Control Engineering Project
Assignment 24

Zhaochen Xie 5712939

Q2 2022

Continuous-time Control

1 Tracking

1.1 P-controller

By observing the transfer function, one can see that there are three open-loop poles, which indicates that there is going to have also three closed-loop poles for no controller case. Now plot the root locus to see how these three closed-loop poles move while increasing the gain, as shown in Figure 1.1.

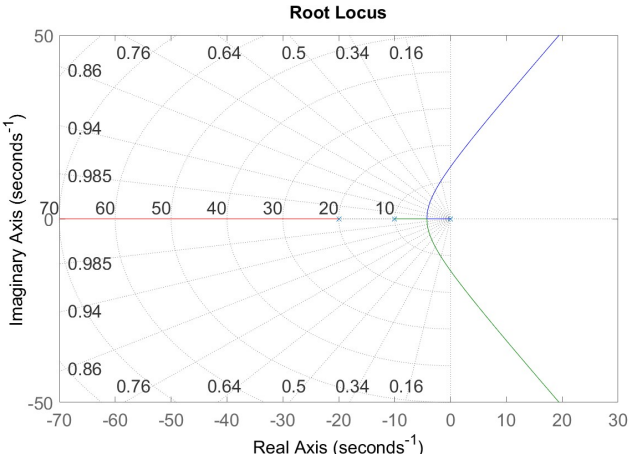


Figure 1.1: Root locus for the plant

It can be seen that for small gain, there would be 3 real negative poles, two of which are closer to the imaginary axis, and one is farther from the imaginary axis. For slightly larger gain, two of the poles will become a conjugated complex poles. With too large gain, poles with positive real part will be produced. This

gives us a rough intuition for the selection of P-parameter: Proportional gain should not be too big.

Since the plant itself contains an integrator $\frac{1}{s}$, a single P controller would let the closed-loop system satisfies the zero-steady-state-error requirement. Thus, we first try to use a P controller. Since we want to guarantee a fast response of the closed loop system, we want to have a pair of complex closed-loop poles near the imaginary axis. Also, since the third pole is far from the two complex poles, the dynamics it brings can be neglected temporarily. Now, we have a simplified second-order system, which could be analyzed by using the analytical method. We know that for a second-order system, the overshoot complies to the following relationship:

$$M_p\% = e^{-\frac{\xi}{\sqrt{1-\xi^2}}\pi} \tag{1.1}$$

which could be illustrated by Figure 1.2.

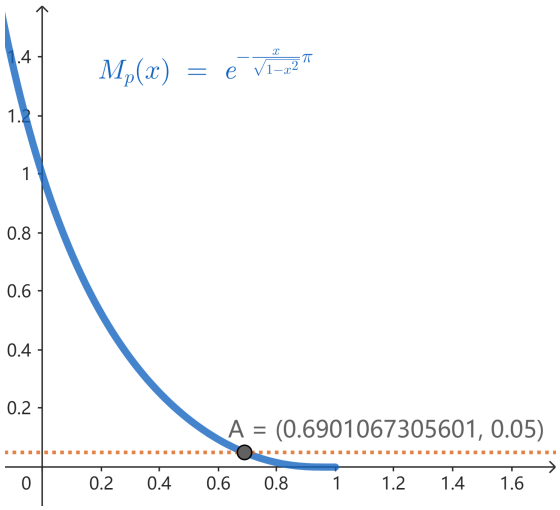


Figure 1.2: Analytical relationship between the damping ratio and overshoot

The intersection indicates that only when the closed loop damping ratio falls in the range of $[0.69, 1]$ can the requirement of overshoot smaller than 5% be satisfied. To show this constraint in a more intuitive way, a cone is drawn on the zoomed-out root locus, shown in Figure 1.3.

Besides the analytical relationship of overshoot, we also have the relationships for rise time t_r and settle time t_s for under-damped system as follows:

$$t_r = \frac{\pi - \beta}{\omega_d} \tag{1.2}$$

$$t_s = \frac{c}{\xi\omega_n} \tag{1.3}$$

$$t_s \geq t_r \tag{1.4}$$

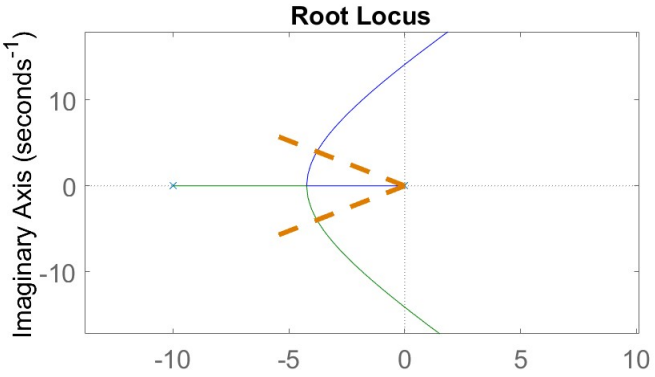


Figure 1.3: Overshoot requirement expressed on root locus

where $\beta = \arccos \xi$, ω_d is the imaginary part of the complex pole in the second quadrant, $\xi\omega_n$ is the real-part magnitude of the complex pole pair, c is the constant corresponding to 1% tolerance for the settling time. In this way, the requirement of minimal settling time can be interpreted as: under the premise of both that $t_s \geq t_r$ and complex poles are inside the orange cone, let t_r and t_s be as short as possible. Thus, according to eq.(1.2), the imaginary part should be as large as possible; meanwhile, according to eq.(1.3) the real part should be as large as possible. The above two requirements restrain the optimal solution on a small range of K_P, according to the root locus, roughly between 0.1 and 0.16. By tuning the P parameter in the constrained region, we got the optimal parameter for P controller: $K_P = 0.13$. By using this value, we have the minimal settling time of $0.9125s$ for 1% tolerance, overshoot of $0.9910\% < 5\%$, and zero steady state error. The step response of controlled system is shown in Figure 1.4.

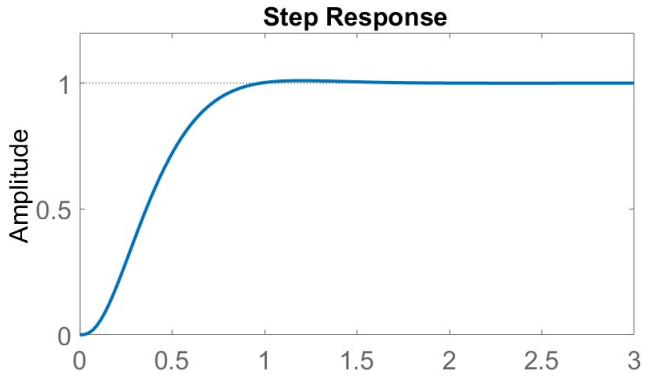


Figure 1.4: Step response with P-controller

1.2 PD-controller

By using a single P-controller, the optimal parameter renders the overshoot to be 0.9910%, which means there is a big margin to the requirement of 5% maximum overshoot. From this angle, we know that there is still improvement that can be made by adding another element to P-controller. Since the plant itself has an innate integrator, there is no need to add an I-element into the controller; otherwise, unnecessary oscillation would be brought to the closed-loop performance. Now we observe the bode diagram of the plant.

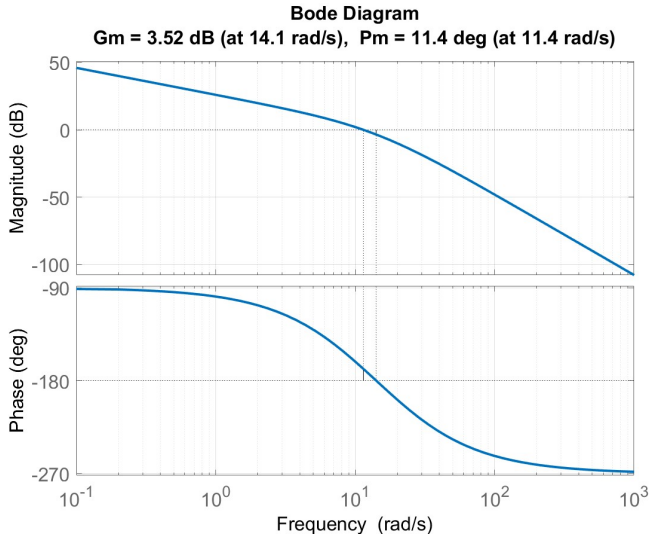


Figure 1.5: Bode diagram of the plant

Since we want to improve the transient performance, we could use a phase-lead compensator to increase the gain margin and phase margin.

The phase-lead compensator has the following form:

$$G_C(s) = K_P \frac{aTs + 1}{Ts + 1} \tag{1.5}$$

where $a > 1$ and T are parameters to be considered later. The bode diagram of this compensator without the term K_P is shown in Figure 1.6. Besides, the phase-lead compensator can also be written in the form of a realizable PD-controller with a first-order derivative filter:

$$G_{PDF}(s) = K_P + K_D \frac{s}{Ts + 1} \tag{1.6}$$

where the different parameters a and K_D have the relationship:

$$a = 1 + \frac{K_D}{TK_P} \tag{1.7}$$

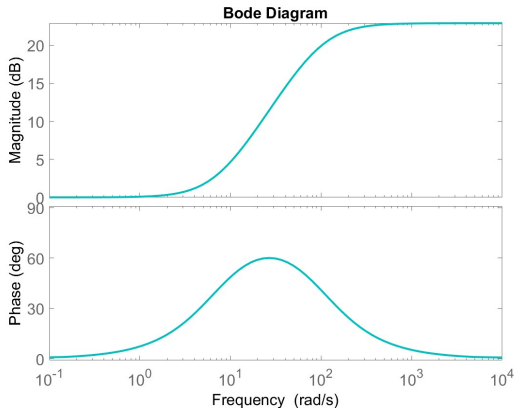


Figure 1.6: Bode diagram of phase-lead compensator

By observing eq.(1.6), we know that when T approaching 0, the derivative term becomes more ideal, which also means it gets harder to be realized. Thus, considering the cut-off frequency of the original plant, we choose $T = \frac{1}{100}$.

The next parameter to be determined is a. The analytical relationship between a and maximal phase-lead angle ϕ_m tells us that the value of a should be taken between $[4, 10]$. In this way, we choose $a = 10$. Using eq.(1.7) to calculate the relationship between K_P and K_D and use $K_P = 0.13$ as the starting point of tuning. After a few tuning processes and under the constraint of overshoot being less than 5%, we obtain the optimal parameters for the PD controller: $K_P = 0.48$,$K_D = 0.05$,$T = \frac{1}{100}$. Under this configuration, we obtain the settling time $t_s = 0.4067s$ and overshoot of value 4.66%, shown in Figure 1.7. The phase margin and gain margin both get larger, shown in Figure 1.8 , and the dynamic performance gets improved. Compare to proportional control, the settling time is reduced to almost 50% of the only P-controller case.

To sum up, the final controller for the tracking task is in the form:

$$G_{PDF}(s) = K_P + K_D \frac{s}{Ts + 1} \tag{1.8}$$

where $K_P = 0.48$, $K_D = 0.05$, and $T = \frac{1}{100}$.

