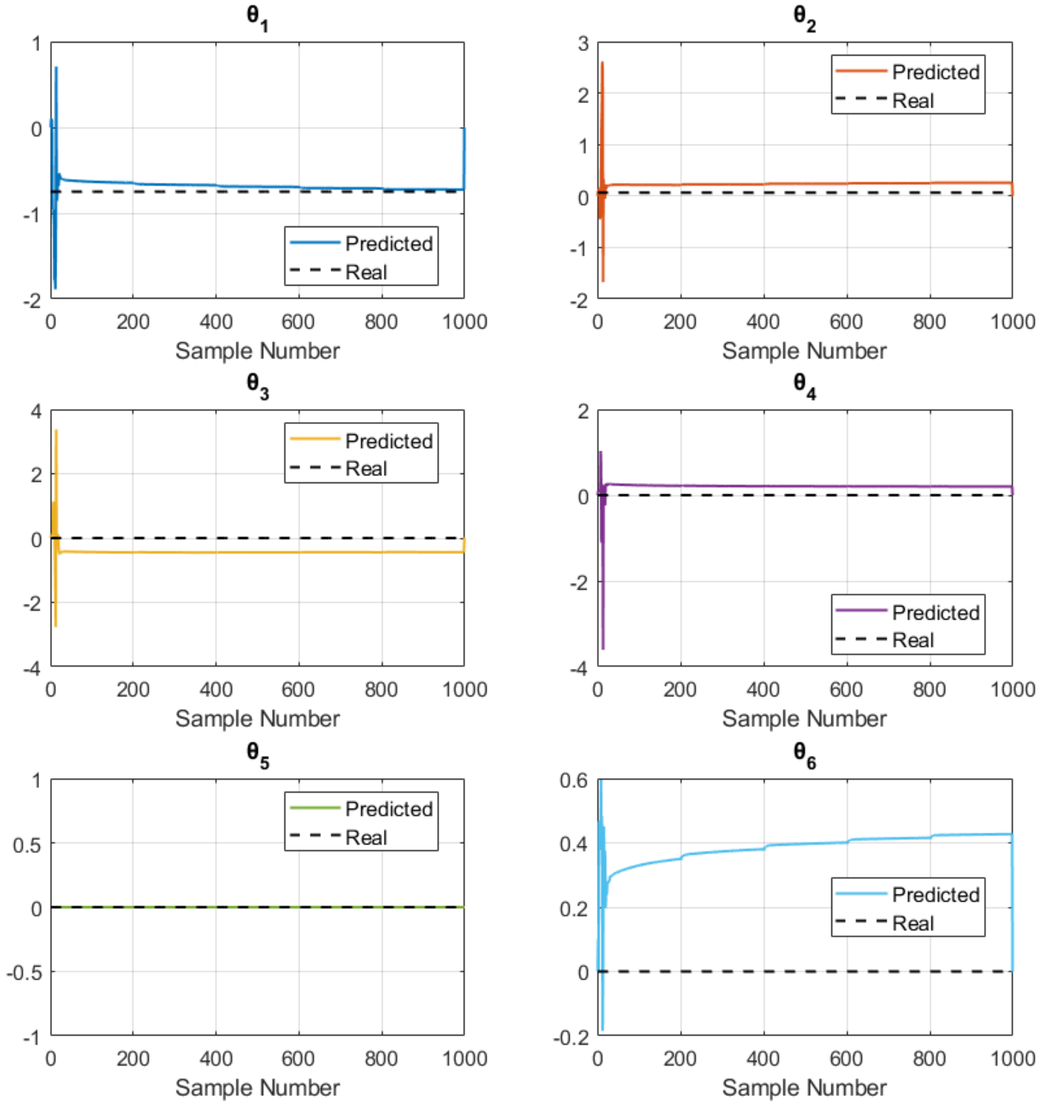


Figure 52: System parameters estimation (Weighted One Step Ahead, Adaptive)

