# EE311FZ Control System Design — Project 2

Hanlin CAI (20122161)  20th Dec. 2022

System defined in transfer function form:

$$G(s) = \frac{2.3(s+18)}{(s+1)(s+3)(s+8)} \tag{1}$$

Performance specs:

$$\begin{cases} \text{Settling time: } T_s = 0.9 \\ \text{Peak overshoot: } PO \leq 10\% \\ \text{Control signal limits: } -24 \leq u \leq 24 \end{cases} \tag{2}$$

**All the specs above have been explained in detail in the Producer 3.**

## Procedure 1 — Prior Analysis

First and foremost, as Equation (1) shown, the open-loop three order system is casual, which means that the output of the system is only related to present and past input, not to the future inputs. And we can verify as follows:

$$G(s) = \frac{2.3(s+18)}{(s+1)(s+3)(s+8)} = \frac{Y(s)}{X(s)} \tag{3}$$

Then, we utilize Inverse Laplace Transformation:

$$y(t) = \left(\frac{391}{280}\mathrm{e}^{-\mathrm{t}} - \frac{299}{120}\mathrm{e}^{-5\mathrm{t}} + \frac{23}{21}\mathrm{e}^{-8\mathrm{t}}\right) \cdot x(t) \tag{4}$$

A system is said to be causal if its output depends upon present and past inputs and does not depend upon future input. For non-causal system, the output depends upon future inputs also. As Equation (4) shown, we can know that the system is a **casual system**. Furthermore, we can convert given transfer function into standard form:

$$G(s) = \frac{2.3s + 41.4}{s^3 + 14s^2 + 53s + 40} \tag{5}$$

Moreover, we can use Nyquist plot to analyze the stability of the system.

The following Figure 1 shows the Nyquist plot of the given system. Since the Nyquist plot does not enclose $(-1, 0)$, we can get that:

$$N = 0 \tag{6}$$

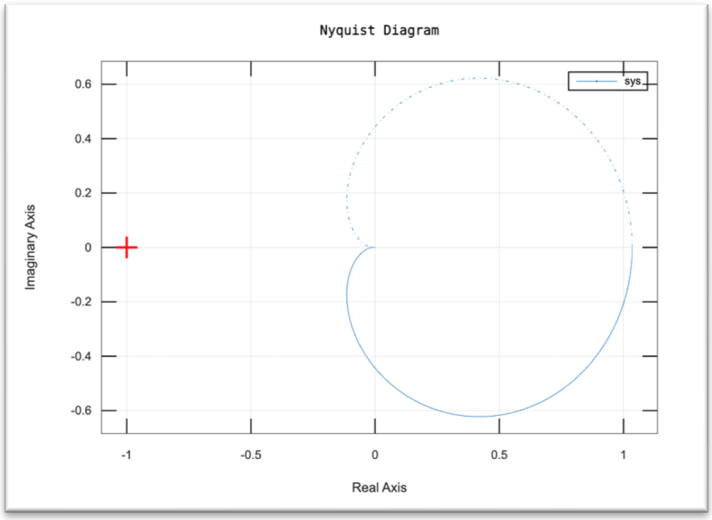


Figure 1 Nyquist plot of the given 3-order system

Considering the poles of the system, easy to get that

$$\begin{cases} p_1 = -1 \\ p_2 = -3 \\ p_3 = -8 \end{cases} \tag{7}$$

All of the poles of the system meet $p < 0$, hence

$$P = 0 \tag{8}$$

We have already known that $Z = P - N$ , so we can get

$$Z = 0 \tag{9}$$

To summarize, since $Z = 0$, we can confirm **the 3-order system is stable**. The corresponding MATLAB program is shown in Table 1.

**Table 1: MATLAB Program 1**

*This program is used to implement inverse Laplace transform and draw Nyquist plot.*

```
%% Procedure 1 Prior Analysis
sqyms s;
TF1 = (2.3*s + 41.4)/( s^3 + 14*s^2 + 53*s + 40);
DE1 = ilaplace(TF1);  % inverse
num1 = [2.3 , 41.4];
den1 = [1, 14, 53, 40];
sys1 = tf(num1,den1);
nyquist(sys1);  % nyquist
```

Now, we have gained a basic knowledge of the system, let's move to next step.

## Procedure 2 — State-space Description

The state-space description in controllable canonical form for a 3-order system can be written as follows,

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_3 & -a_2 & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \tag{10}$$

$$\mathrm{y} = \begin{bmatrix} b_3 - a_3 b_0 & b_2 - a_2 b_0 & b_1 - a_1 b_0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b_0 u \tag{11}$$

As for the given transfer function,

$$G(s) = \frac{2.3s + 41.4}{s^3 + 14s^2 + 53s + 40} = \frac{Y(s)}{X(s)} \tag{12}$$

In this case, we can get the set of corresponding parameters,

$$\begin{cases} a_1 = 14 \\ a_2 = 53 \\ a_3 = 40 \\ b_0 = 0 \\ b_1 = 0 \\ b_2 = 2.3 \end{cases} \tag{13}$$

Now, let's rearrange the Equations above, and we can get:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -40 & -53 & -14 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \tag{14}$$

$$\mathrm{y} = \begin{bmatrix} 41.4 & 2.3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + 0u \tag{15}$$

Also, we can utilize MATLAB to calculate directly, as shown in Table 2.

**Table 2: MATLAB Program 2**

*This program is used to convert transfer function to state-space description.*

```
%% Procedure 2 State-space Description
num2 = [2.3 , 41.4];
den2 = [1, 14, 53, 40];
[A,B,C,D] = tf2ss(num2,den2);
sys2 = ss(A,B,C,D);
Qc = ctrb(A,B);  % controllability matrix
pc = [0 0 1]*inv(Qc);
Pc = inv([pc;pc*A;pc*A*A]);
sys22 = ss2ss(sys2,inv(Pc));  % controllable canonical form
```

## Procedure 3 — Performance Criteria

Considering the specification of a set of performance criteria, the following Table 3 illustrates the summary of the different performance criteria defined by (1) Transient performance specifications and (2) Limits on the control signal, based on Ref.[1].

Table 3 Summary of the different performance criteria (By Hanlin CAI)

| Criteria | Brief Description |
|---|---|
| Transient Performance Specifications: | |
| Delay time, $t_d$ | The time required for the response to reach half the final value the very first time. |
| Rise time, $t_r$ | The time required for the response to rise from 10% to 90%, 5% to 95%, or 0% to 100% of its final value. For underdamped 2nd-order systems, the 0% to 100% rise time is normally used. For overdamped systems, the 10% to 90% rise time is commonly used. |
| Peak time, $t_p$ | The time required for the response to reach the first peak of the overshoot. |
| Overshoot, $M_p$ | The maximum overshoot is the maximum peak value of the response curve measured from unity. The amount of the maximum overshoot directly indicates the relative stability of the system. |
| Settling time, $t_s$ | The time required for the response curve to reach and stay within a range about the final value of size specified by absolute percentage of the final value (usually 2% or 5%). |
| Limits on the Control Signal: | |
| Control limit, $u$ | Control limits are used to detect signals in process data that indicate that a process is not in control and, therefore, not operating predictably. A value in excess of the control limit indicates a special cause is affecting the process. |
| Control Graph | The control chart is a graph used to study how a process changes over time. More detail can be accessed in Ref. [2]. |

Moreover, according to the Ref.[2, 3], the following Table 4 contrasts the control limits and the specification limits.

Table 4 Difference between control limits and specification limits

| Control Limits | Specification Limits |
|---|---|
| Voice of the process | Voice of the Dr. Siyuan Zhan |
| Calculated from Data | Defined by the customer |
| Appear on control charts | Appear on histograms |
| Apply to subgroups | Apply to items |
| Guide for process actions | Separate good items from bad |
| What the process is doing | What we want the process to do |

## Procedure 4 — Controllers Design

# A. PID Controllers

### (a) Open-loop Z-N method

**First and foremost, we can use the open-loop ZN method as following steps:**

1. Use MATLAB Simulink to simulate the given system;
2. Draw the step response diagram of the system;
3. Use for-loop to determine the max slope (point) of the step response curve;
4. Analyze the parameters of the fitted linear equation;
5. Utilize the ZN table to obtain the corresponding parameters of PID controller.

Figure 2 illustrates the Simulink diagram of the given system. Now, we use Simulink to capture the datasets of the system step response.