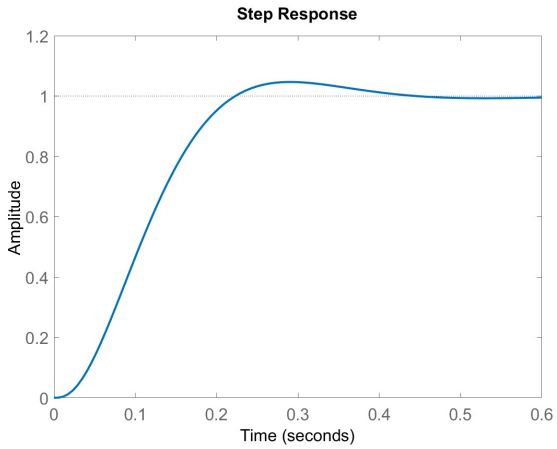


Figure 1.7: Step response with PD-controller

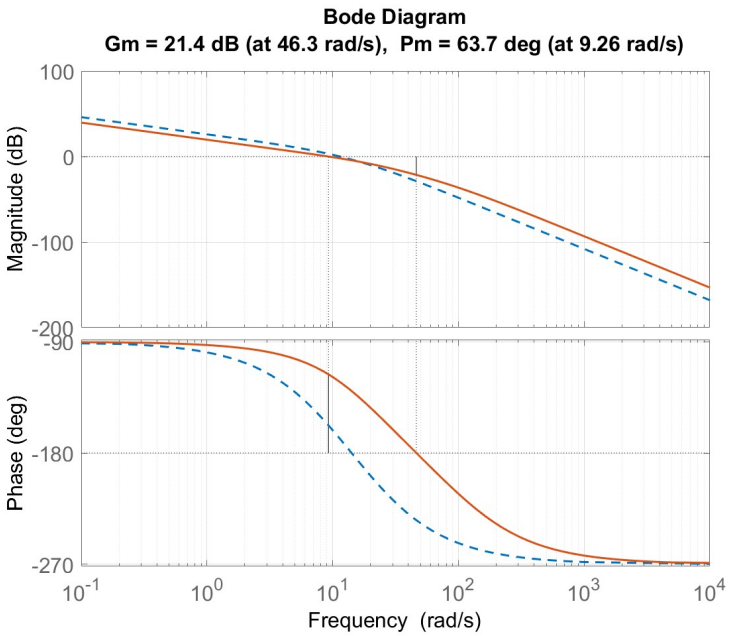


Figure 1.8: Bode diagram of compensated loop transfer

2 Disturbance Rejection

For this task, the block diagram can be redrawn as shown in Figure 2.1. Meanwhile, the relationship between the input q and the output y can be written as

$$y = \frac{G(s)}{1 + C(s)G(s)} q \tag{2.1}$$

Since the disturbance is a unit step signal, the steady-state value of y is:

$$\begin{aligned} y_{ss} &= \lim_{s \to 0} s \cdot \frac{G(s)}{1 + C(s)G(s)} \cdot \frac{1}{s} \\ &= \lim_{s \to 0} \frac{G(s)}{1 + C(s)G(s)} \\ &= \lim_{s \to 0} \frac{1}{\frac{1}{G(s)} + C(s)} \end{aligned} \tag{2.2}$$

Suppose we use P-controller, we already know that there is an integrator in the plant $G(s)$; thus, the term $\frac{1}{G(s)}$ goes to 0 as s approaches to 0. However, the term $C(s)$ is a constant, which causes $y_{ss} \neq 0$; in other words, offset would appear by using P-controller, even though the larger K_P the smaller the offset.

According to the logic above, in order to eliminate the offset, an integrator must be included in the controller. Thus, PI-controller and PID-controller are our candidates. PI-controller would eliminate the offset indeed; nevertheless, it would also reduce the bandwidth and gain margin. PI-controller is bad for the stability of the system, would slow down the response, but good for high-frequency noise rejection. However, in this task there is no high-frequency noise added, so the advantage is not much use. In this case, PID-controller would be the one that could satisfy all the requirements.

PID-controller is in the form of:

$$\begin{aligned} C_{PID}(s) &= K_P + K_I \frac{1}{s} + K_D \frac{s}{Ts + 1} \\ &= \frac{(K_P T + K_D)s^2 + (K_P + K_I T)s + K_I}{s(Ts + 1)} \end{aligned} \tag{2.3}$$

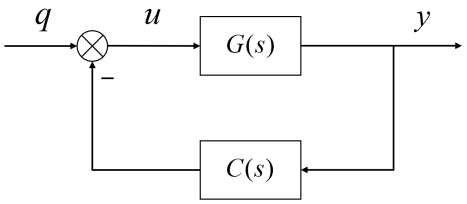


Figure 2.1: Block diagram for disturbance rejection controller design

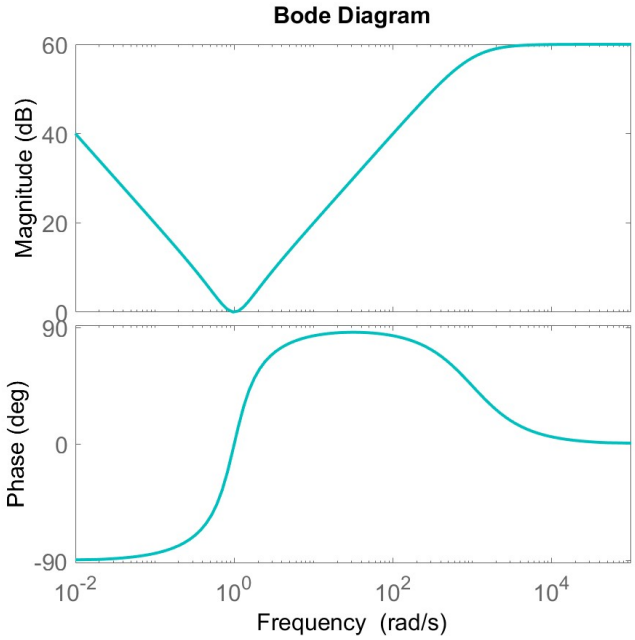


Figure 2.2: Bode diagram of a PID controller

The first-order filter term is added in the derivative part for the purpose of being realizable. It can be observed that if we chose T to be infinite small, then eq.(11) would become the ideal PID controller, which indicates that the smaller T, the harder the controller is to implement. Thus, we choose $T = \frac{1}{1000}$ in order to be realizable and also not hamper its effect much in low and middle frequency regions. Moreover, by using this choice of T terms, $K_P T$ and $K_I T$ can be neglected compared with K_D and K_P, respectively. Thus, for subsequent analytical analysis of the PID controller, we focus on the following simplified form:

$$\hat{C}_{PID}(s) = \frac{K_D s^2 + K_P s + K_I}{s(Ts+1)} \tag{2.4}$$

First, we set $K_P = 1$, $K_I = 1$, and $K_D = 1$ to see what would this compensator bring us, the bode diagram of which is shown in Figure 2.2. The magnitude graph of this controller starts with -20 dB/dec slope. Then, it makes a turn at the frequency determined by the zeros of eq.(2.4): if the zeros appear to be real value, the turn would be obtuse; meanwhile if the zeros appear to be a complex conjugate pair, the turn would be sharp, and the larger the imaginary part compared to its real part, the sharper the turn would be. Finally, the slope becomes 0 dB/dec around 1000 rad/s, which is directly determined by our choice of T. For phase graph, after the frequency corresponding to the turn in magnitude graph, a phase lead is brought to the loop transfer.

Unlike the previous tracking task, for disturbance rejection task, the smaller margins, the better rejection performance. Thus, we would like to use this controller to shape the loop transfer bode diagram to suffice the following specifications: positive gain and phase margin, but be small, and wider bandwidth.

First, we would like to choose the relative relationship between K_P, K_I, and K_D by choosing what kind of zeros we want to add to the system. if we added two real zeros into the system, signals at low frequency region would be less amplified, which would slow down the transient response consequently. Thus, we add a conjugate zero pair to the system. We choose the damping ratio of 0.707 for these two zeros to balance stability and rapidity. Besides, since these two open-loop zeros would produce two closed-loop poles which are close to the imaginary axis, we want the norm of these two open-loop zeros to be as large as possible to achieve a short duration of the disturbance. On the other hand, since we want to use this controller to offer some leading phase at the original phase-drop point, which is around 10 rad/s, the norm of the two open-loop zeros can not be unrestricted large; otherwise, the phase lead would be placed at the wrong frequency region and damage the stability.

In this way, we place the zeros at $-5 + 5i$ and $-5 - 5i$. By choosing this placement, the turning frequency in the magnitude graph of Figure 2.2 is placed at around 7 rad/s, which is able to provide phase lead compensation for the original phase-drop at around 10 rad/s. After calculation of corresponding polynomial terms, we have $K_P = 2K$, $K_I = 10K$, and $K_D = 0.2K$, where K is the controller gain to be determined right next. Assume we take $K = 1$, the loop transfer bode diagram is shown in Figure 2.3. It is visible that now we have quite big phase and gain margin. Thus, by increasing K, which also means decreasing the margins, we could improve the performance of disturbance rejection. In Figure 2.4, the bode diagram of the loop transfer with $K = 10$ is shown, in which the margins shrink. The improvement of disturbance rejection performance is shown in Figure 2.5.

To sum up, the controller for the disturbance rejection task is designed in the following form:

$$C_{PID}(s) = K_P + K_I \frac{1}{s} + K_D \frac{s}{Ts + 1} \tag{2.5}$$

where $K_P = 20$, $K_I = 100$, $K_D = 2$, and $T = \frac{1}{1000}$. The step response for this controller is shown in Figure 2.6.