5.3 MPC J2 Loss Function (Non Minimum Phase, Adaptive)

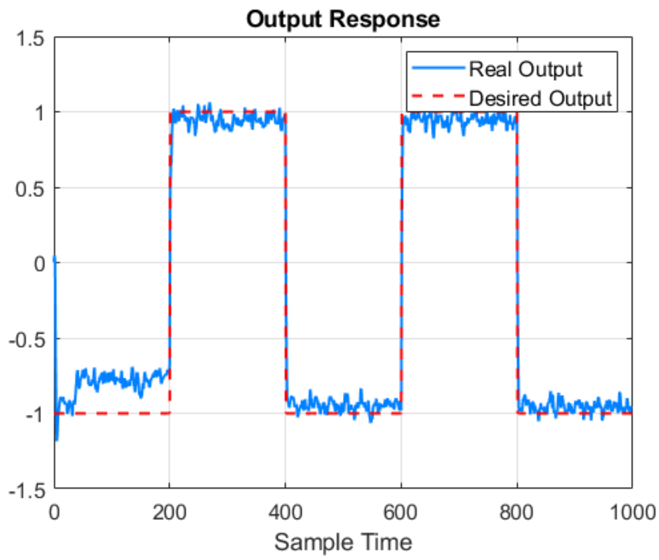


Figure 53: System Response Vs. system input (Weighted One Step Ahead, Adaptive)

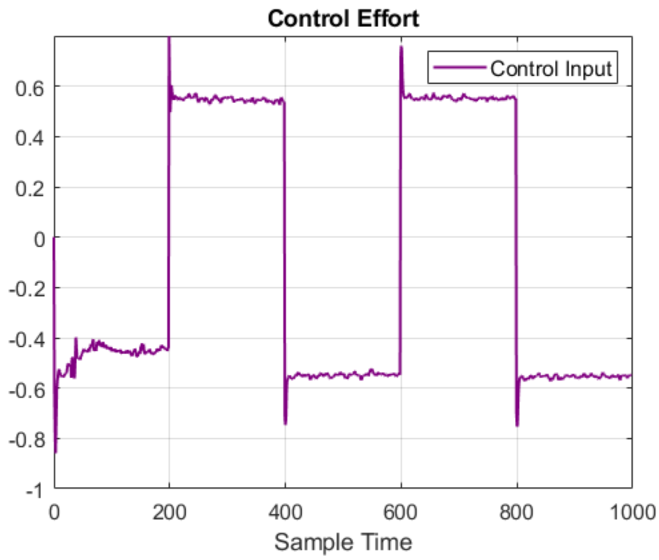


Figure 54: System control effort signal (MPC J2 Loss Function (Non Minimum Phase, Adaptive))