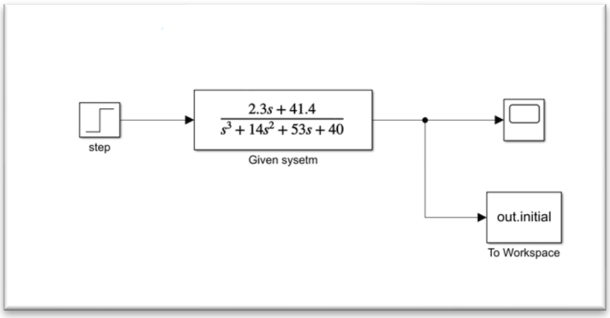


Figure 2 Simulink diagram of the system

Figure 3 shows the step response curve of the system, with the max slope point and the relevant fitted linear equation plot. Table 5 illustrates the relevant MATLAB program.

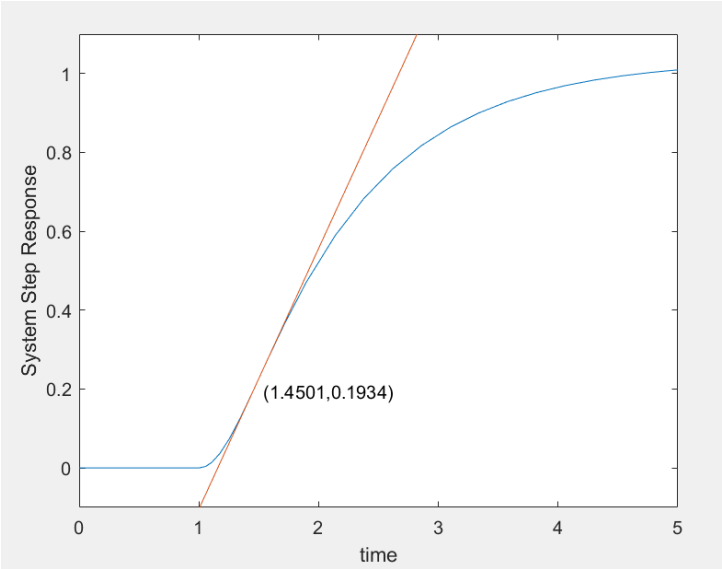


Figure 3 System step response with max slope point

**Table 5: MATLAB Program 3**

*This program is used to find the max slope point of the system step response.*

```
%% Procedure 4-1 Open-loop ZN method
% The data sets were obtained by Simulink (Figure).

time=[0;0.240000000000000;0.480000000000000;0.720000000000000;0.960
000000000000;0.999999999999993;1;1.00000000000001;1.05618173432533;
1.10626211619295;1.17458836333233;1.25586817150269;1.34644068723767
;1.45010569333239;1.57091185431932;1.71585866478841;1.8982108321579
7;2.13821083215797;2.37821083215797;2.61821083215797;2.858210832157
97;3.09821083215797;3.33821083215797;3.57821083215797;3.81821083215
797;4.05821083215797;4.29821083215797;4.53821083215797;4.7782108321
5797;5.01821083215797;5.25821083215797;5.49821083215797;5.738210832
15797;5.97821083215797;6.21821083215797;6.45821083215798;6.69821083
215798;6.93821083215798;7.17821083215798;7.41821083215798;7.6582108
3215798;7.89821083215798;8.13821083215798;8.37821083215798;8.618210
83215798;8.85821083215798;9.09821083215798;9.33821083215798;9.57821
083215798;9.81821083215798;10.0582108321580;10.2982108321580;10.538
2108321580;10.7782108321580;11.0182108321580;11.2582108321580;11.49
82108321580;11.7382108321580;11.9782108321580;12];

out=[0;0;0;0;0;0;0;2.34046557054457e-28;0.00380987847702979;0.01377
44245517423;0.0365675780509246;0.0747908249387998;0.127030268191934
;0.193449674806863;0.273276101769470;0.365919903393339;0.4717215567
80243;0.589268880116967;0.683568065541877;0.758308028425809;0.81727
1725373752;0.863706224494938;0.900248674980520;0.928998768131951;0.
951615842328969;0.969407502894202;0.983403052623937;0.9944123830154
39;1.00307264187492;1.00988504699441;1.01524387619794;1.01945928122
106;1.02277523662095;1.02538365976076;1.02743551824683;1.0290495674
2882;1.03031922358530;1.03131797057083;1.03210361283773;1.032721620
98237;1.03320776344559;1.03359017668171;1.03389099361135;1.03412762
460792;1.03431376515727;1.03446018851098;1.03457536920959;1.0346659
7356330;1.03473724547788;1.03479330995601;1.03483741183999;1.034872
10361339;1.03489939313105;1.03492085982764;1.03493774613059;1.03495
102936801;1.03496147833347;1.03496969778147;1.03497616342877;1.0349
7667719028];

plot(time,out)
hold on;
temp = length(time)-1;

for i=1:temp
 slope(i) = (out(i+1)-out(i)) / (time(i+1)-time(i));
end

[maxSlope,flag] = max(slope);  % Find the max slope

t = time(flag)      % time = 1.4501
y = out(flag)       % out = 0.1934
k = maxSlope        % maxk = 0.6608
b = y - maxSlope*t % b = -0.7648

x = 1:3;
tangent = (k*x + b);
plot(tangent);

text(t,y,"  (1.4501,0.1934)");
axis([0 5 -0.1 1.1]);
xlabel("time");
ylabel("System Step Response");
```

As shown in the Figure 3, the linear equation is fitted as

$$y = 0.6608 * t - 0.7648 \tag{16}$$

Hence, we can obtain that:

$$\begin{cases} R = 0.6608 \\ L = 1.1574 \\ t_i = 2.3148 \\ t_d = 0.57869 \end{cases} \tag{17}$$

Then, we can obtain $k_p, k_i, k_d$ as follows,

$$\begin{cases} k_p = \dfrac{1.2}{RL} = 1.5690 \\ k_i = \dfrac{k_p}{t_i} = \dfrac{k_p}{2L} = 0.67782 \\ k_d = k_p \cdot t_d = k_p \cdot 0.5L = 0.90798 \end{cases} \tag{18}$$

Table 6 Open-loop ZN table

| Controller | $K_p$ | $T_i$ | $T_d$ |
|---|---|---|---|
| P | $T/L$ | $\infty$ | 0 |
| PI | $0.9T/L$ | $L/0.3$ | 0 |
| PID | $1.2T/L = 1.569$ | $2L$ | $0.5L$ |

Finally, as the Table 6 shown, the PID controller of open-loop ZN method is

$$G_c(s) = k_{\mathrm{p}} \cdot \left(1 + \frac{1}{t_s s} + t_d s\right) = 1.569 \cdot (1 + 0.57869 \cdot s + \frac{1}{2.3148 \cdot s}) \tag{19}$$

### (b) Closed-loop Z-N method

**Now, we utilize closed-loop Z-N method through following steps:**

1. Set $k_d = k_i = 0$, hence, $G_c(s) = k_p$;
2. Then, find out $k_{p0} = k_{cr}$ (when system is marginally stable);
3. Lastly, using ZN table to obtain corresponding $k_p, k_i, k_d$.

The characteristic equation $D(s)$ of the given system is defined by

$$D(s) = 1 + k_p \cdot OLTF = 0 \tag{20}$$

$$s^3 + 14 \cdot s^2 + (53 + 2.3 \cdot k_p)s + (40 + 41.4 \cdot k_p) = 0 \tag{21}$$

Table 7 Routh table

| $D(s)$ | $s^3 + 14 \cdot s^2 + (53 + 2.3 \cdot k_p)s + (40 + 41.4 \cdot k_p)$ | |
|---|---|---|
| $s^3$ | 1 | $53 + 2.3 \cdot k_p$ |
| $s^2$ | 14 | $40 + 41.4 \cdot k_p$ |
| $s^1$ | (-702+9.2$k_p$)/(-14) | / |
| $s^0$ | -14 | / |

Table 7 illustrates the Routh table of the characteristic equation above. Now, we let

$$(-702 + 9.2k_p)/14 = 0 \tag{22}$$

Hence, $k_{p0} = k_{cr} = 76.3$, then we consider that

$$14s^2 + 40 + 41.4 \cdot k_p = 0 \tag{23}$$

We can easily get the set of the related parameters

$$\begin{cases} s = 15.12j \\ \omega_c = 15.12 \text{ rad/sec} \\ t_c = \dfrac{2\pi}{\omega_c} = 0.4156 \text{ sec} \end{cases} \tag{24}$$

Then, we can obtain $k_p, k_i, k_d$ as follows,

$$\begin{cases} k_p = 0.6k_{cr} = 45.78 \\ k_i = \dfrac{k_p}{t_i} = \dfrac{k_p}{\frac{t_c}{2}} = 220.3 \\ k_d = k_p \cdot t_d = k_p \cdot (t_c/8\,) = 2.378 \end{cases} \tag{25}$$

Table 8 Closed-loop ZN table

| Controller | $K_p$ | $T_i$ | $T_d$ |
|---|---|---|---|
| P | $0.5k_{cr}$ | $\infty$ | 0 |
| PI | $0.45k_{cr}$ | $1.2P_{cr}$ | 0 |
| PID | $0.6k_{cr} = 45.78$ | $0.5P_{cr}$ | $0.125P_{cr}$ |

In this case, as the Table 8 shown, the PID controller of closed-loop ZN method is

$$G_c(s) = \frac{(\mathrm{k_d} \cdot \mathrm{s}^2 + k_\mathrm{p} \cdot \mathrm{s} + \mathrm{k_i})}{s} = 45.78 + 2.378s + \frac{220.3}{s} \tag{26}$$

### (c) Performance Index (ITAE) method

**Utilizing ITAE method, we should follow these steps:**

1. Determine the closed-loop transfer function;
2. Obtain the corresponding $\omega_c$;
3. Obtain the related parameters of optimum 4-order PID controller;
4. Determine the pre-filter $P(s)$.