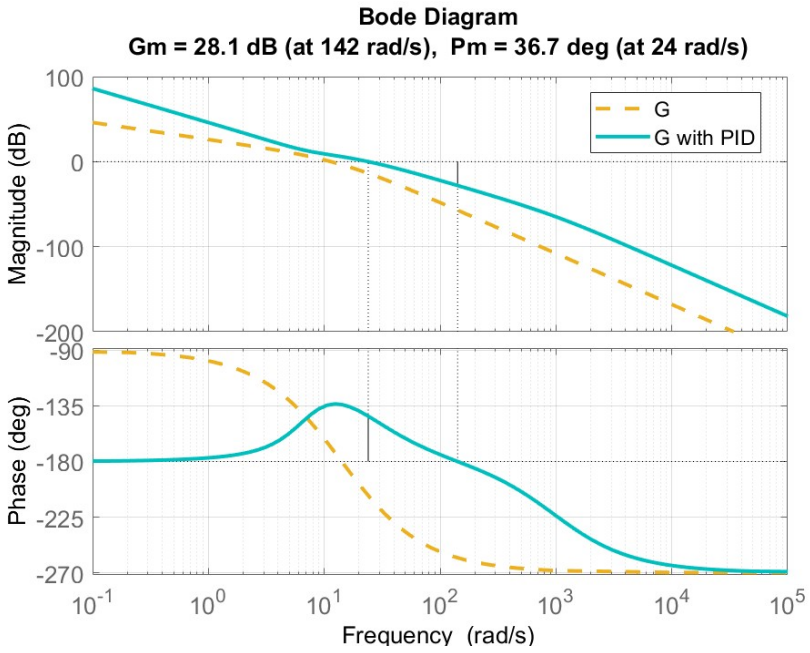


Figure 2.3: Bode diagram of controlled open-loop transfer function with $K = 1$

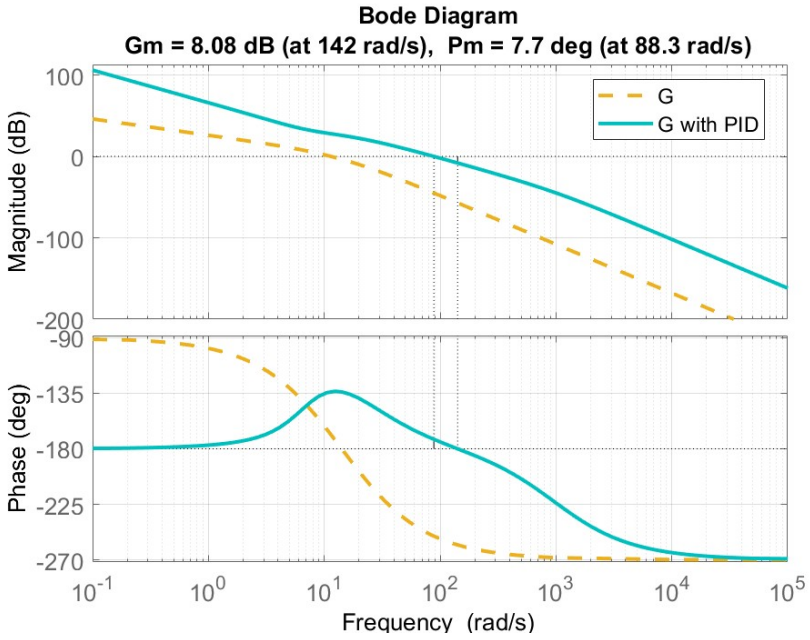


Figure 2.4: Bode diagram of controlled open-loop transfer function with $K = 10$

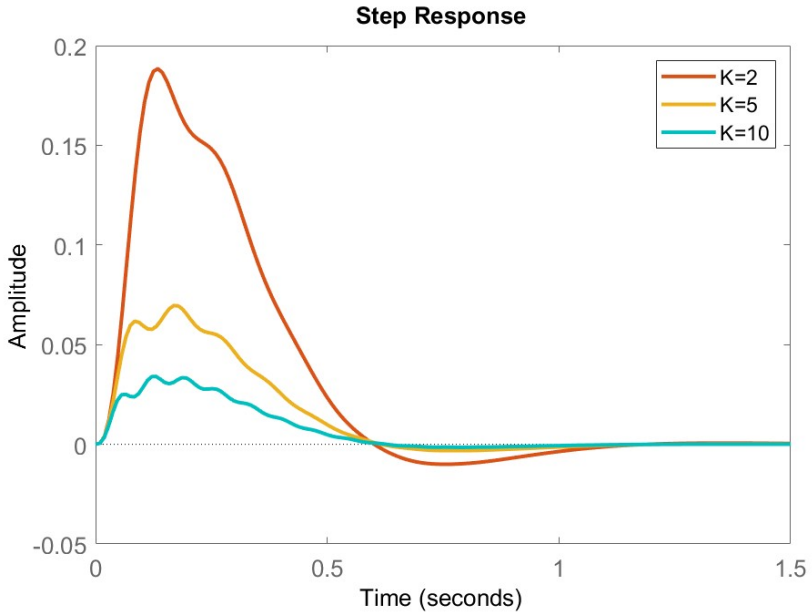


Figure 2.5: Step response of different K

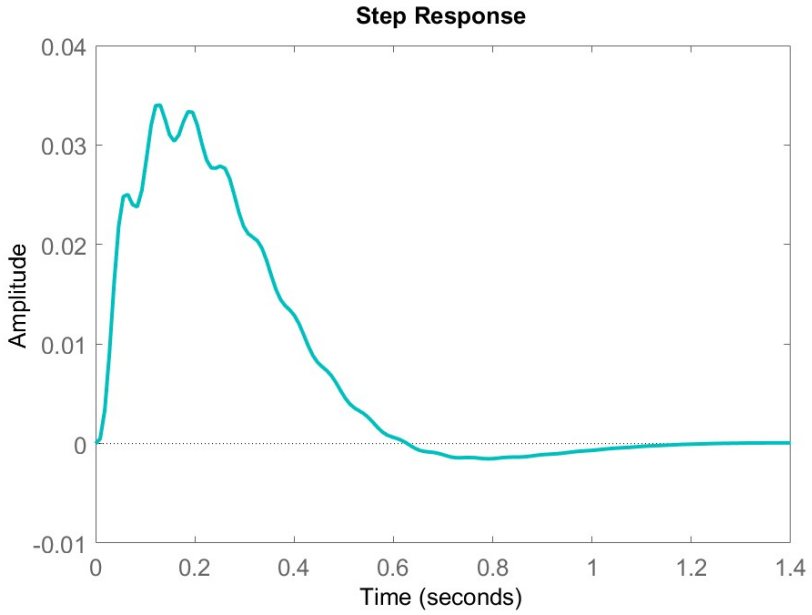


Figure 2.6: Step response with the proposed controller

Discrete-time Control

3 State Space Model

3.1 Continuous state space model

Recall the transfer function:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{4000}{s(s+10)(s+20)} \tag{3.1}$$

where $Y(s)$ is the output, the heading angle of the aircraft, in Laplace domain, and $U(s)$ is the input in Laplace domain. Besides, we denote $y(t)$ and $u(t)$ as the output and the input respectively in time domain for follow-up use. By multiplying both denominators to the other sides, and expanding the simple expressions we have:

$$s^3Y(s) + 30s^2Y(s) + 200sY(s) = 4000U(s) \tag{3.2}$$

transform the above expression into time domain:

$$\dddot{y}(t) + 30\ddot{y}(t) + 200\dot{y}(t) = 4000u(t) \tag{3.3}$$

Then, by defining $x_1 = y$, $x_2 = \dot{y}$, and $x_3 = \ddot{y}$, we realize the transfer function into the following state space model.

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -200 & -30 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 4000 \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \end{bmatrix} u \end{aligned} \tag{3.4}$$

The physical meaning of x_1 is the heading angle of the aircraft, x_2 is the heading angular velocity of the aircraft, and x_3 is the heading angular acceleration of the aircraft.

3.2 Discrete state space model

First, we need to select an appropriate sampling rate. According to Section 2.6 in textbook[1], reasonable sampling rates are ten to thirty times the bandwidth. From Figure 1.5 we know that, the bandwidth is at somewhere slightly larger than open-loop crossover frequency 11.4 rad/s. Thus, we choose sampling rate of 50 Hz; in other words, sampling time is set to be $h = 0.02s$.

Next, we calculate matrix Φ and Γ. We choose Zero-Order-Holder method to discretize the system. In this way, two matrices can be calculated by the following formula:

$$\Phi = e^{Ah} \quad , \quad \Gamma = \int_0^h e^{As}\, dsB \tag{3.5}$$

[1] Åström Karl J; Wittenmark Björn. Computer Controlled Systems : Theory and Design; Prentice-Hall Information and System Sciences Series; Prentice-Hall: Englewood Cliffs, NJ, 1984.

To calculate exponent of a matrix, we first calculate the Jordan form and the corresponding transfer matrix T. By doing so, we get:

$$A = SJS^{-1} \tag{3.6}$$

where $A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -200 & -30 \end{bmatrix}$, $J = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -20 & 0 \\ 0 & 0 & -10 \end{bmatrix}$, $S = \begin{bmatrix} 1 & 0.0025 & 0.01 \\ 0 & -0.05 & -0.1 \\ 0 & 1 & 1 \end{bmatrix}$.

Then,

$$\Phi = e^{Ah} = S \begin{bmatrix} 1 & 0 & 0 \\ 0 & e^{-20 \cdot 0.02} & 0 \\ 0 & 0 & e^{-10 \cdot 0.02} \end{bmatrix} S^{-1} = \begin{bmatrix} 1 & 0.0198 & 0.0002 \\ 0 & 0.9671 & 0.0148 \\ 0 & -2.9682 & 0.5219 \end{bmatrix}$$

$$\Gamma = S \begin{bmatrix} 0.02 & 0 & 0 \\ 0 & -\frac{1}{20}(e^{-20 \cdot 0.02} - 1) & 0 \\ 0 & 0 & -\frac{1}{10}(e^{-10 \cdot 0.02} - 1) \end{bmatrix} S^{-1}B = \begin{bmatrix} 0.0046 \\ 0.6572 \\ 59.3643 \end{bmatrix}$$

By using MATLAB, the above result is verified as shown in Figure 3.1.

```
A =
            x1         x2          x3
 x1          1    0.01977   0.0001643
 x2          0     0.9671     0.01484
 x3          0     -2.968      0.5219

B =
           u1
 x1  0.004603
 x2    0.6572
 x3     59.36

C =
     x1  x2  x3
 y1   1   0   0

D =
     u1
 y1   0

Sample time: 0.02 seconds
Discrete-time state-space model.
```

Figure 3.1: Discretized system by using MATLAB

4 Discretization Simulation

4.1 Set-point step tracking

We tried two methods to discretize the controller: ZOH and Tustin. The sampling time for this task is set to be $h = 0.02s$. The tracking result is shown in Figure 4.1. It can be seen that by using ZOH the resulting system shows quite different dynamics compared to the continuous system; however, by using Tustin the resulting system shows a much more resemble dynamics with the continuous system.

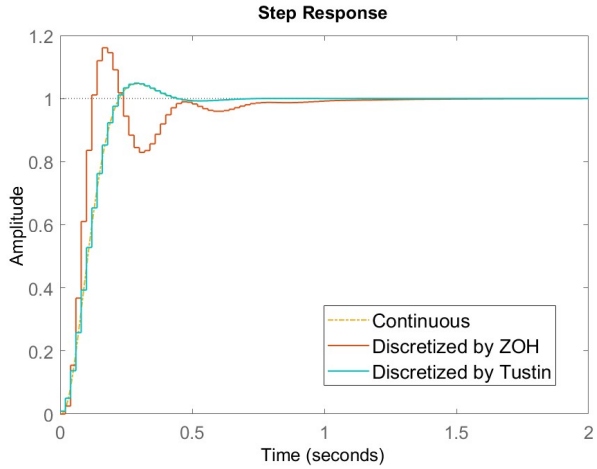


Figure 4.1: Step response with different discretization methods

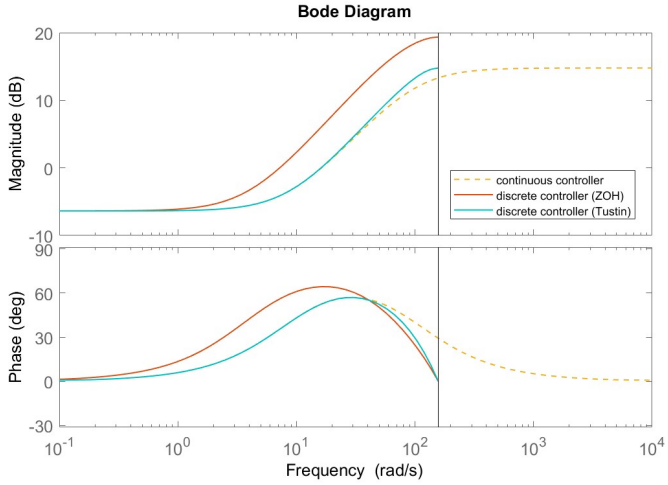


Figure 4.2: Bode diagram of controllers with different discretization methods

This difference can be explain by looking at the bode diagram of different discretized controllers, as shown in Figure 4.2. From the bode diagram in Figure 1.8, we can observe that the phase curve cross over $-180°$ at 46.3 rad/s, which means if we want the discretized system to have similar behavior, we want both discretized magnitude and phase to be similar to the continuous one for all frequency smaller than 46.3 rad/s. In Figure 4.2, distortion brought by Tustin method only happens after around 40 rad/s, which explains the high resemblance by using Tustin. In contrast, distortion brought by ZOH method takes place almost across the entire frequency range, which leads to a different closed-loop dynamics. It is worth mentioning that if we shortened the sampling time, the performance by using ZOH would be improved; however, due to the nature of this method, the improved behavior would still be quite different from that of the continuous system.