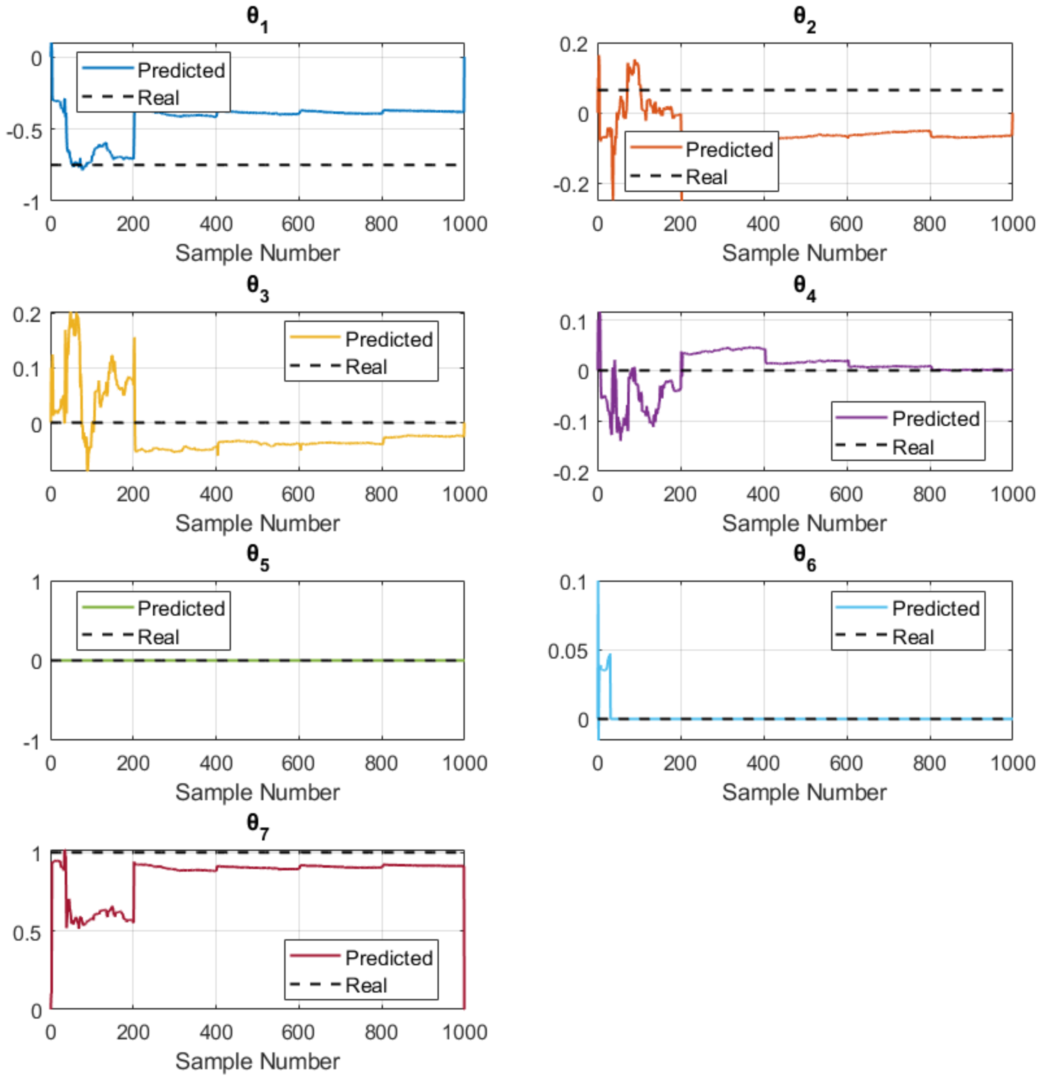


Figure 55: System parameters estimation (MPC J2 Loss Function (Non Minimum Phase, Adaptive))

5.4 MPC J3 Loss Function (Non Minimum Phase, Adaptive)

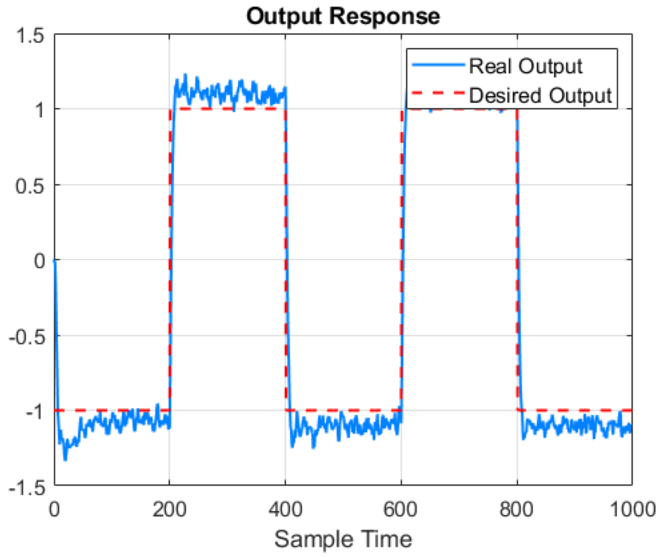


Figure 56: System Response Vs. system input (MPC J3 Loss Function (Non Minimum Phase, Adaptive))

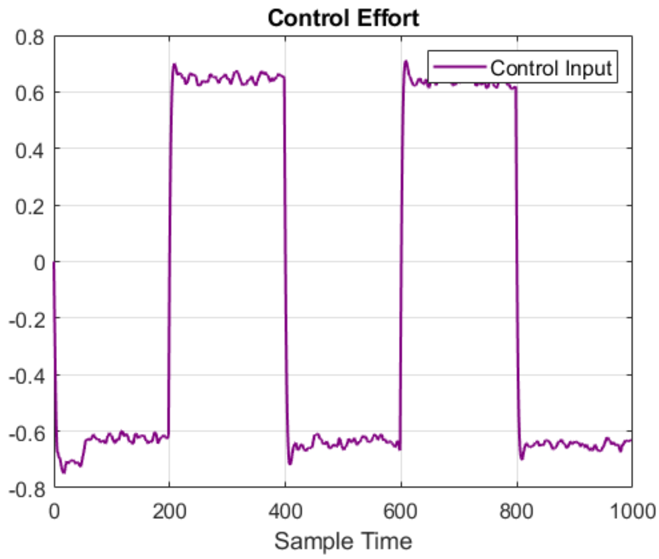


Figure 57: System control effort signal (MPC J3 Loss Function (Non Minimum Phase, Adaptive))