The transfer function of the open loop 3-order system is given,

$$G(s) = \frac{2.3s + 41.4}{s^3 + 14s^2 + 53s + 40} \tag{27}$$

The closed-loop transfer function is defined as:

$$G_{CL}(s) = \frac{G_c(s) \cdot G(s)}{1 + G_c(s) \cdot G(s)} \tag{28}$$

where

$$G_c(s) = \frac{\left(\mathrm{k_d} \cdot \mathrm{s}^2 + k_\mathrm{p} \cdot \mathrm{s} + \mathrm{k_i}\right)}{\mathrm{s}} \tag{29}$$

Referring to the equation above, we can get the closed-loop transfer function:

$$G_{CL}(s) = \frac{2.3k_d s^3 + (2.3k_p + 41.4k_d)s^2 + (2.3k_i + 41.4k_p)s + 41.4k_i}{s^4 + (2.3k_d + 14)s^3 + (2.3k_p + 41.4k_d + 53)s^2 + (2.3k_i + 41.4k_p + 40)s + 41.4k_i} \tag{30}$$

Recalling that we have the performance specs

$$\begin{cases} \text{Settling time: } T_s = 0.9 \\ \text{Peak overshoot: } PO \leq 10\% \\ \text{Control signal limits: } -24 \leq u \leq 24 \end{cases} \tag{31}$$

which means,

$$\begin{cases} T_s = 0.9 = \dfrac{4}{\zeta \omega_n} \\ M_p = e^{-\left(\zeta/\sqrt{1-\zeta^2}\right)\pi} \leq 10\% \end{cases} \tag{32}$$

Then, we can obtain the suitable value that

$$\begin{cases} \zeta = 0.59121 \\ \omega_n = 7.5175 \end{cases} \tag{33}$$

According to the EE311FZ PID Control, the optimum 4-order system is given as

$$\mathrm{T(s)} = \frac{\omega_n^4}{s^4 + 2.1\omega_n s^3 + 3.4\omega_n^2 s^2 + 2.7\omega_n^3 s + \omega_n^4} \tag{34}$$

Contrast Equation (30) with Equation (34), we can approximate that

$$\begin{cases} 2.1\omega_n \approx 2.3k_d + 14 \\ 3.4\omega_n^2 = 2.3k_p + 41.4k_d + 53 \\ 2.7\omega_n^3 = 2.3k_i + 41.4k_p + 40 \\ \omega_n^4 = 41.4k_i \end{cases} \tag{35}$$

Now, easy to obtain

$$\begin{cases} k_p = 22.455 \\ k_i = 77.142 \\ k_d = 2.1135 \end{cases} \tag{36}$$

Finally, we can design the related pre-filter $P(s)$ as,

$$P(s) \cdot G_{CL}(s) = \mathrm{T(s)} \tag{37}$$

$$P(s) = \frac{\omega_n^4}{2.3k_d s^3 + (2.3k_p + 41.4k_d)s^2 + (2.3k_i + 41.4k_p)s + 41.4k_i} \tag{38}$$

Where,

$$P(s) = \frac{3193.6973}{4.8611s^3 + 139.15s^2 + 1107.064s + 3193.6788} \tag{39}$$

### (d) Manual method

**Using manual method means that we need to reduce computational complexity as much as possible. In a nutshell, we can follow the widely accepted method below:**

1. Similar to closed-loop ZN method, firstly, we set $k_i = k_d = 0$, then $G_c(s) = k_p$;
2. Increasing $k_p$, until the system becomes marginally stable;
3. Increasing $k_i$, until the output error comes to eliminated;
4. Set a suitable value of $k_i$ to make the system becomes more stable and possess less oscillations;
5. Finally, fine-tune the parameters to make the system more compliant.

**Firstly**, let's set $k_i = k_d = 0$, and generally increase the value of $k_p$ until the system gets into marginally stable. Then, we decrease $k_p$ until the oscillation gets small enough. Repeat this process several times, finally, we have got a general value of $k_p = 22.9$.

**The second step** is to increase $k_i$, until the output error comes to eliminated, which means the steady output equal to 1. In this step, we should be careful to the response time $(t_r, t_s)$ of the system and make them as lower as possible. At the end, we make the value of $k_i = 109$.

**Third**, considering the $k_d$, we can make the system more stable and realizable, through utilizing the differentiation element of PID controller. Now, we fine-tune $k_d$ to decrease the oscillations. Finally, we get a value of $k_d = 3.20$. Now, we have got all of the three parameters of PID controller as follows:

$$\begin{cases} k_p = 22.9 \\ k_i = 109 \\ k_d = 3.20 \end{cases} \tag{40}$$

But we have not finished yet. Now, we need to simulate the system using **MATLAB Simulink**, to see whether the system is stable (Figure 4). Unfortunately, at this moment, although the response time and steady-state error have been good, the system overshoot is too large, which may lead to unsatisfactory impacts.

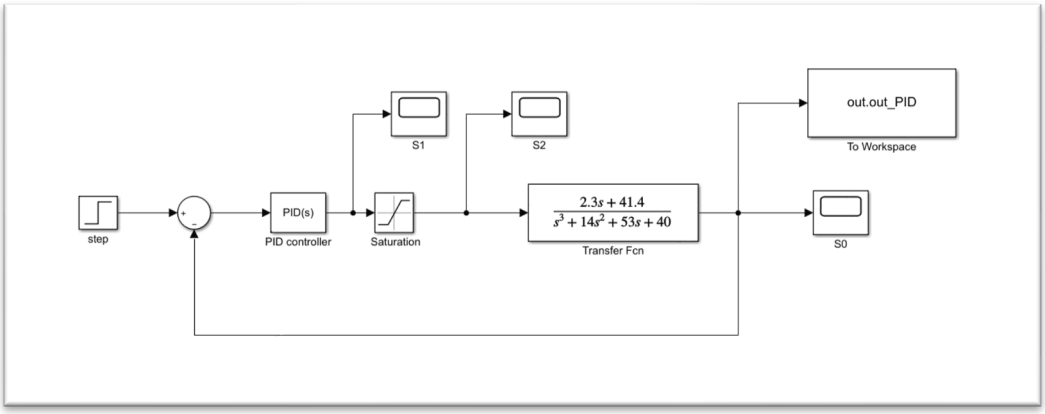


Figure 4 Simulink diagram of PID controller

In this case, according to the Ref. [4], we can try to decrease the value of $k_i$, while increases $k_p$ and $k_d$ (sightly). Through a hundred times of fine tuning. Ultimately, we have obtained a satisfactory set of PID parameters as follows:

$$\begin{cases} k_p = 32.8 \\ k_i = 62.9 \\ k_d = 4.32 \end{cases} \tag{41}$$

Therefore, the corresponding PID controller is

$$G_c(s) = \frac{\left(\mathrm{k_d} \cdot \mathrm{s}^2 + k_\mathrm{p} \cdot \mathrm{s} + \mathrm{k_i}\right)}{s} = 32.8 + 4.32s + \frac{62.9}{s} \tag{42}$$

## B. State Feedback Controller Design

### (a) Controllability and Observability

The transfer function of the 3-order system is given:

$$G(s) = \frac{2.3s + 41.4}{s^3 + 14s^2 + 53s + 40} \tag{43}$$

And we have obtained the corresponding state-space function:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -40 & -53 & -14 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \tag{44}$$

$$\mathrm{y} = \begin{bmatrix} 41.4 & 2.3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{45}$$

which means that

$$\begin{cases} \mathrm{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -40 & -53 & -14 \end{bmatrix} & \mathrm{B} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\ \mathrm{C} = \begin{bmatrix} 41.4 & 2.3 & 0 \end{bmatrix} & \mathrm{D} = 0 \end{cases} \tag{46}$$

Now, to design the State Observer, firstly we need to make sure the given system is observable and controllable. According to the Ref. [1], we calculate that:

$$\begin{cases} \mathrm{M} = \begin{bmatrix} \mathrm{B} & \mathrm{AB} & \mathrm{A^2B} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & -14 \\ 1 & -14 & 143 \end{bmatrix} \\ \mathrm{N} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \\ \mathrm{CA^2} \end{bmatrix} = \begin{bmatrix} 41.4 & 2.3 & 0 \\ 0 & 41.4 & 2.3 \\ -92.0 & -121.9 & 9.2 \end{bmatrix} \end{cases} \tag{47}$$

Using **Octave** online, we can easily determine that

$$\begin{cases} \mathrm{rank(M)} = 3 \\ \mathrm{rank(N)} = 3 \end{cases} \tag{48}$$

Because $\det(\mathrm{M}) = 3$ and $\mathrm{rank(N)} = 3$, thus the system is controllable and observable.