4.2 Step disturbance rejection

We first try to use the same sampling time $h = 0.02s$. However, ZOH method failed by using this sample rate: the step response to the whole discretized system diverges, as shown in Figure 4.3. The reason for this situation is that the sampling rate is too small while the phase crossover frequency of the continuous system is at 142 rad/s according to Figure 2.4. Thus, in order to let ZOH method work, we need to increase the sampling rate.

Unlike ZOH method, Tustin would only bring distortion for frequency that are close to the Nyquist frequency $\frac{\pi}{h}$. Thus, Tustin method gives a decent performance when using $h = 0.02s$. In Figure 4.4, the step disturbance rejection of two different discretization methods are shown. It is visible that, even with $h = 0.001s$ for ZOH the dynamics change due to discretization is still significant; meanwhile, Tustin doesn't require the sampling rate to be that high while still showing a better performance.

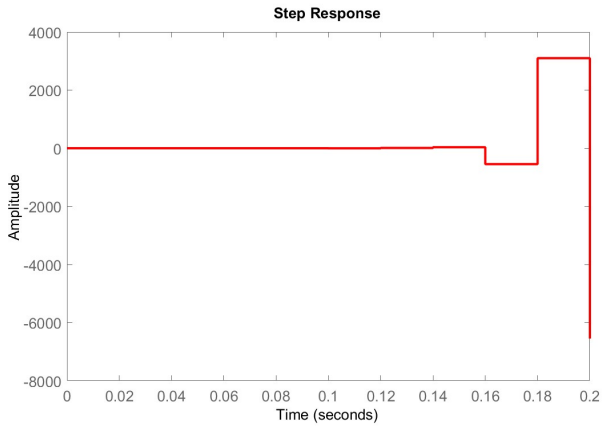


Figure 4.3: Divergence happens when ZOH using a big sampling time interval

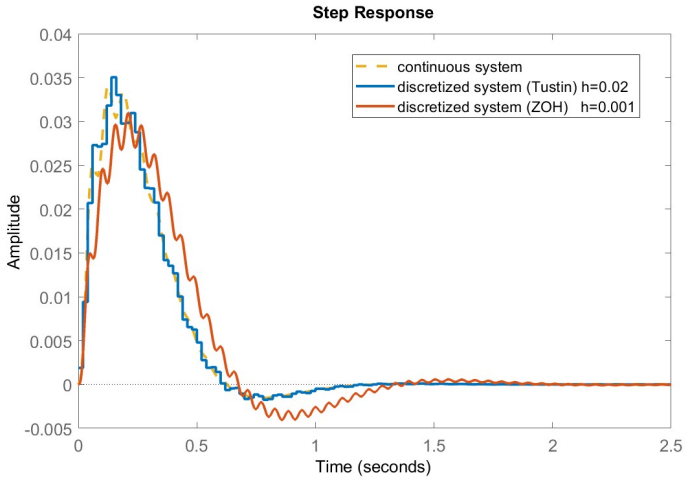


Figure 4.4: Step response with different discretization methods

5 Servo Controller by Using Pole-placement

According to Åström Karl J's book, the primary concern in pole-placement design is to make sure that errors in the initial state of a system will decay in a specified way. For regularization problem, we want to drive the state to the zero. Thus, the following control law is proposed, assume we could measure the state directly:

$$u(k) = -Lx(t) \tag{5.1}$$

However, for servo problem, we want the system to track a reference. Therefore, the following control law will be applied in this section.

$$u(t) = L(r(t) - x(t)) = -Lx(t) + Lr(t) \tag{5.2}$$

where $r(t)$ is the user-defined reference, and of the same dimension as $x(t)$.

Before designing the feedback vector L, we would like to check the reachability of the system; in other words, we would like to check whether matrix $\begin{bmatrix}\Gamma & \Phi\Gamma & \Phi^2\Gamma\end{bmatrix}$ has rank of 3. The answer is yes, the rank is of 3, which means we are able to place the poles wherever we want.

After plugging the control law in the state space model, we have the following form:

$$\dot{x} = (A - BL)x + BLr \tag{5.3}$$

where in our case $r = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$, since we want $x_1 = y$ to follow the step reference. It is worth mentioning that we use $r = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$ as the reference for states is equivalent to using reference $r_y = 1$ for output y mathematically, because $\dot{r}_y = \ddot{r}_y = 0$ for $k > 0$, which are the corresponding references for state

x_2 and x_3 respectively. Thus, no additional information is introduced to the system by doing so.

We chose four sets of different poles locations, as listed in Table 1. Meanwhile, the corresponding step response is shown in Figure 5.1. It can be seen that the closer the poles to the origin, the faster response the system will have. Since we can place the poles at arbitrary locations, we could further choose poles that are closer to the origin to get better performance compared to all the chosen poles locations in Table 1.

P_i	Pole 1	Pole 2	Pole 3
1	0.9	0.8+0.5i	0.8-0.5i
2	0.7	0.6+0.5i	0.6-0.5i
3	0.5	0.4+0.5i	0.4-0.5i
4	0.5	0.4+0.3i	0.4-0.3i

Table 1: Choices of different poles locations

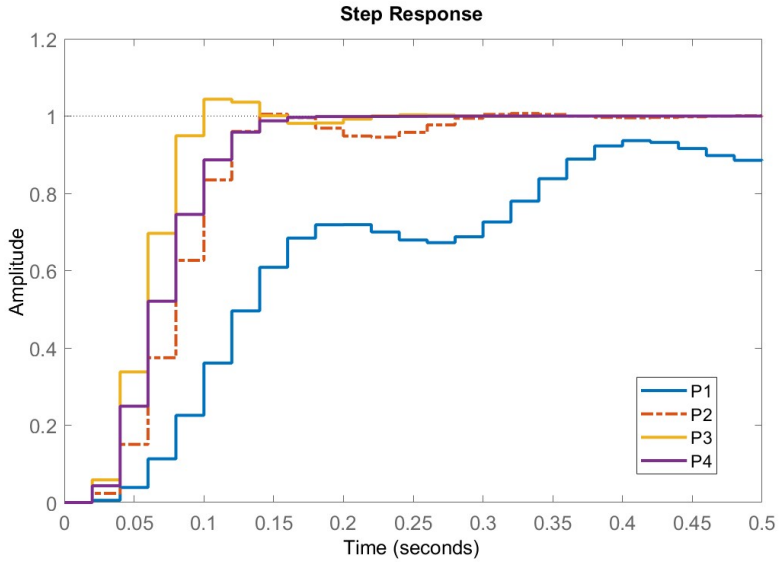


Figure 5.1: Step response by choosing different poles locations

However, there is no free lunch. When we get a better performance, we must sacrifice something or make bigger effort somewhere else. In our case, the better performance indicates the higher requirement for the actuator, which is crucial in practice. Therefore, we shall illustrate this point briefly here. We observe the controller output for poles locations set P_2 and P_3, as shown in Figure 5.2.

From Figure 5.1, we know that P_2 gives weaker performance compared to P_3. On the other hand, the maximum controller output required by using P_2 is only 5, smaller than that by using P_3, larger than 10. The trade-off between performance and requirement to actuator is manifested here.

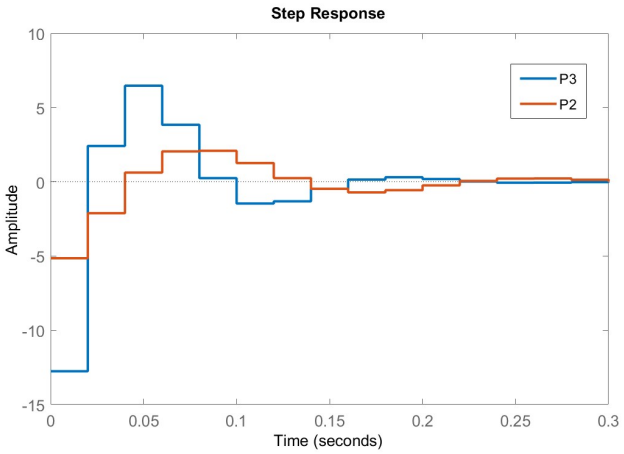


Figure 5.2: Controller output comparison

Finally, the pole location for the final design is $[0,\ 0.0001+0.0001i,\ 0.0001-0.0001i]$, of which the step response is shown in Figure 5.3. This is the best it can do with sampling interval $h = 0.02s$. If we further shorten the sampling time, the settling time would be shorter, while it would still require the same number of steps.

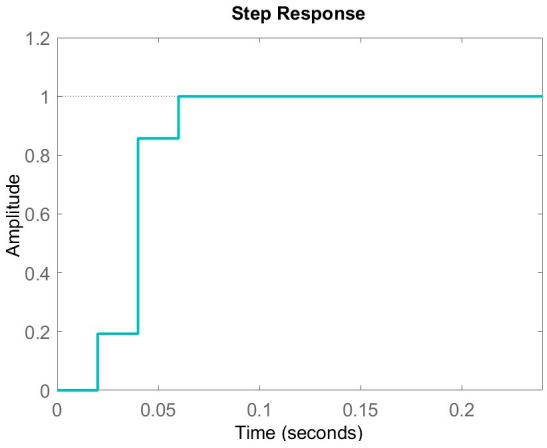


Figure 5.3: Step response of the final design

6 Output-feedback Control

6.1 Servo tracking scheme

The system is described by:

$$\begin{aligned} x((k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= Cx(k) \end{aligned} \tag{6.1}$$

Unlike problem 5, in this section we need to estimate states by only using information of input u and output y. Thus, a state observer will be introduced:

$$\hat{x}(k+1) = \Phi\hat{x}(k) + \Gamma u(k) + K[y(k) - C\hat{x}(k)] \tag{6.2}$$

where the hat notation stands for estimation, and vector K is to be designed. By defining error signal $e = x - \hat{x}$, and combining eq.(6.1) and (6.2), we obtain the error dynamics:

$$e(k+1) = [\Phi - KC]e(k) \tag{6.3}$$

By choosing different K, poles of matrix $\Phi - KC$ can be placed at certain locations. Under the premise that $\Phi - KC$ is set to be Hurwitz, error signal would converge to zero, which means estimation $\hat{x}$ would reflect the real value of x.

First, we would like to investigate the observabliliy of this system. The rank of observablility matrix $W_o = \begin{bmatrix} C^T & \Phi^T C^T & \Phi^T\Phi^T C^T \end{bmatrix}^T$ is 3, which means the system is observable. Say the poles we chose for closed-loop dynamics is $[0.4,\ 0.5 + 0.35i,\ 0.5 - 0.35i]$, we set the poles of the observer at $[0.2,\ 0.3 + 0.2i,\ 0.3 - 0.2i]$. The reason is that we want the error dynamics to have a faster response than state dynamics, in order not to affect state dynamics with big and slowly converging estimation error.