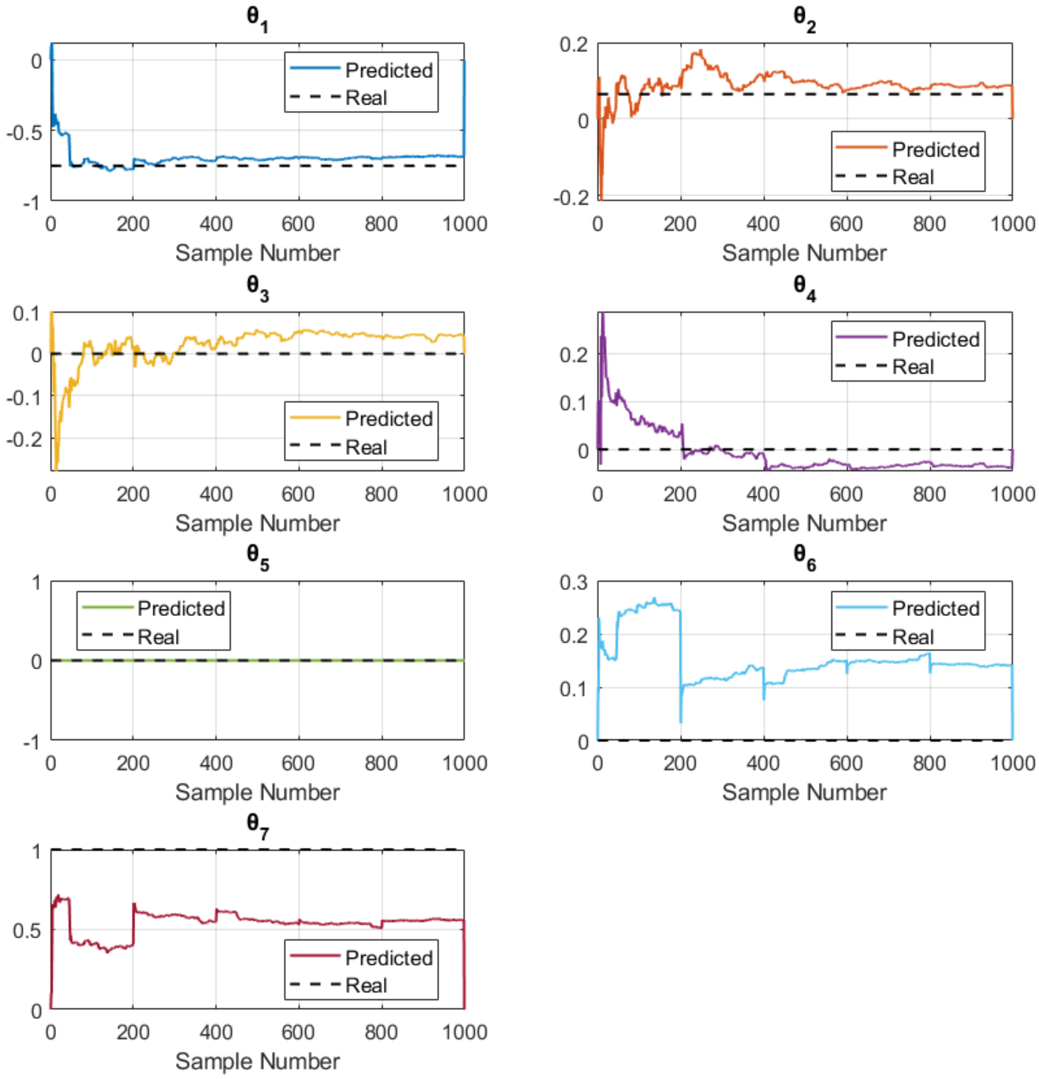


Figure 58: System parameters estimation (MPC J3 Loss Function (Non Minimum Phase, Adaptive))