### (b) State Observer Design

According to Ref.[1], the mathematical model of the observer is defined as

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= \mathbf{A}\tilde{\mathbf{x}} + \mathbf{B}u + \mathbf{K}_e(y - \mathbf{C}\tilde{\mathbf{x}}) \\ &= (\mathbf{A} - \mathbf{K}_e\mathbf{C})\tilde{\mathbf{x}} + \mathbf{B}u + \mathbf{K}_e y\end{aligned} \tag{49}$$

In the equation above, where $\tilde{\mathbf{x}}$ is the estimated state and $\mathbf{C}\tilde{\mathbf{x}}$ is the estimated output. The inputs to the observer are the output $y$ and the control input $u$. Matrix $\mathbf{K}_e$, which is called the observer gain matrix, is a weighting matrix to the correction term involving the difference between the measured output $y$ and the estimated output $\mathbf{C}\tilde{\mathbf{x}}$. Also, the observer error equation is defined as:

$$\begin{aligned}\dot{\mathbf{x}} - \dot{\tilde{\mathbf{x}}} &= \mathbf{A}\mathbf{x} - \mathbf{A}\tilde{\mathbf{x}} - \mathbf{K}_e(\mathbf{C}\mathbf{x} - \mathbf{C}\tilde{\mathbf{x}}) \\ &= (\mathbf{A} - \mathbf{K}_e\mathbf{C}) \cdot (\mathbf{x} - \tilde{\mathbf{x}})\end{aligned} \tag{50}$$

Then, defined the difference between $\dot{\mathbf{x}}$ and $\dot{\tilde{\mathbf{x}}}$ as the error vector $\boldsymbol{e}$, so

$$\begin{cases} \mathbf{e} = \mathbf{x} - \tilde{\mathbf{x}} \\ \dot{\mathbf{e}} = (\mathbf{A} - \mathbf{K}_e\mathbf{C})\mathbf{e} \end{cases} \tag{51}$$

We see that the dynamic behavior of the error vector is determined by the eigenvalues of matrix $\mathbf{A} - \mathbf{K}_e\mathbf{C}$. That is, if matrix $\mathbf{A} - \mathbf{K}_e\mathbf{C}$ is a stable matrix, the error vector will converge to zero for any initial error vector $\mathrm{e}(0)$. So far, we have obtained the value of $\mathbf{A}, \mathbf{B}, \mathbf{C}$. Therefore, the problem of designing a full-order observer becomes that of determining the observer gain matrix $\mathbf{K}_e$, such that the error dynamics $\boldsymbol{e}$ defined by Equation above are asymptotically stable with sufficient speed of response.

**Overall, the design of the full-order observer becomes that of determining an appropriate $\mathbf{K}_e$ such that $\mathbf{A} - \mathbf{K}_e\mathbf{C}$ has desired eigenvalues.**

Recall that the closed-loop transfer function of the given system is

$$G_{CL}(s) = \frac{G(s)}{1 + G(s)} = \frac{2.3s + 41.4}{s^3 + 14s^2 + 55.3s + 81.4} \tag{52}$$

Figure 3 shows the Pole-zero map of $G_{CL}(s)$. And the poles of $G_{CL}(s)$ is given as:

$$\begin{cases} p_1 = -8.7369 \\ p_2 = -2.6316 + 1.5465\,j \\ p_3 = -2.6316 - 1.5465\,j \end{cases} \tag{53}$$

As shown in the following Figure 5, the two complex poles $p_2,\ p_3$ are the predominant poles while the real pole $p_1$ will be the non-dominant pole

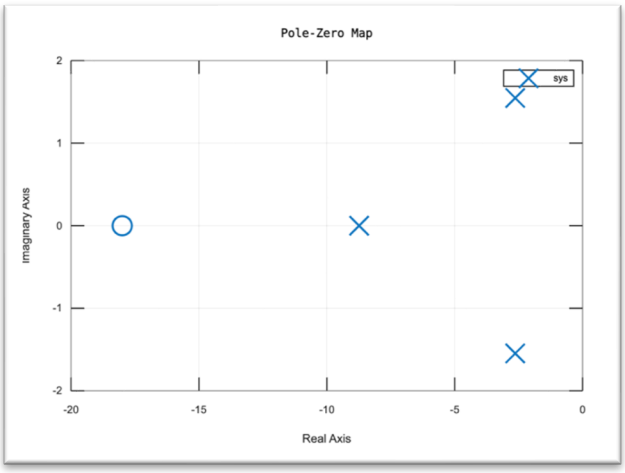


Figure 5 Pole-zero map of $G_{CL}(s)$

In this case, the closed-loop transfer function $\mathrm{G_{CL}(s)}$ can be simplified to:

$$\mathrm{G_{CL}(s)} \approx \frac{2.3s + 41.4}{(s + 2.6316 - 1.5465j)(s + 2.6316 + 1.5465j)} \tag{54}$$

Recall the standard 2-order transfer function is defined as

$$\mathrm{G_{std2}(s)} = \frac{K\omega n^2}{\left(s + \xi\omega n - \omega n\sqrt{\xi^2 - 1}\right)\left(s + \xi\omega n + \omega n\sqrt{\xi^2 - 1}\right)}$$
$$\mathrm{G_{std2}(s)} = \frac{K\omega n^2}{s^2 + 2\xi\omega n + \omega n^2} \tag{55}$$

Comparing the Equations above, we can get

$$\begin{cases} \zeta = 0.8621 \\ \omega_n = 3.0525 \end{cases} \tag{56}$$

Before pole placement, the system possesses

$$\begin{cases} T_s = \dfrac{4}{\zeta\omega_n} = 1.5200\ \mathrm{s} \\ M_p = e^{-\left(\zeta/\sqrt{1-\zeta^2}\right)\pi} \leq 10\% \end{cases} \tag{57}$$

However, at this moment, the settling time does not meet $T_s \leq 0.9$. Thus, the system needs to conduct pole placement. Through analyzing the design specs above, we can obtain the desired placement as follows:

$$\begin{cases} p_1 = -4.4446 + 6.0635\,j \\ p_2 = -4.4446 - 6.0635\,j \end{cases} \tag{58}$$

According to Ref. [1, 5], the desired poles for observer should be twice as the poles of closed-loop system. Therefore, the poles placement for the observer should be

$$\begin{cases} p_{o_1} = -8.8892 + 12.127\,j \\ p_{o_2} = -8.8892 - 12.127\,j \end{cases} \tag{59}$$

Once we get the desired poles for the observer, we can obtain the $\mathbf{K}_e$ through MATLAB program, as shown in the following Table 9.

$$\mathbf{K}_e = \begin{bmatrix} 0.0155 \\ 5.1623 \\ -26.314 \end{bmatrix} \tag{60}$$

**Table 9: MATLAB Program 4**

*This program is used to obtain the observer gain matrix.*

```
%% Q4-(b) State Obverser Design
clear;clc;
A = [0 1 0;0 0 1;-40 -53 -14];
B = [0;0;1];
C = [41.4 2.3 0];
% pole1 = [-4.4446 + 6.0635j,-4.4446-6.0635j,-8.7369];
% K = acker(A',B,pole1)
pole2 = [-8.8892 + 12.127j,-8.8892-12.127j,-8.7369];
Ke = acker(A',C',pole2)
```

Recall the mathematical model of the observer,

$$\begin{aligned} \dot{\tilde{\mathbf{x}}} &= \mathbf{A}\tilde{\mathbf{x}} + \mathbf{B}u + \mathbf{K}_e(y - \mathbf{C}\tilde{\mathbf{x}}) \\ &= (\mathbf{A} - \mathbf{K}_e\mathbf{C})\tilde{\mathbf{x}} + \mathbf{B}u + \mathbf{K}_e y \end{aligned} \tag{61}$$

Now we have already got all of the parameters, let's rearrange the Equation as follows

$$\begin{bmatrix} \dot{\tilde{x}} \\ \ddot{\tilde{x}} \\ \dddot{\tilde{x}} \end{bmatrix} = \begin{bmatrix} -0.642 & 0.964 & 0 \\ -213.719 & -11.873 & 1 \\ 1049.404 & 7.522 & -14 \end{bmatrix} \begin{bmatrix} \widetilde{x_1} \\ \widetilde{x_2} \\ \widetilde{x_3} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0.0155 \\ 5.1623 \\ -26.314 \end{bmatrix} \mathrm{y} \tag{62}$$

## (c) Feedback Controller Design

**After implementing the observer, the next step is to design the feedback controller. Referring to Ref.[1] (heavily), we shall use the following design procedure:**

1. Derive a state-space model of the plant;
2. Choose the desired closed-loop poles for pole placement. Choose the desired observer poles;
3. Determine the state feedback gain matrix **K** and the observer gain matrix $\mathbf{K}_e$.
4. Using the gain matrices **K** and $\mathbf{K}_e$ obtained in step 3, derive the transfer function of the observer controller. If it is a stable controller, check the response to the given initial condition. If the response is not acceptable, adjust the closed-loop pole location and/or observer pole location until an acceptable response is obtained.