The tracking performance is shown in Figure 6.1. The initial condition is set to be 1.2 in order to investigate the estimation error. It can be seen that at the transient phase there is huge difference between using observer or without. That is because the observer had not kept up with the real value of states yet. The estimation error for three states are shown in Figure 6.2, in which we can see that all the errors converge to zero eventually; in other words, the observer estimates the states perfectly after 0.1s.

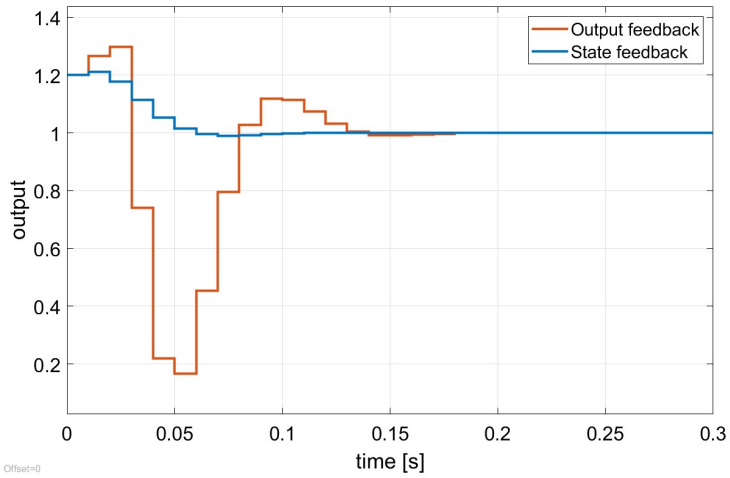


Figure 6.1: Step tracking comparison by using different schemes

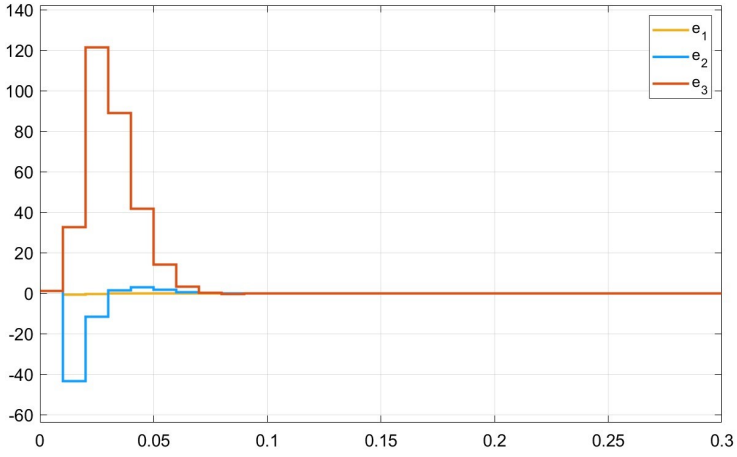


Figure 6.2: Estimation error of three states

6.2 Output feedback with integral action

We first place a input step disturbance on the system used in Section 6.1. The disturbance of magnitude 1 is triggered at $t = 1.5s$, and the resulting tracking behavior is shown in Figure 6.3 by the red line.

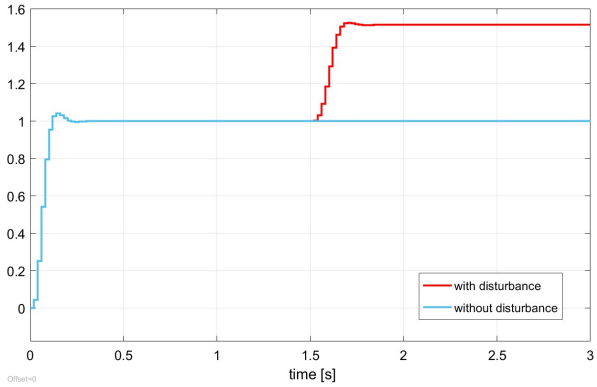


Figure 6.3: Tracking error appears when disturbance is added

It is visible that our control strategy in Section 6.1 does not own the ability to reject step disturbance at the plant input side. Inspired by classical control theory, we would like to add integral action to the controller. One way to introduce a integrator to the system is to add a new state x_{n+1} to integrate the output error $r_y - y$, i.e.,

$$x_{n+1}(k+1) = x_{n+1}(k) + r_y - y \tag{6.4}$$

where r_y in our case equals to 1. Then, the system can be augmented as:

$$\begin{bmatrix} x(k+1) \\ x_{n+1}(k+1) \end{bmatrix} = \begin{bmatrix} \Phi & 0 \\ -C & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ x_{n+1}(k) \end{bmatrix} + \begin{bmatrix} \Gamma \\ 0 \end{bmatrix} u(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r_y(k) \tag{6.5}$$

Next, pole placement can be implemented to the new augmented system. For our servo task, the control law is then extended to:

$$u(k) = L[r(k) - \hat{x}(k)] + l_{n+1}x_{n+1}(k) \tag{6.6}$$

where in Z domain $x_{n+1}(k) = \frac{1}{z-1}(r_y - y)$, which can be derived from eq.(6.4).

Finally, the poles we chose for the augmented system is shown in Table 2. Moreover, the step disturbance rejection performance for the final design is shown in Figure 6.5 by the green line. It can be seen that the input step disturbance is rejected after a period of time, while the tracking ability is slightly weakened. Furthermore, we change the location of pole 4 to show the trade-off relationship between tracking ability and disturbance rejection ability. When we put pole 4 at 0.4, which means we want the disturbance rejection to have a faster response, we get a bigger overshoot for tracking; conversely, if we put pole 4 at 0.92, we get a worse disturbance rejection result, while we have a smaller overshoot for step tracking.

Pole 1	Pole 2	Pole 3	Pole 4
0.01	0.01+0.03i	0.01-0.03i	0.85

Table 2: Final choice for the augmented system

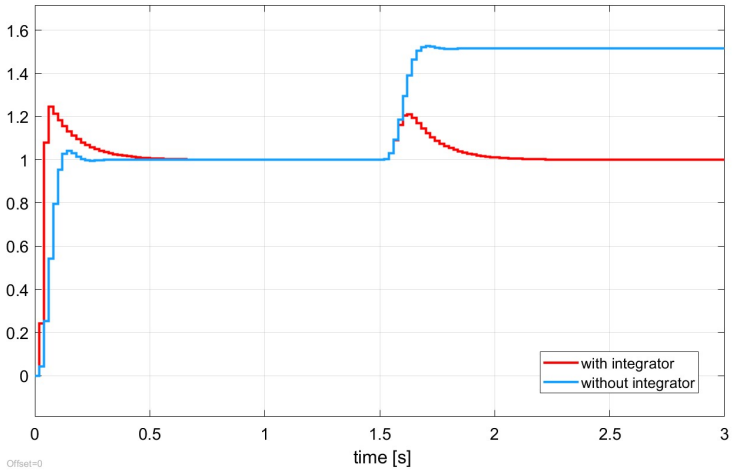


Figure 6.4: Step disturbance rejection for system with and without an integrator

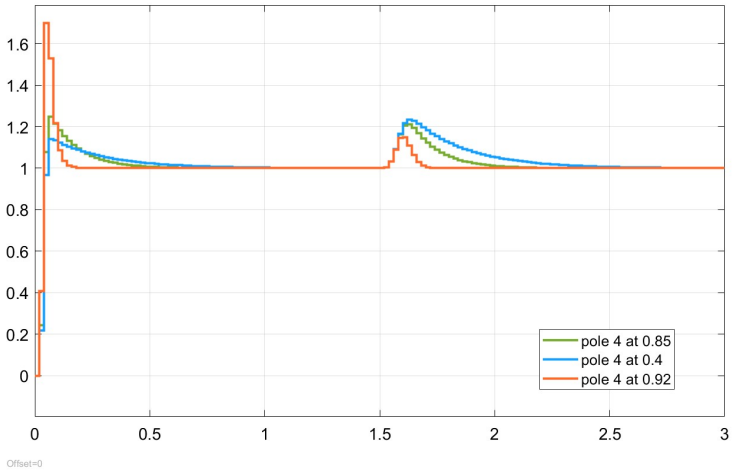


Figure 6.5: Performance comparison between different pole locations

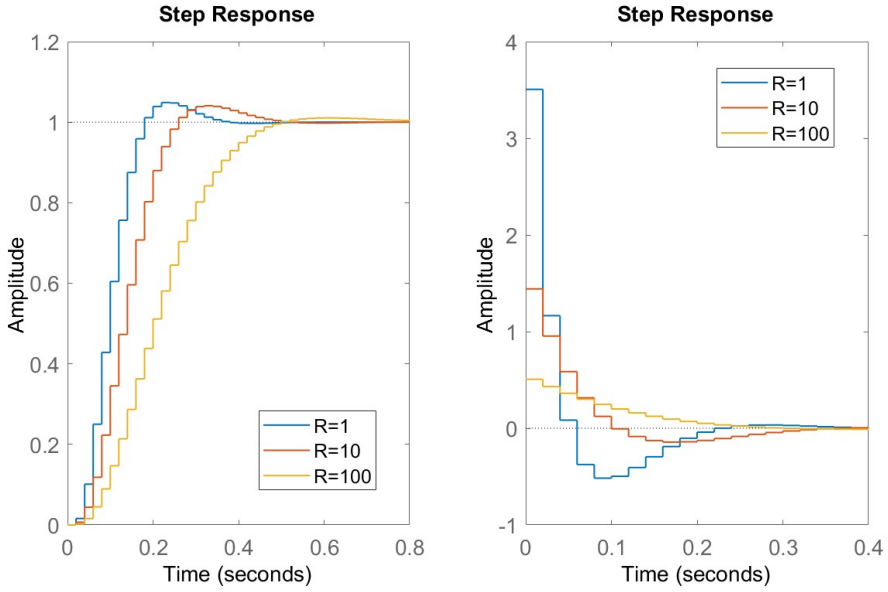
7 LQ Control

Since the system has three states and one input, weighting matrix Q should be a 3×3 matrix, and R should be a scalar. For simplicity, we chose Q to be in diagonal form:

$$Q = \begin{bmatrix} q_1 & 0 & 0 \\ 0 & q_2 & 0 \\ 0 & 0 & q_3 \end{bmatrix}$$

For our case and chosen matrix form, we have four weighting factors: q_1, q_2, q_3 and R. They are responsible for penalizing x_1, x_2, x_3 and u respectively.