5.5 Constant Future MPC (Adaptive)

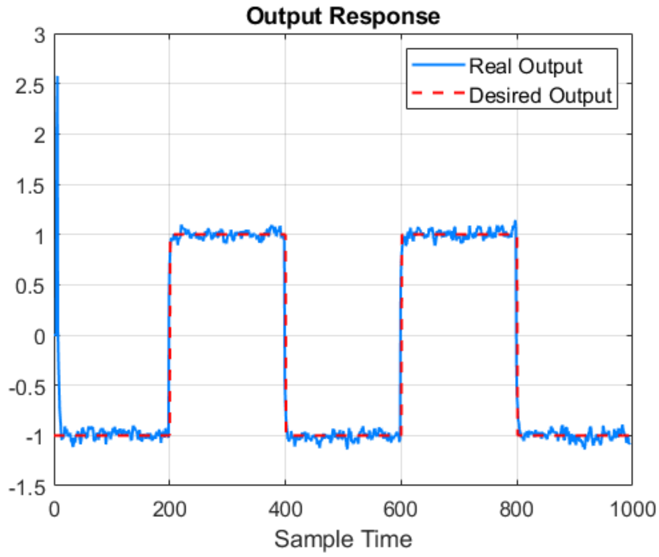


Figure 59: System Response Vs. system input (Constant Future MPC (Adaptive))

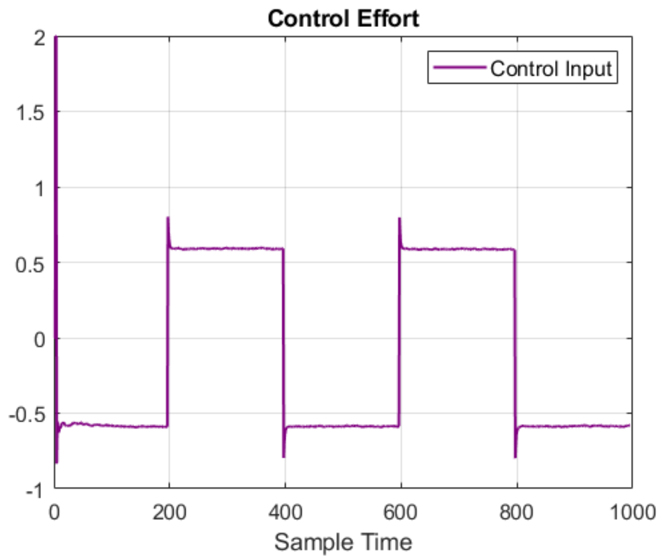


Figure 60: System control effort signal (Constant Future MPC (Adaptive))