Based on the Procedure above, we have already obtained the desired gain matrix. Now we need to adjust the closed-loop pole location or observer pole location. Also, we utilize the **MATLAB Simulink** for system simulation, as shown in Figure 6.

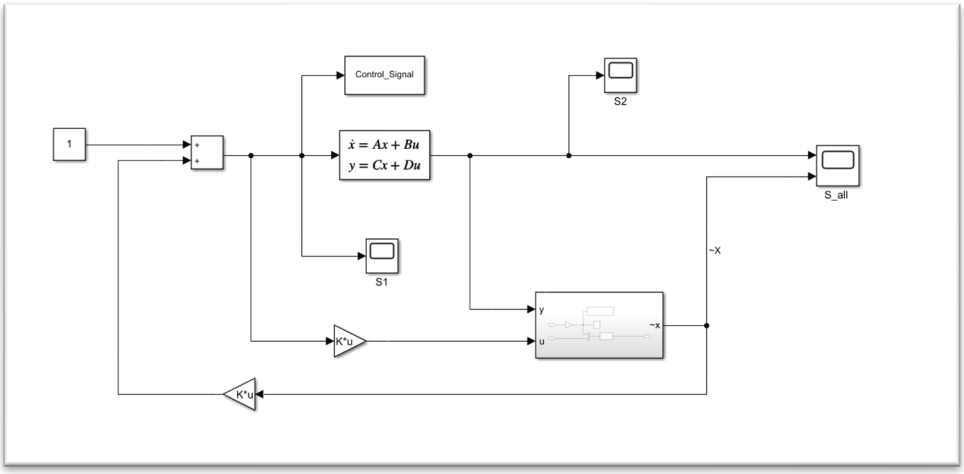


Figure 6 Simulation of the system

As for the pole placement, firstly we need to obtain the value of initial point $x_i$ and $u_i$, where the expected value would be

$$\begin{cases} \mathbf{A}x_i + \mathbf{B}u_i = 0 \\ \mathbf{C}x_i = r \end{cases} \tag{63}$$

which is

$$\begin{bmatrix} x_{SS} \\ u_{SS} \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{64}$$

Substituting the matrix above, we get that

$$\begin{bmatrix} x_i \\ u_i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -40 & -53 & -14 & 1 \\ 41.4 & 2.3 & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.024 \\ 0 \\ 0 \\ 0.966 \end{bmatrix} \tag{65}$$

So, we can know that

$$\begin{cases} x_i = 0.024 \cdot x \\ u_i = 0.966 \cdot u \end{cases} \tag{66}$$

Now, we can utilize the MATLAB function to obtain the state feedback gain matrix **K**, as shown in the following Table 10.

$$\mathbf{K} = \begin{bmatrix} 13.003 \\ -11.345 \\ 0 \end{bmatrix} \tag{67}$$

**Table 10: MATLAB Program 5**

*This program is used to obtain the state feedback gain matrix.*

```
%% Q4-(c)Feedback Controller Design
clear;clc;
A = [0 1 0;0 0 1;-40 -53 -14];
B = [0;0;1];
C = [41.4 2.3 0];
pole1 = [-4.4446 + 6.0635j,-4.4446-6.0635j,-8.7369];
K = acker(A',B,pole1)
pole2 = [-8.8892 + 12.127j,-8.8892-12.127j,-8.7369];
Ke = acker(A',C',pole2)
```

Then, through a hundred times of fine tuning. Ultimately, we obtained a satisfactory value of $\mathbf{K}_e$ and **K** as follows:

$$\begin{cases} \mathbf{K}_e = \begin{bmatrix} 0.0155 \\ 5.1623 \\ -26.314 \end{bmatrix} \\ \mathbf{K} = \begin{bmatrix} -1.5020 \\ -11.345 \\ 0 \end{bmatrix} \end{cases} \tag{68}$$

**Now, we have finished all the designs of the controllers based on various methods. In the next section, we will compare the pros and cons of each method in detail.**

## Procedure 5-1 — Time-domain Simulink Simulation

**In this section, we will simulate all the systems we design in Procedure 4 above.**

### (a) PID Controller

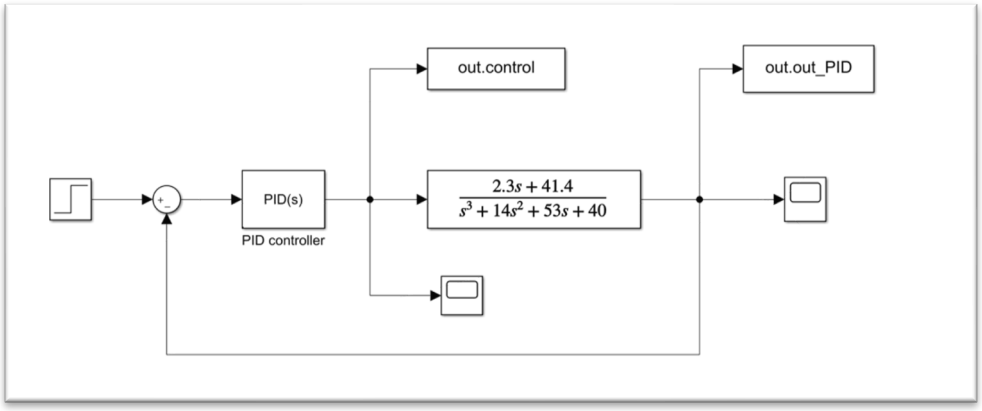


Figure 7 Block diagram for PID controller

### (b) PID Controller with control signal limit

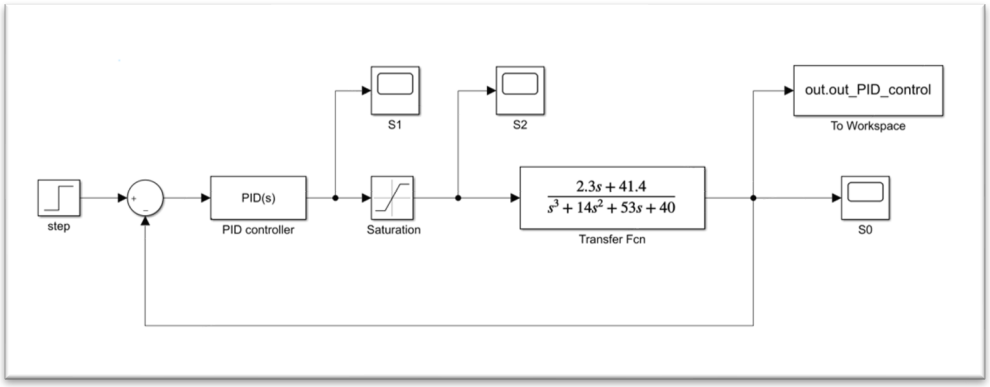


Figure 8 Block diagram for PID controller (control signal limit)