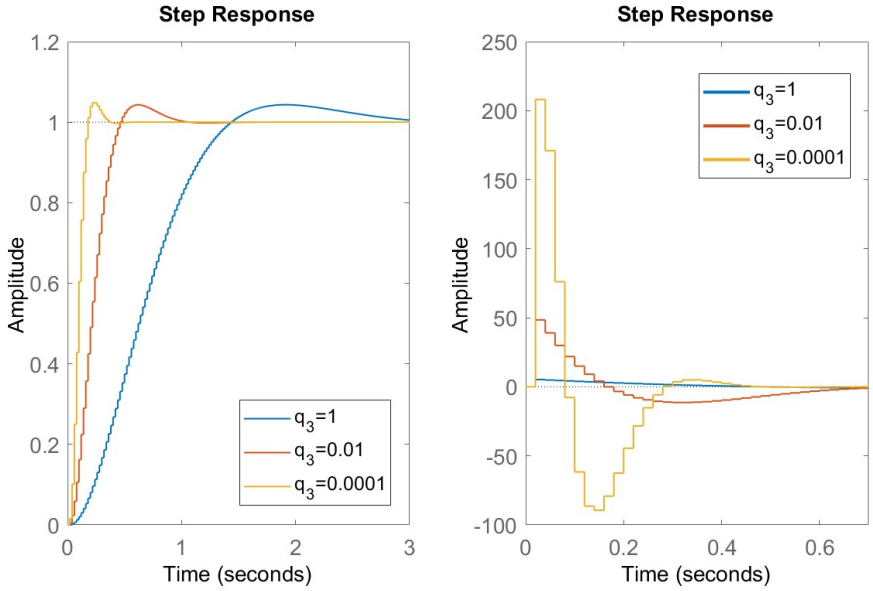
We first investigate the effect of R. For a larger R, smaller maximal value of controller output u is restricted, while if we set a smaller R, the restriction for u would be slacked. That is to say, if we set R big, the controller would have a more limited effect in each time interval to the system, which would bring the closed-loop system a slower response. This effect of R is illustrated by Figure 7.1a, in which three different values of R are chosen: 1, 10 and 100. Besides, the controller output for corresponding R are shown in Figure 7.1b. By comparison we can find out that the cost of fast response is a bigger allowed maximal controller output value, that is, a higher requirement for the actuator.



(a) Output y with different R (b) Controller output u with different R

Figure 7.1: Effect of control weight R

Next, we focus on the effect of state weight matrix Q. For bigger q_i, the behavior of the corresponding state is more limited. We choose q_3 to illustrate this point. The physical meaning of x_3 is the angular acceleration of the heading angle. If we set a large penalty to x_3, say $q_3 = 1$, the resulting x_3 would be limited to a small value, as shown in Figure 7.2b by the blue line. That is to say, the acceleration allowed for each step is limited, which cause the tracking response slow, as shown in Figure 7.2a by the blue line. In contrast, if we set the penalty $q_3 = 0.00001$, which means we do not apply much restriction on the behavior of x_3, the resulting x_3 would be changing in a larger range compared to $q_3 = 1$, as shown in Figure 7.2b by the yellow line. Intuitively, the system has a larger angular acceleration at each step, which is also manifested by the fast tracking response as shown in Figure 7.2a by the yellow line.



(a) Output y with different q_3 (b) State x_3 with different q_3

Figure 7.2: Effect of control weight q_3

Last but not least, the effect of two weighting matrices Q and R are relative. If we get a better state behavior, we will get a larger controller output as the cost; In contrast, if we want a small control output, we can not expect a better state behavior at the same time. The adjustment of Q is also adjusting R in a general perspective. The trade-off relationship is the conclusion by choosing different weighting matrices.

In Figure 7.3, the step tracking for the final design using LQ control is

shown, where $R = 0.01$, and Q is set to be:

$$Q = \begin{bmatrix} 1000 & 0 & 0 \\ 0 & 0.0001 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix}$$

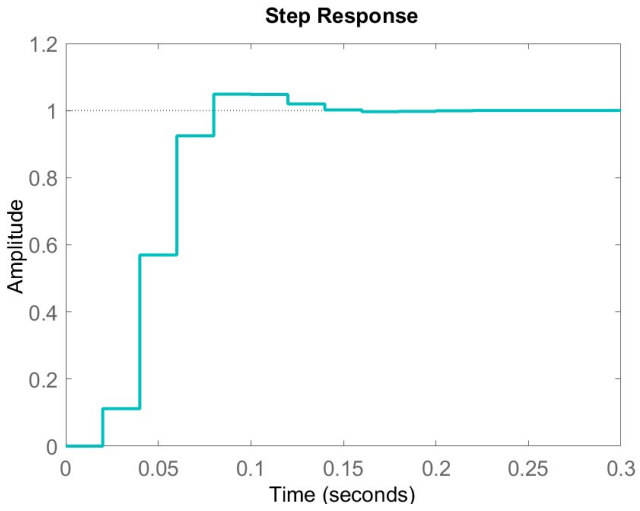
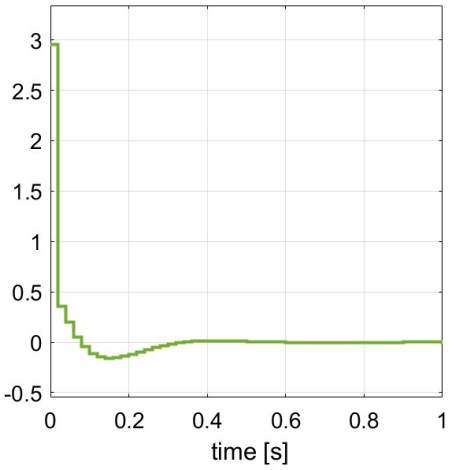


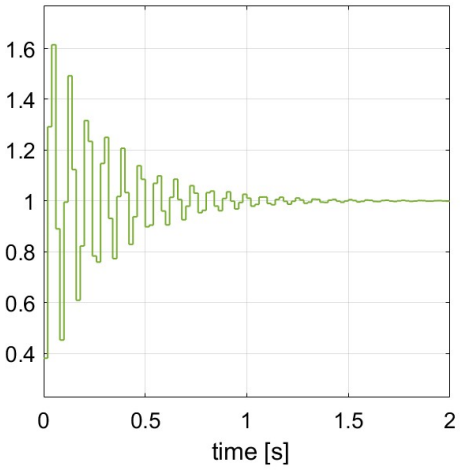
Figure 7.3: Final design performance for LQ control

8 Controller outputs

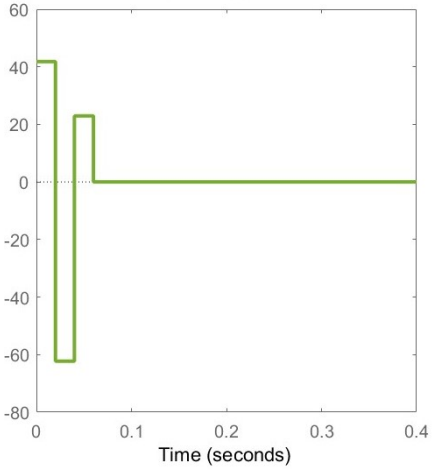
In this section, the outputs of discrete controllers designed in the previous sections are shown. It can be observed that most of the controller outputs require actions of magnitude larger than range ± 1, which is not quite doable for practice. Thus, redesign and modifications are to be done in the following sections.



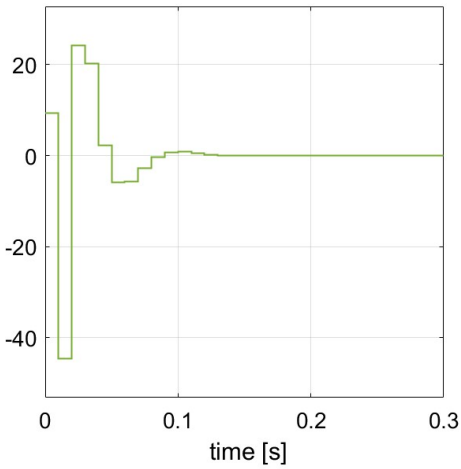
(a) Corresp. to the lake-blue-line in Fig 4.1



(b) Corresp. to the blue-line in Fig 4.4

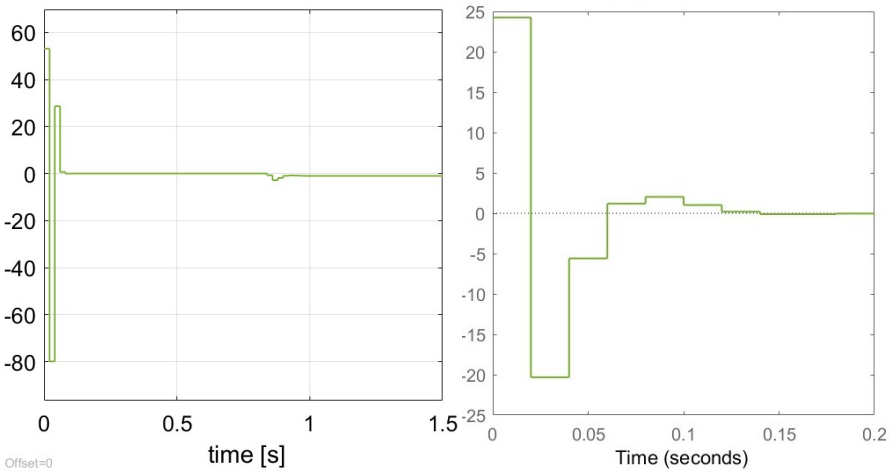


(c) Corresp. to Fig 5.3



(d) Corresp. to the orange-line in Fig 6.1

Figure 8.1: Controller outputs part 1



(a) Corresp. to the green-line in Fig 6.5 (b) Corresp. to Fig 7.3

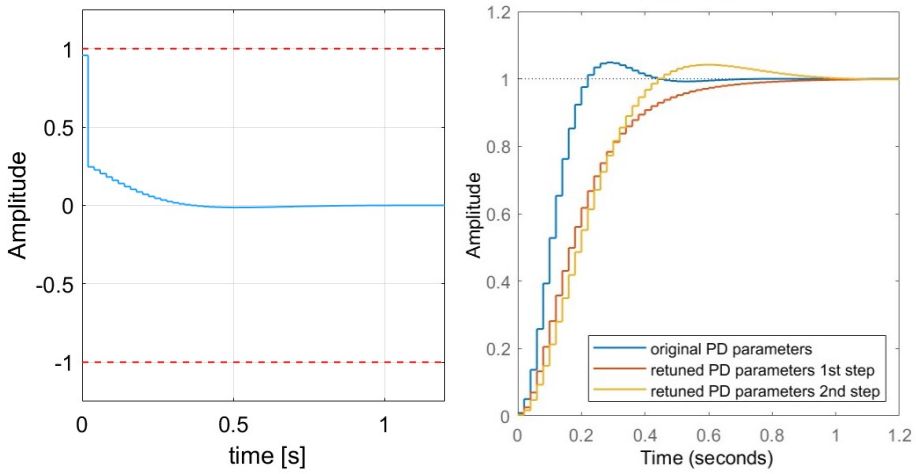
Figure 8.2: Controller outputs part 2

9 Discrete PID-type Controller Redesign

9.1 Tracking problem

The original parameter for the PD-controller is $K_P = 0.48$, $K_D = 0.05$, and $T = \frac{1}{100}$. Since the controller output in Figure 8.1a has magnitude of almost 3, we would like to scale down K_P and K_D altogether to its 50% first, that is $K_P = 0.24$, $K_D = 0.025$. By doing so, we obtain the red line in Figure 9.1b. However, since this setting would still require controller to have a output slightly larger than 1, further tuning is necessary. That overshoot disappears enlightens us to decrease K_D, since derivative term tends to let system be more stable which produces less overshoot. Therefore, by reducing K_D, controller output would enter the bound of ± 1 while the overshoot still being acceptable.

As a result, the final re-tuned parameters for tracking is $K_P = 0.26$, $K_D = 0.014$, and $T = \frac{1}{100}$. The tracking performance is shown in Figure 9.1b by the yellow line, and the controller output is shown in Figure 9.1a.