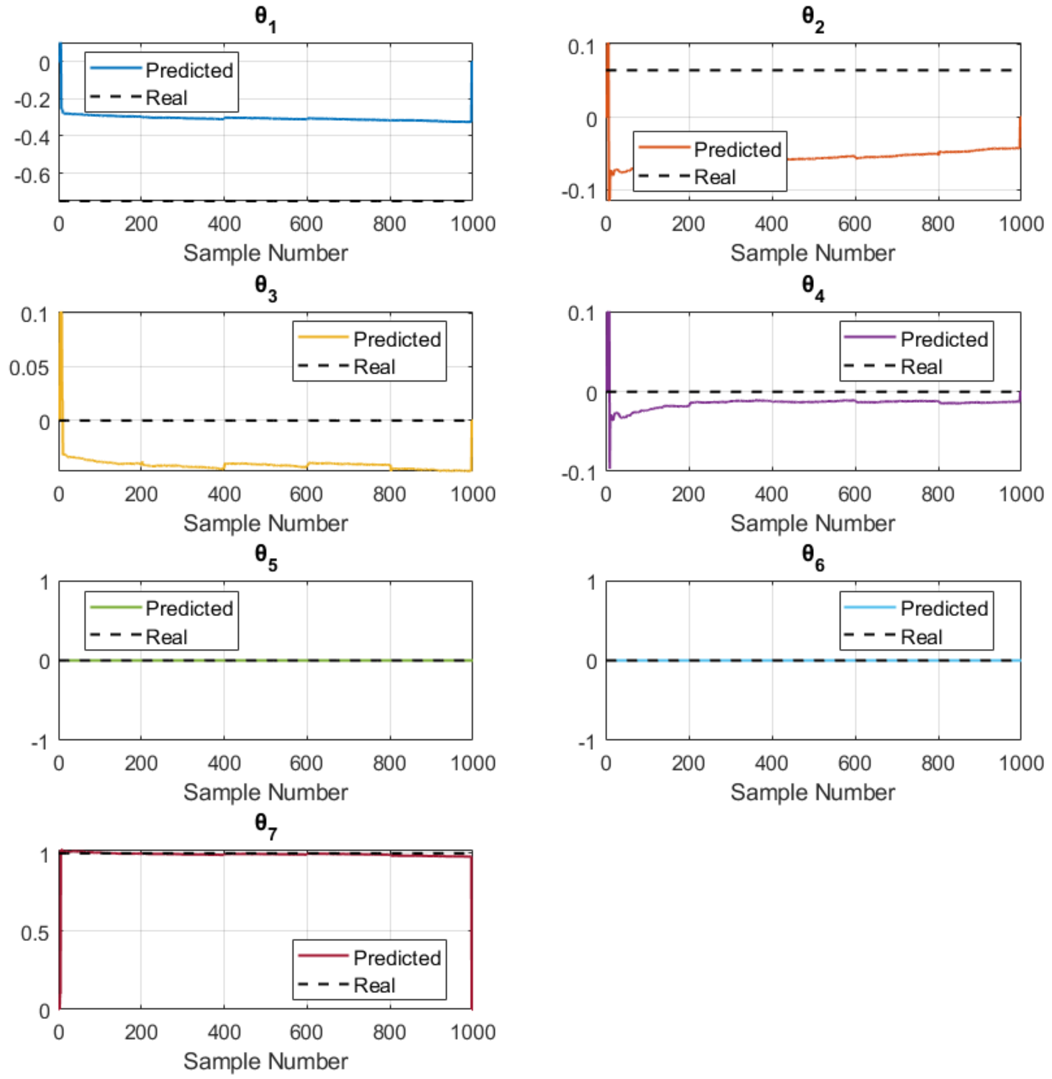


Figure 61: System parameters estimation (Constant Future MPC (Adaptive))

5.6 Conclusion On Adaptive MPC Methods

Method	Steady State Error	Noise Rejection	Disturbance Rejection	Unmatched Delay Sensitivity
One Step Ahead	High initially, converges	Poor	Poor	Sensitive
Weighted One Step Ahead	Improved, no saturation	Poor	Poor	Not sensitive
J2 MPC (Non Minimum Phase)	Initial high error, then stable	Poor	Poor	Not sensitive
J3 MPC (Non Minimum Phase)	Poor control, needs correction	Increased noise impact	Poor	Not sensitive
Constant Future MPC	Best performance	Good	Good	Not applicable

Table 2: Comparing the MPC methods performances.

- **One Step Ahead:** As observed, there is a significant error until time step 200, but after that, the error has relatively decreased and has almost converged to the actual values.

- **Weighted One Step Ahead:** The system's performance has improved, and due to the limitation of u, saturation was not necessary.

- **J2 MPC in Non Minimum Phase System:** The error in the value considered for d in the non-minimum phase system initially created a larger error, but this issue is resolved over time. The system stabilizes, and the outputs approach the desired value.

- **J3 MPC in Non Minimum Phase System:** P and Q are chosen in such a way that an integrator is added to the system. Additionally, λ is set to 10. However, the system is not well controlled because adding R effectively adds a degree to the denominator of the system. Therefore, after each estimation round, the estimated value of A should be divided by R. Despite correcting this issue, the system still exhibits undesirable behavior due to inaccurate initial estimation. Furthermore, the integrator has increased the impact of noise on the final response, affecting the system's performance.

- **Constant Future MPC:** This controller has the best performance in both control effort and tracking.