### (c) ITAE-PID Controller

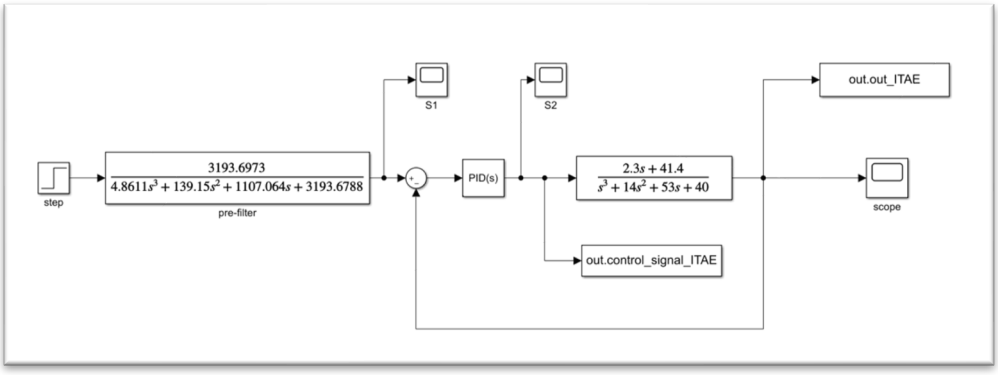


Figure 9 Block diagram for ITAE-PID controller

## (d) ITAE-PID Controller with control signal limit

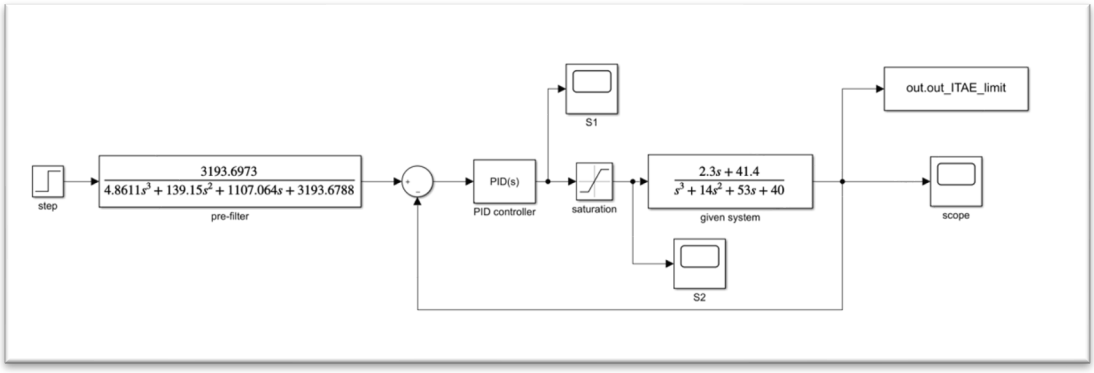


Figure 10 Block diagram for ITAE-PID controller (control signal limit)

## (e) State-space Feedback Controller

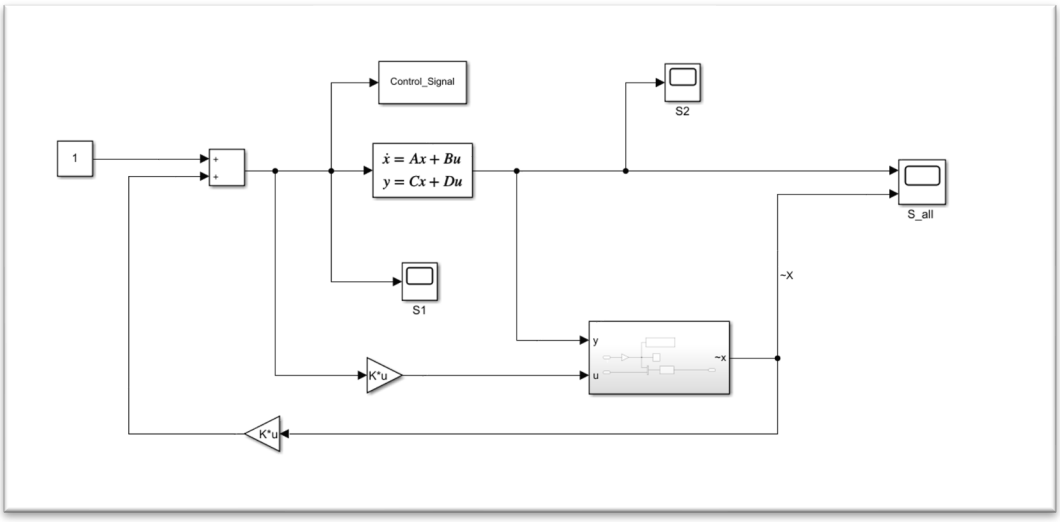


Figure 11 Block diagram for the State-space feedback controller

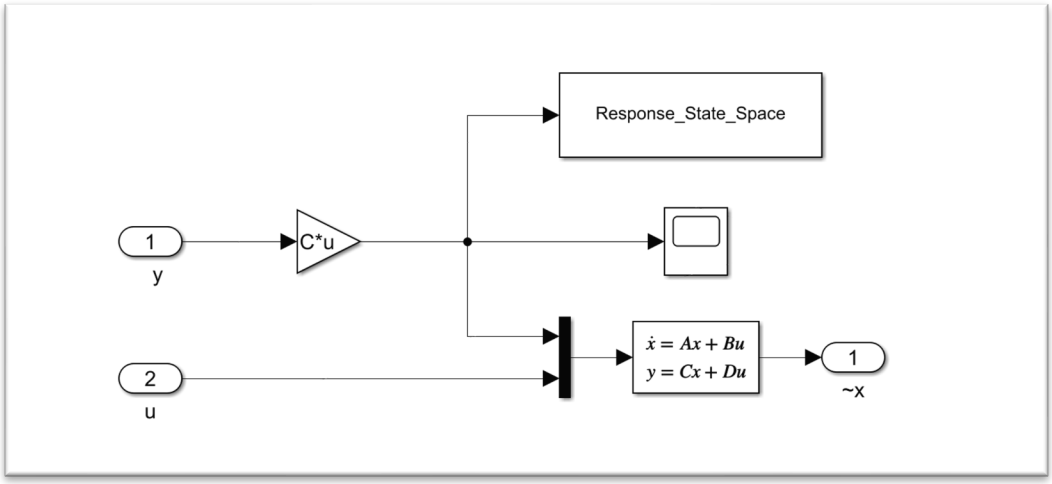


Figure 12 Subsystem of the state-space feedback controller (State-space Observer)

## (f) State-space Feedback Controller with control signal limit

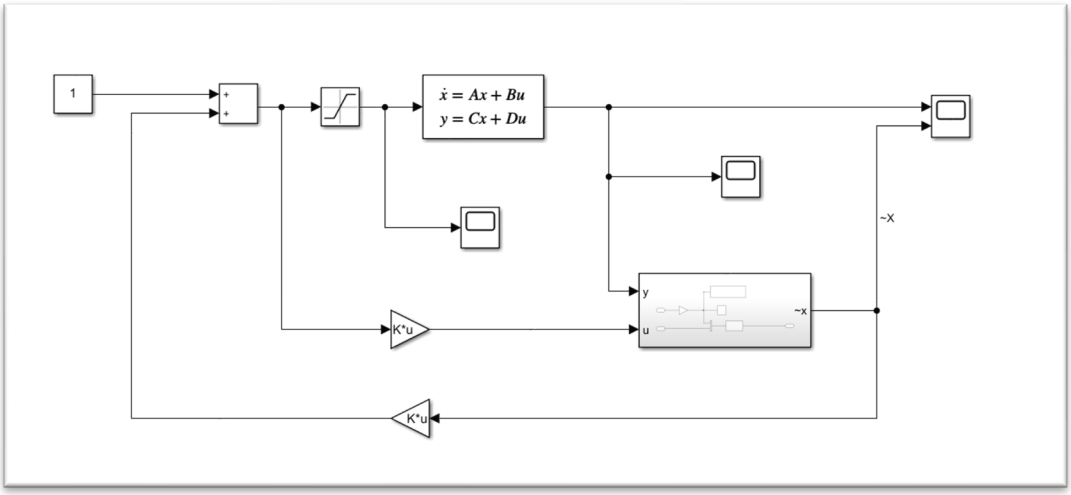


Figure 13 Block diagram for the State-space feedback controller (control signal limit)

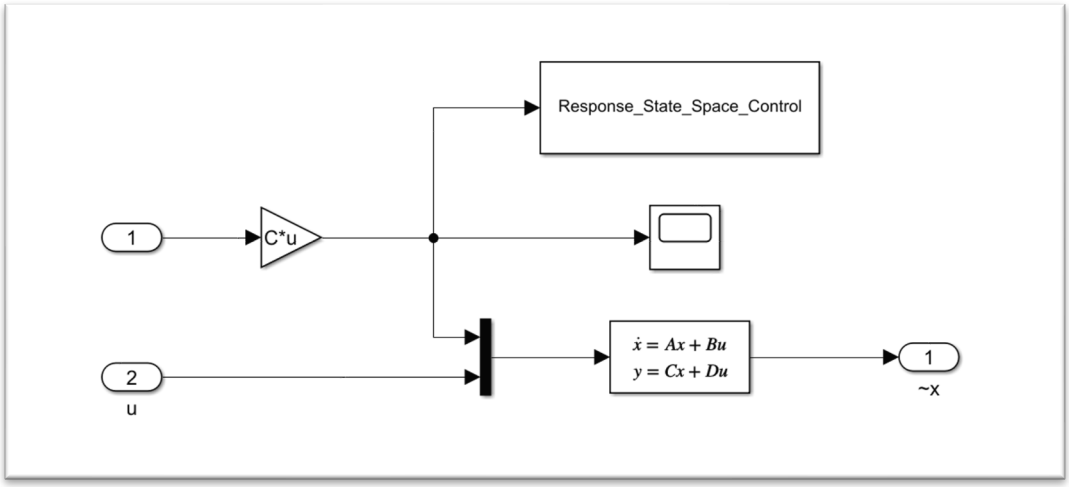


Figure 14 Subsystem of the state-space feedback controller (State-space Observer)

## Procedure 5-2 — Time-domain Simulink Simulation

### (1) PID Controller: open-loop ZN method

Recall that the PID controller of open-loop ZN method is

$$G_c(s) = 1.569 \cdot (1 + 0.57869 \cdot s + \frac{1}{2.3148 \cdot s}) \tag{69}$$

Figure 15 shows the output response of the system.