(a) Controller output with new parameters (b) Outputs y with different parameters

Figure 9.1: Redesign performance

9.2 Disturbance rejection

There are two ways to interpret the limitation range ± 1. One interpretation is that the range is dynamic, that is to say an offset is allowed, and the maximum of the controller output minus the minimum of the controller output is smaller than $1 - (-1) = 2$. According to this interpretation, by observing Figure 8.1b, we can see that the controller output has already lain in the range ± 1 with an offset of 1. Thus, no redesign is needed for this PID-controller.

The other interpretation of the limitation is that the controller output

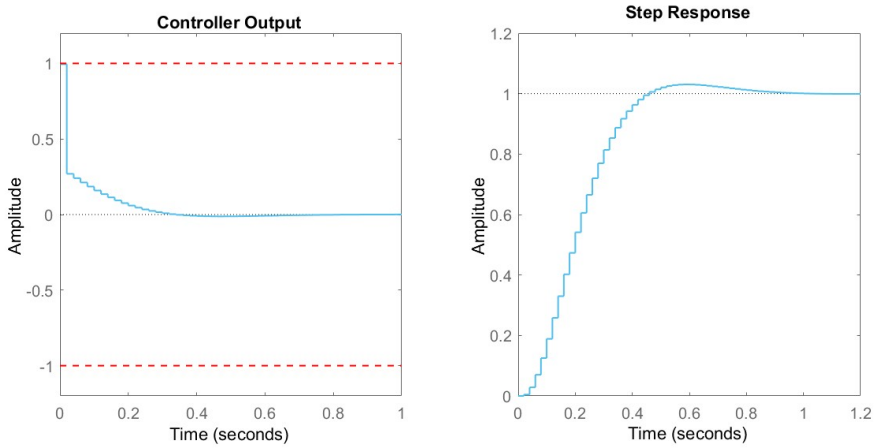
should lie in range $[-1, 1]$. However, for such interpretation, it is impossible to both keep controller output inside the static range $[-1, 1]$ and achieve zero-steady-state disturbance rejection. This can be explain in this way: for a zero-steady-state disturbance rejection, the controller output should converge to 1 to cancel out the unit step disturbance at the plant input. However, due to the limitation $[-1, 1]$, the allowed full power for the controller is 1. This would cause the error signal $e = d - y_c \leq 0$, where y_c is the output of the controller and the error signal e is fed into the plant. Besides, no matter what controller we use, inevitable deviation will always take place at the beginning, because the controller would not immediately reach 1 to counteract the effect from disturbance. That is to say, in order to achieve zero steady state error, the effect of a negative error signal must be mended by a later on positive error signal which could only be produced by controller output that is larger than 1. Therefore, we can conclude that to achieve zero-steady-state-error disturbance rejection, oscillation around the magnitude of the step disturbance should be expected in the controller output. For our case, the step disturbance has maginitude 1, thus, oscillation around 1 will happen for zero-steady-state-error disturbance rejection in the controller output behavior. Thus, if we must limit controller output to be smaller or equal to 1, zero-steady-state-error unit step disturbance should not be expected.

10 Discrete Pole-placement Redesign

The performance of the final design of discrete pole placement scheme regardless of controller output limitation is shown in Figure 5.3. Meanwhile, the corresponding controller output is shown in Figure 8.1c. Obviously, the value is out of range ± 1. The original chosen poles are $[0,\ 0.0001 + 0.0001i,\ 0.0001 - 0.0001i]$. In order to restrict the controller output, we would like to place the poles further from origin. After placing two conjugate poles at $0.9 \pm 0.9i$, the controller output finally fell into the range ± 1. However, the overshoot became unacceptable. Thus, we then reduced the imaginary part to reduce the overshoot.

Finally, by tuning a couple of times, we obtain the optimal poles $[0,\ 0.88 + 0.0968i,\ 0.88 - 0.0968i]$, of which the performance is shown in Figure 10.1. It is visible that controller output is within the range ± 1 and meanwhile the tracking satisfies all the specifications.

One thing we could learn from this redesigning process is that designing a controller is essentially making compromise among different specifications. In order to avoid controller output saturation, we must sacrifice some of the system's rapidity in order to still satisfy the original specifications.



(a) Controller output with new parameters (b) Outputs y with different parameters

Figure 10.1: Redesign performance

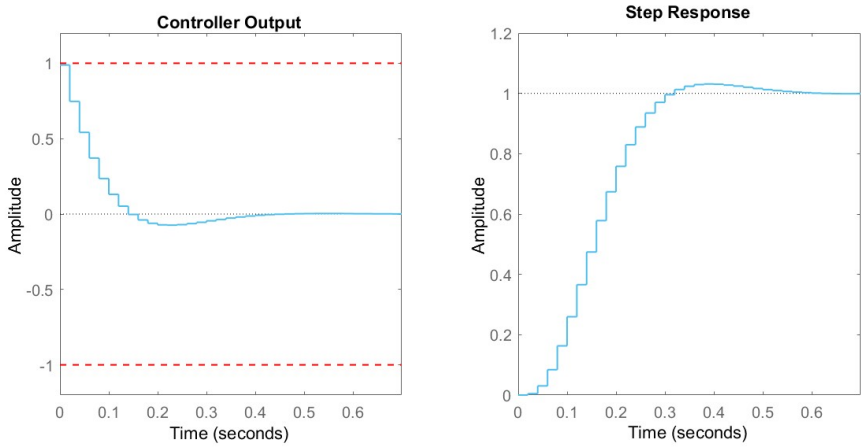
11 Discrete LQ-controller Redesign

The performance of the final design of discrete pole placement scheme regardless of controller output limitation is shown in Figure 7.3. Meanwhile, the corresponding controller output is shown in Figure 8.2b. Obviously, the value is out of range ± 1. In order to limit the controller output, we could directly increase the weight R to penalize controller output u. After putting u inside the allowable range, we then adjust diagonal elements of matrix Q to avoid big overshoot.

In the end, the step tracking for the final design using LQ-control with limitiation on u is shown in Figure 11.1, where $R = 4000$, and Q is set to be:

$$Q = \begin{bmatrix} 5000 & 0 & 0 \\ 0 & 0.0001 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix}$$

Similarly, some of the rapidity is sacrificed to guarantee the controller output will not break the limitation. Compared with using pole-placement in Section 10, the tracking by using LQ-control is slightly better. One reason could be found out by checking the poles of the closed-loop system by using LQ-control: there is a non-zero real-value pole near the conjugate pair. Recall that we set a pole at zero using pole placement in the last section, we can conclude that a non-zero real-value would improve the performance, even though it would increase the complexity of analysis.



(a) Controller output with new parameters (b) Outputs y with different parameters

Figure 11.1: Redesign performance

12 Steady-state Error Elimination

Throughout the project, we investigated and designed two main control approaches: one based on transfer function and one based on state-space model. For the former one, we first design a continuous controller and then discretize it, which is done in Section 4. For zero-steady-state tracking, we need to guarantee at least one integrator included in the loop transfer $L = C \cdot G$. The reason behind this conclusion can be found by deriving the error dynamics. For zero-steady-state disturbance rejection, we need to include at least one integrator in the controller C. This is concluded from eq.(2.2) in Section 2.

To show this, we tuned a PID controller with final parameters $K_P = 2$, $K_I = 5$, $K_D = 0,5$, and $T = 1/1000$. We set a unit step reference at $t = 0s$ and a unit step disturbance at $t = 3s$, of which the corresponding output is shown in Figure 12.1. Through the whole process of tuning, we could get the intuition that if we use one controller to track a reference and reject disturbance at the same time, we must make compromise between the performance of them. One of the theoretical supports behind this trade-off phenomenon is that good tracking performance requires big margins, while good disturbance rejection performance requires small margins. Therefore, although indeed adding an integrator would eliminate steady-state error for both step reference and step disturbance, necessary tuning must be done in practice. It is worth mentioning that since an integrator is added, it can be also observed that for tracking more oscillation is produced compared to using P or PD controller.

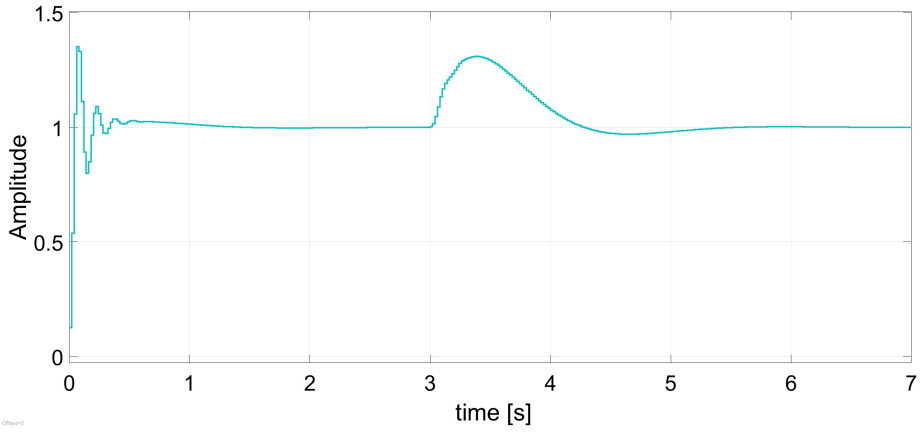


Figure 12.1: Steady-state error elimination performance

For the latter scheme, which is based on state-space model, we have two branches: state-feedback and output-feedback, both of which will be capable of eliminating steady-state error if we add integral action to the system. The analysis of the conclusion can be found in Section 6.2. It is worth mentioning that since LQ-control is implemented by the exact same way as state-feedback pole-placement, adding integral action will also eliminate steady-state error for

both disturbance rejection and tracking by using LQ-control. Finally, for this state-space-model-based category, we take output-feedback with integral action as an example to show the steady-state error elimination ability. We set a step reference at $t = 0s$, and a step disturbance at input of the plant at $t = 1.5s$. The result has been already shown once in Figure 6.5; besides, Figure 6.4 could be used as a comparison, which does not contain an integrator, thus is not capable of elimination steady-state error from the step disturbance.

To sum up, steady-state error elimination methods for each control approaches are summarized in Table 3.

Control Scheme	Steady-state Error Elimination Method	Corresp. Section(s)
Discretizing continuous controller	**Tracking:** at least one integrator included in the loop transfer; **Disturbance rejection:** at least one integrator included in the controller;	4
State-feedback pole-placement	**Tracking:** all poles need to be placed inside the unit circle; **Disturbance rejection:** integral action needs to be added;	5
Output-feedback pole-placement	same as above	6
LQ-control	same as above	7

Table 3: Steady-state error elimination methods

13 Time Step Delay

13.1 Add the delay in Simulink model

In Simulink, the extra time step delay can be implemented in the way shown in Figure 13.1 and Figure 13.2 respectively. The corresponding resulting tracking performance is shown in Figure 13.3. From Figure 13.2a, we can observe that a bigger overshoot is produced while still being stable. On the contrary, in Figure 13.2b, the output diverges. The reason why divergence appears can be easily explained: since we used the old feedback vector L produced by the original system parameter, it is not possible to place the poles of the new system which includes a time delay as we wanted before. Thus, we must augment the system description and then place all poles altogether.