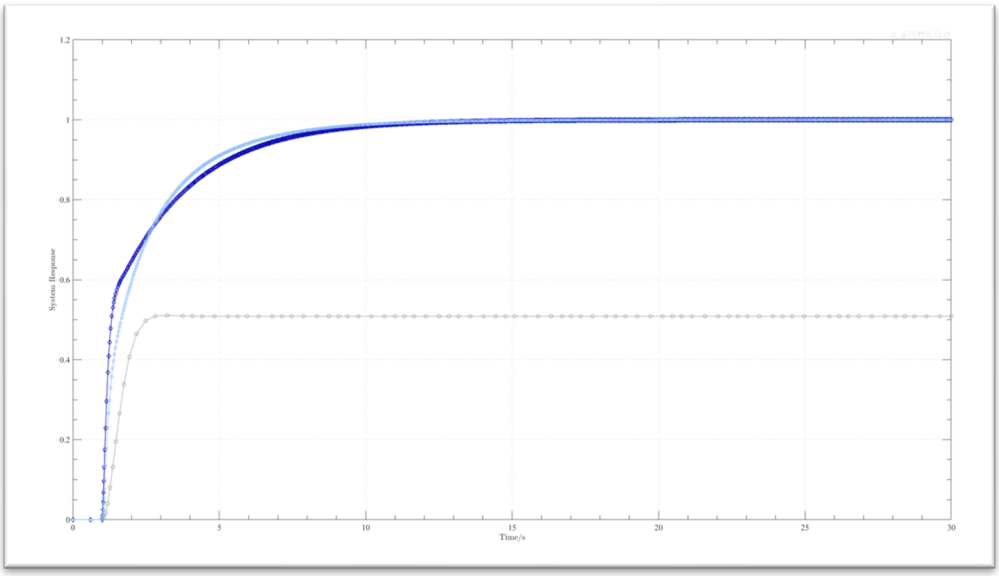

Figure 15 System output response (open-loop ZN method)

And Table 11 illustrates the performance specs of the system above.

Table 11 Performance specs

| System | $T_s$ | $M_p$ | $u_{max}$ | $e_{ss}$ |
|---|---|---|---|---|
| Without limit | 9.4568s | 0% | / | 0 |
| With limit | 8.6679s | 0% | 92.4 | 0 |

### (2) PID Controller: closed-loop ZN method

Recall that the PID controller of open-loop ZN method is

$$G_c(s) = 45.78 + 2.378s + \frac{220.3}{s} \tag{70}$$

Figure 16 shows the output response of the system.

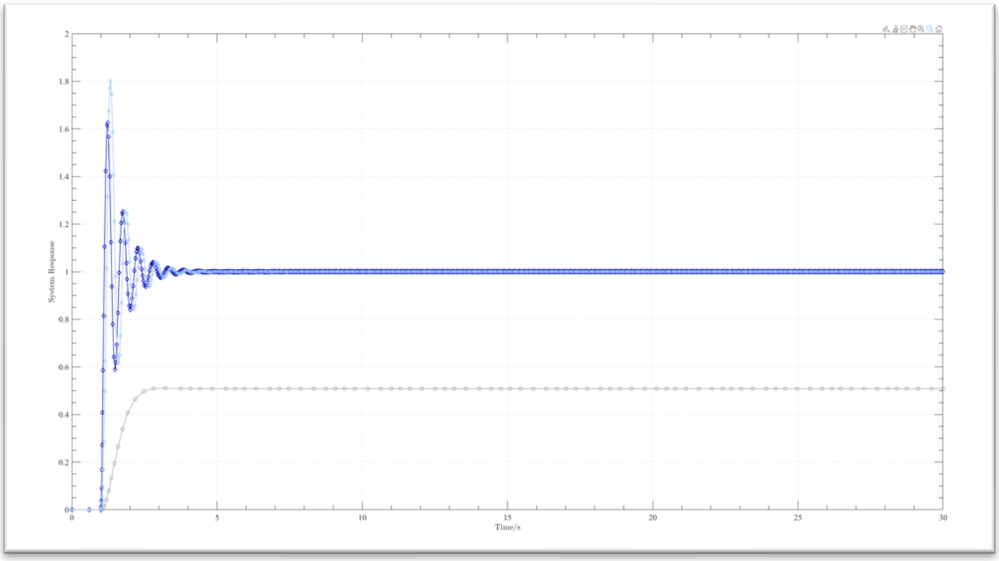


Figure 16 System output response (closed-loop ZN method)

And Table 12 illustrates the performance specs of the system above.

Table 12 Performance specs

| System | $T_s$ | $M_p$ | $u_{max}$ | $e_{ss}$ |
|---|---|---|---|---|
| Without limit | 3.1278s | 62.9% | / | 0 |
| With limit | 3.2601s | 79.8% | 283.3 | 0 |

### (3) PID Controller: ITAE method

Recall that the PID controller of ITAE method is

$$G_c(s) = 22.455 + 2.1135s + \frac{77.142}{s} \tag{71}$$

Figure 17 shows the output response of the system.

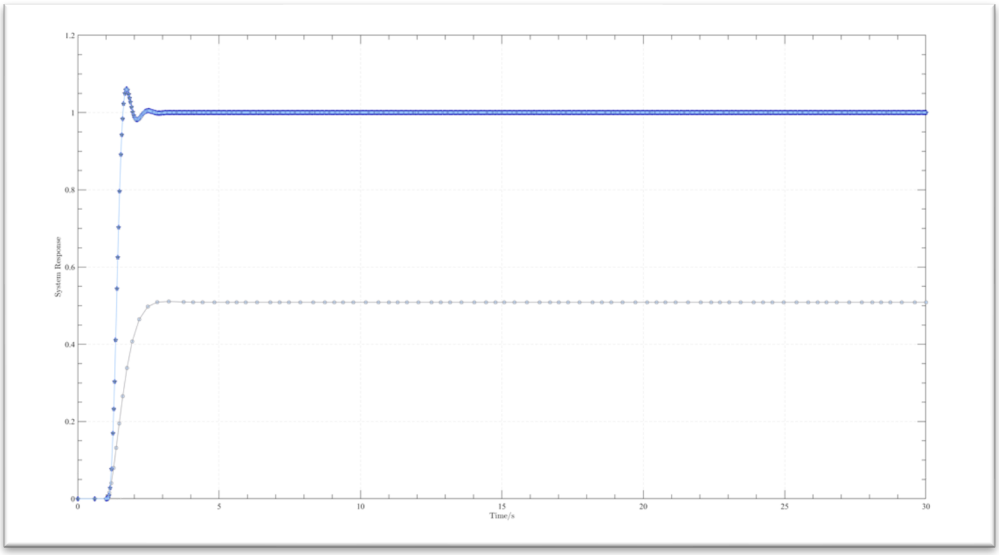

Figure 17 System output response (ITAE method)

And Table 13 illustrates the performance specs of the system above.

Table 13 Performance specs

| System | $T_s$ | $M_p$ | $u_{max}$ | $e_{ss}$ |
|---|---|---|---|---|
| Without limit | 1.9006s | 6.14% | / | 0 |
| With limit | 1.0906s | 6.14% | 6.013 | 0 |

### (4) PID Controller: mutual method

Recall that the PID controller of ITAE method is

$$G_c(s) = 32.8 + 4.32s + \frac{62.9}{s} \tag{72}$$

Figure 18 shows the output response of the system.