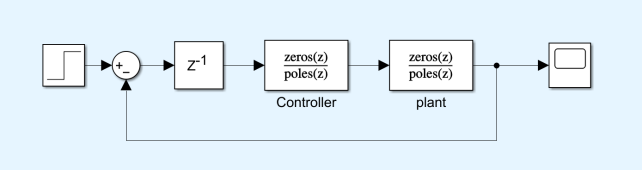


Figure 13.1: Time delay implementation on transfer function in Simulink

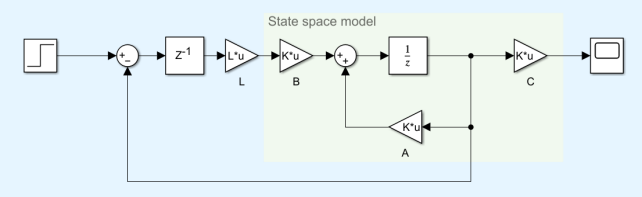
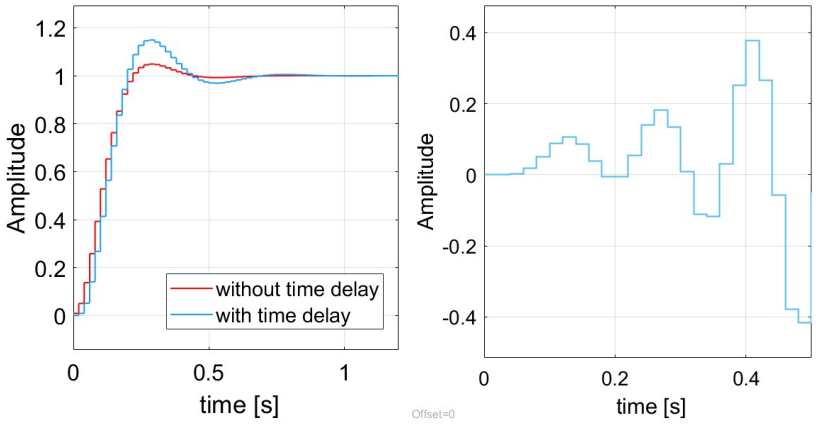


Figure 13.2: Time delay implementation on state-space model in Simulink



(a) Transfer function model result (b) State space model result

Figure 13.3: Adding a time step delay directly to the models

13.2 New system models

For control approaches based on transfer function, the time delay can be introduced to the old system by multiplying plant $G(s)$ with z^{-1}; expressed in mathematical terms, it is:

$$G_{new}(s) = z^{-1}G(s) \tag{13.1}$$

For control approaches based on state-space model, the time delay can be included first in the following way:

$$x(k+1) = \Phi x(k) + \Gamma u(k-1) + 0 \cdot u(k) \tag{13.2}$$

However, the above form could not be used for pole-placement. Thus, by adding a new state, the new augmented system is shown as follow:

$$\begin{bmatrix} x(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} \Phi & \Gamma \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T u(k) \tag{13.3}$$

The new augmented system is equivalent to the old system with an time step delay at controller input. Moreover, since the corresponding controllable matrix has still full rank, we are able to place poles anywhere we would like to. Thereafter, we are going to use the augmented model to investigate the effect of time delay.

13.3 Discretized controller redesign

First, we would like to investigate what would the added z^{-1} from eq.(13.1) bring to the loop transfer. From Figure 13.4, we can find out that the extra time delay wraps the phase at frequency that are closer to the Nyquist frequency (the black vertical line), which means, by using the original controller, we will get a smaller phase margin for the resulting loop transfer.

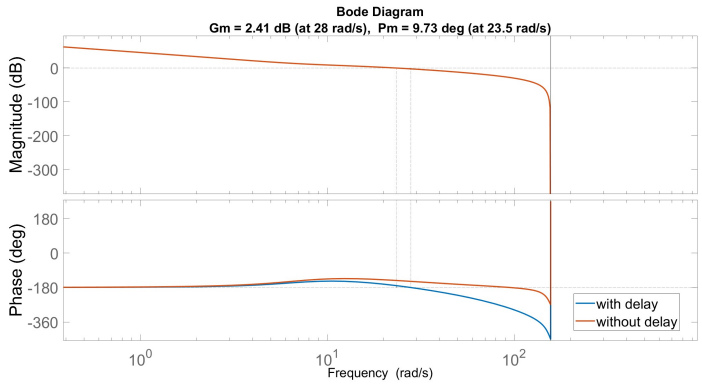


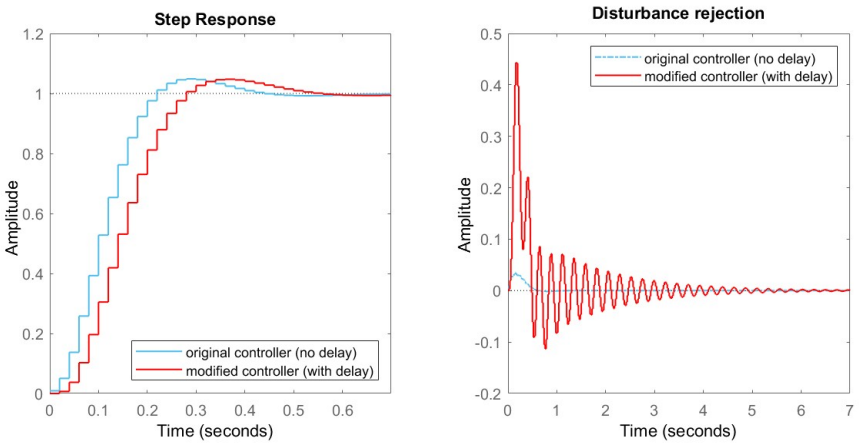
Figure 13.4: Effect of z^{-1} from the perspective of bode diagram

The decrease of phase margin would increase the overshoot, which explains the larger overshoot in Figure 13.2a. After knowing the effect of the delay, we started the redesigning of our controllers for tracking and disturbance rejection.

First, we redesigned the PD-controller for step tracking. The original parameter is $K_P = 0.48$, $K_D = 0.05$ and $T_s = \frac{1}{100}$. Knowing that time delay decrease the original phase margin, we would like to scale down the gain of the controller to restore the original phase margin and render a response as good as possible. As a result, our final design is $K_P = 0.35$, $K_D = 0.037$ and $T_s = \frac{1}{100}$, and the corresponding performance is shown in Figure 13.5a by the red line.

Next, we re-tuned the PID-controller for disturbance rejection task. In order to guarantee the stability of the closed-loop system, we scaled down the gain of the original controller to its 10%. The final parameter setting is $K_P = 2.4$, $K_I = 12$, $K_D = 0.24$ and $T_s = \frac{1}{1000}$. Due to that the larger controller gain, the smaller maximum deviation, the disturbance rejection can not get any further evident improvement with PID configuration; otherwise, system would become unstable. Finally, the performance with the new parameter is shown in Figure 13.5b by the red line. It can be seen that time delay brings longer settling time and more oscillation to the closed-loop system.

However, It is important to point out that the redesigned PID controller is still based on sampling time $h = 0.02s$. Thus, given that the phase next to the Nyquist frequency would be pushed down, if we increase sampling frequency, a better disturbance rejection performance could be achieved. This is also in line with the fact that if we chose a very high sampling time, the effect of the introduced extra time delay would not be very evident from the output of the plant.



(a) Step tracking (b) Disturbance rejection

Figure 13.5: Redesign performance for discretization method

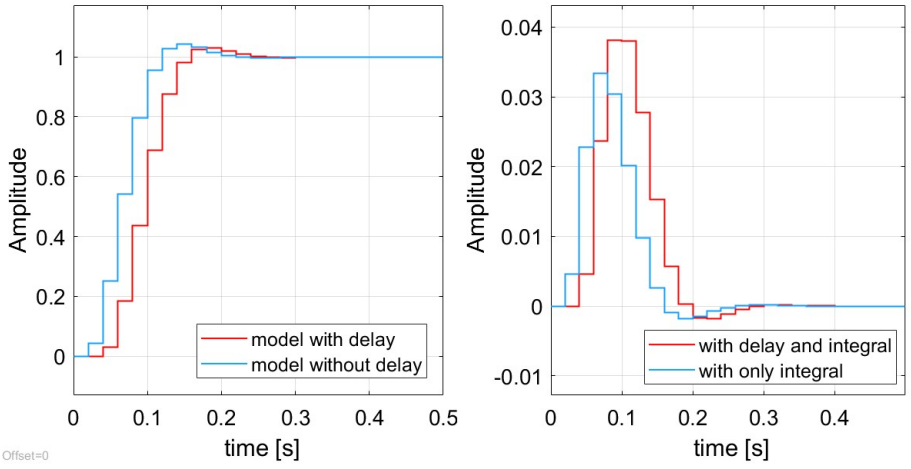
13.4 State-feedback control redesign

For zero-steady-state-error step tracking, we could augment the system into the form shown in eq.(13.3). By doing so, we obtain an augmented Φ matrix of dimension 4×4. Since the delay term has already been included in the new system, we could then place all poles inside the unit circle to guarantee the stability. As a result, four poles are placed at $[0.4,\ 0.5+0.35i,\ 0.5-0.35i,\ 0.3]$. The output tracking simulation is shown in Figure 13.6a by the red line. One can find out that system with delay (red line) shows response one step later than that without delay (blue line) at the start. Similarly, by choosing poles closer to origin closed-loop system could achieve faster response.

For zero-steady-state-error disturbance rejection, we need to augment not only the delay but also the integral action into the original system, which could be done by combining eq.(6.5) and eq.(13.3). The resulting augmented system is in the form of:

$$\begin{bmatrix} x(k+1) \\ x_{n+1}(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} \Phi & 0 & \Gamma \\ -C & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ x_{n+1}(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(k) \tag{13.4}$$

It is worth mentioning that not all 0 and 1 in the above model are scalar, because some of them are supposed to be consistent with system parameter Φ, Γ and C. We can then use the augmented system to design feedback vector L using pole-placement, which also guarantee the stability of the closed-loop system at the same time. For our system, the resulting augmented system is of dimension 5×5. As a result, five poles are placed at $[0.4,\ 0.5+0.35i,\ 0.5-0.35i,\ 0.1,\ 0.2]$. The disturbance rejection simulation is shown in Figure 13.6b by the red line.



(a) Step tracking (b) Disturbance rejection

Figure 13.6: Redesign performance for pole placement

Summary

After doing all the tasks, one can say that controlling the heading angle of a bi-wing aircraft is not very difficult. The fact that this system does not contain any zeros provides great convenience for design. Besides, the locations of the poles does not limit system responsiveness too much, and the response speed of the final designs is consistent with intuitive guesses about controlling an bi-wing aircraft.

For discretizing a continuous controller, different discretization methods would bring different dynamics to the closed-loop system, and an appropriate choice of the sampling time is crucial for different methods.

For methods based on state-space model, since these methods take advantage of more information of the system, the controller would produce better performance compared to just using transfer function. Besides, placing poles of an augmented system will not only guarantee the stability of closed-loop system, but the locations of the poles would also offer an intuitive sense of how the closed-loop would behave.

Last but not least, we can sense the tip of the iceberg of the philosophy behind control science through this project. That is, there is no perfect controller, there is only good controller that satisfies all of our different specifications: for instance, our optimal disturbance rejection controller does not give us the best step tracking performance; another example, if we limit the maximal controller output, then we must expect a slower acceptable response. After all, one should always bare in mind that the process of designing a controller is a process of making trade-offs.