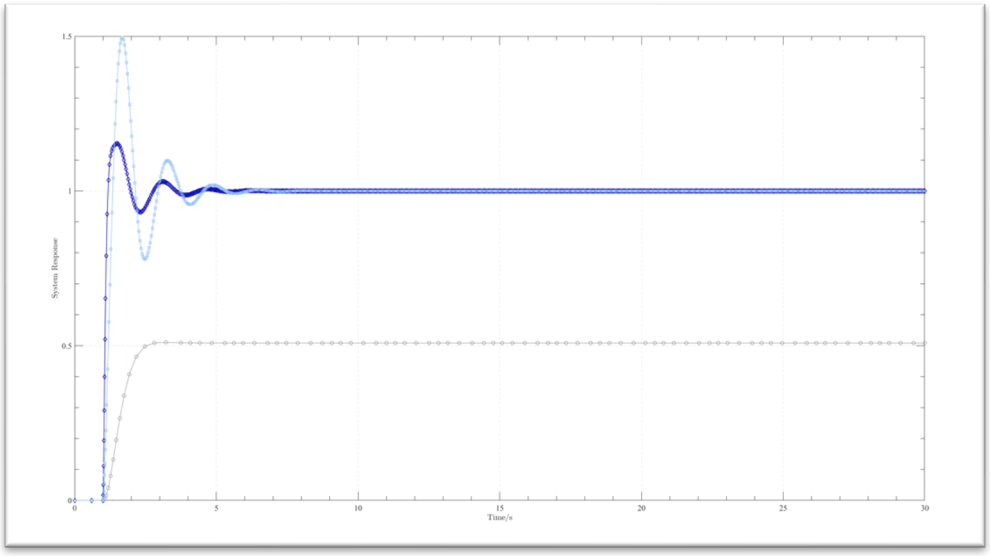

Figure 18 System output response (mutual method)

And Table 14 illustrates the performance specs of the system above.

Table 14 Performance specs

| System | $T_s$ | $M_p$ | $u_{max}$ | $e_{ss}$ |
|---|---|---|---|---|
| Without limit | 3.3485s | 19.24% | / | 0 |
| With limit | 4.3940s | 49.21% | 429.1 | 0 |

### (5) State-space: Feedback Controller

Recall the gain matrices of the state feedback controller system is

$$\begin{cases} \mathbf{K}_e = \begin{bmatrix} 0.0155 \\ 5.1623 \\ -26.314 \end{bmatrix} \\ \mathbf{K} = \begin{bmatrix} -1.5020 \\ -11.345 \\ 0 \end{bmatrix} \end{cases} \tag{73}$$

Figure 19 shows the output response of the system above.

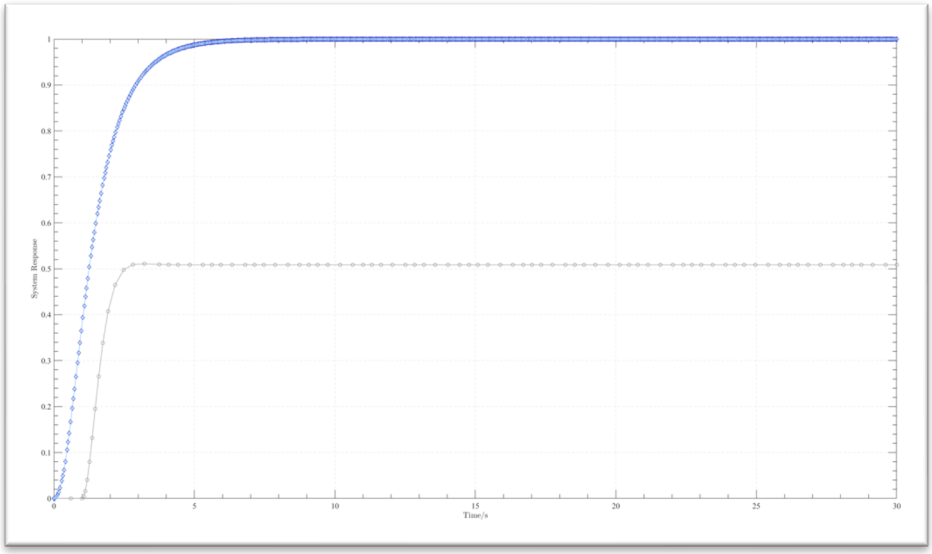

Figure 19 System output response (State feedback controller)

And Table 15 illustrates the performance specs of the system above.

Table 15 Performance specs

| System | $T_s$ | $M_p$ | $u_{max}$ | $e_{ss}$ |
|---|---|---|---|---|
| Without limit | 4.4953s | 0% | / | 0.0003% |
| With limit | 4.4953s | 0% | 1.002 | 0.0003% |

**So far, we have presented all the performance specs of the controller we designed. In the following section, we will summarize and compare all the methodologies.**

## Procedure 6 — Comparison of Methods

**Congratulations!** We have come a long way to get here. Now, we will compare the system performance of all the controllers we design above. The following Table16 and Table 17 illustrate the performance specs of various methods. And the Table 18 shows the summary for the control signal limit of the different methods.

Table 16 Summary of various design methodologies (without control signal limit)

| Without limit | $T_r$ | $T_s$ | $M_p$ | $e_{ss}$ |
|---|---|---|---|---|
| Open loop ZN | 4.4358s | 9.4568s | 0% | 0% |
| Closed loop ZN | 0.6065s | 3.1278s | 62.9% | 0% |
| ITAE | 0.3321s | 1.9006s | 6.14% | 0% |
| Manual | 0.1349s | 3.3485s | 14.24% | 0% |
| State Feedback | 2.4987s | 4.4953s | 0% | 0.00037% |

Table 17 Summary of various design methodologies (with control signal limit)

| Without limit | $T_r$ | $T_s$ | $M_p$ | $e_{ss}$ |
|---|---|---|---|---|
| Open loop ZN | 3.5867s | 8.6689s | 0% | 0% |
| Closed loop ZN | 0.1408s | 3.2601s | 79.8% | 0% |
| ITAE | 0.3321s | 1.9006s | 6.14% | 0% |
| Manual | 0.2502s | 4.3940s | 49.12% | 0% |
| State Feedback | 2.4987s | 4.4953s | 0% | 0.00037% |

Table 18 Summary for the control signal limit of the different methods

| Methods | Open-loop ZN | Closed-loop ZN | ITAE | Manual | State Feedback |
|---|---|---|---|---|---|
| $u_{max}$ | 92.4 | 283.3 | 6.013 | 397.1 | 1.002 |

## Procedure 7 — Brief Conclusion

**As compared in the Table 16-18 above, we can draw the conclusion as follows:**

1. **As for rise time $t_r$**, ITAE controller performs better than any other controller. And the manual and closed-loop controllers share the rank 2nd. The state feedback and open-loop controller do not perform well.
2. **As for settling time $t_s$**, ITAE controller outperform other controller, while open-loop controller possesses an unsatisfactory performance. And the other controller performs similarly.
3. **Considering maximum overshoot $M_P$**, obviously, open-loop controller and state feedback controller have the best performance. ITAE controller also perform well. However, the closed-loop and manual controller would not be the suitable option.
4. **In the domain of steady state error $e_{ss}$**, all five controllers perform satisfactorily.

**As shown in the following Table 19, all the methodologies have been summed up.**

Table 19 Summary of various controller design methodologies

| Methodology | $T_r$ | $T_s$ | $M_p$ | $e_{ss}$ |
|---|---|---|---|---|
| Open loop ZN | 5th | 5th | 1st ✓ | 1st ✓ |
| Closed loop ZN | 3rd | 2nd | 5th | 1st ✓ |
| **ITAE (SOTA)** | **1st ✓** | **1st ✓** | **3rd** | **1st ✓** |
| Manual | 2nd | 2nd | 4th | 1st ✓ |
| State Feedback | 4th | 2nd | 1st ✓ | 5th |

## Acknowledgements

At the end, I gratefully acknowledge Dr. Siyuan Zhan for his generous guidance and support during this semester. And I hope to thank Mr. Jian Lang for his helpful suggestions. Also, I would like to thank the reviewer who carefully evaluates this report. Best wishes!

Hanlin CAI

20th Dec. 2022

# References


[1] Ogata K. Modern control engineering [M]. Prentice hall Upper Saddle River, NJ, 2010.

[2] CONTROL CHART by Northwest Analytics Inc. Available online URL: https://asq.org/quality-resources/control-chart [Z]. 2022

[3] Control Limits Vs. Specification Limits by Northwest Analytics Inc. Available online URL: https://www.nwasoft.com/resources/information-center/article/control-limits-vs-specification-limits [Z]. 2022

[4] How to Tune a PID Controller. Available online URL: https://pidexplained.com/how-to-tune-a-pid-controller/ [Z]. 2022

[5] Pole Placement Method. Available online URL: https://www.sciencedirect.com/topics/engineering/pole-placement-method [Z]. 2013