UNIVERSITY OF LIVERPOOL

Experiment 81 - Design of a Feedback Control System

ELEC273*

March 16, 2020

Abstract

In this report, P model, PI model and PID model was utilized to design controllers in Matlab SimuLink. A high order plant was set to be simulated to find its unit-step response directly. FOPDT method was used to approximate the original response and to find the suitable parameters. It was investigated that comparing with P model, PI model seems to have higher OS% but with nearly removed steady error. PI model shows more expected ability to against disturbance because of less Max value, nearly zero steady error compared with P model. Finally, a PID model was designed to make the anti-disturbance ability of the system to be as high as possible.

Declaration

I confirm that I have read and understood the University's definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with ant other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as described in the Code of Practice on Assessment (Appendix L).

*IMPORTANT: In a standard technical report, you would need to include here your personal details as the author of the document. However, remember that marking of coursework is anonymous and therefore you should remove this part before submitting your report for Year 2 labs! Do not include your name, student ID, email address or any other personal information.

Contents

1 Introduction **1**
- 1.1 Background . . . 1
- 1.2 Objective . . . 1
- 1.3 method . . . 1

2 Part 1 **2**
- 2.1 Open Loop Respond . . . 2
  - 2.1.1 Method and procedure . . . 2
  - 2.1.2 Result and Comment . . . 2
- 2.2 FOPDT Approximation . . . 3
  - 2.2.1 Method and procedure . . . 3
  - 2.2.2 Result and Comment . . . 4

3 Part 2 **5**
- 3.1 Method and procedure . . . 5
- 3.2 Result and Comment . . . 6

4 Part 3 **7**
- 4.1 Method and procedure . . . 7
- 4.2 Result and Comment . . . 7
  - 4.2.1 Change K_I . . . 7
  - 4.2.2 Change K_p . . . 8

5 Part 4 **9**
- 5.1 Method and procedure . . . 9
- 5.2 Result and Comment . . . 9

6 Bonus **10**
- 6.1 Method and procedure . . . 10
- 6.2 Result and Comment . . . 11

7 Discussion and Conclusion **11**

References **11**

Appendices **12**

A Part 1 Screenshot **12**

B Part 2 Screenshot **13**

C Part 3 Screenshot **13**

D Part 4 Screenshot **15**

E Part 5 Screenshot **16**

1 Introduction

In this lab, control systems with a proportional and proportional-integral control was designed and simulated with MATLAB Simulink.[1]

1.1 Background

Control systems are everywhere and generally used to achieve control behavior to make the system acting as it is supposed to do. In general, there are two types of systems. One is open-loop systems which do not have a feedback from the result. They act with the logic set but can not fix errors automatically. This may mean that, if the errors of the system can be accumulated while the system continuing. Therefore, this kind of system is not suitable for some persistent running cases. The other type is close-loop systems, which have the output of their result as one of their input. This feedback mechanism makes them can automatically adjust themselves with a typical error of the output.

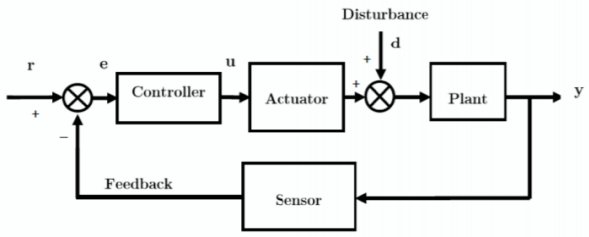

Figure 1: Control Model [1]

Figure 1 displays the general structure of a close-loop control model. In this model, the output is sent back to the input before the controller. After that, there could be an actuator and some disturbance being input into the system. Then, the plant which indicates the object of machine that needed to be controlled. The output of the result will be collected by sensors with possible errors.

This kind of control behavior is widely used in industry for achieving the control process with feedback to provide a sustainable and reliable control service. Therefore, it is important to learn and practice the design of such close-loop control systems.

1.2 Objective

The objectives of this lab was, firstly, get a approximate model of a high order open-loop system using First-order Plus Time delay (FOPDT) model. After that, proportional (P) controller, proportional-integral (PI) controller and proportional-integral-derivative (PID) controller will be designed and evaluated.

1.3 method

$$G(s) = \frac{K}{(Ts+1)^2} \tag{1}$$

According to the lab script [1], the plant can be described by a transfer function, which is shown as Equation 1. In this equation, K presents the day of birth while T indicate the month

of birth, in this case it is 4 and 9 accordingly.

2 Part 1

2.1 Open Loop Respond

2.1.1 Method and procedure

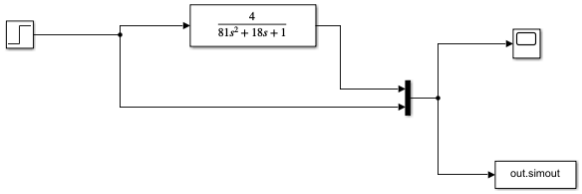

Figure 2: Open Loop Simulink model

Figure 2 shows the Simulink model of the direct simulation of open loop respond of the plant. As it shown in the picture, an unit-step change was generated and be input into the plant transfer function. After the plant, the signal was import into a scope to display the respond. Meanwhile, the data was also output to the workplace via a simout module.

2.1.2 Result and Comment

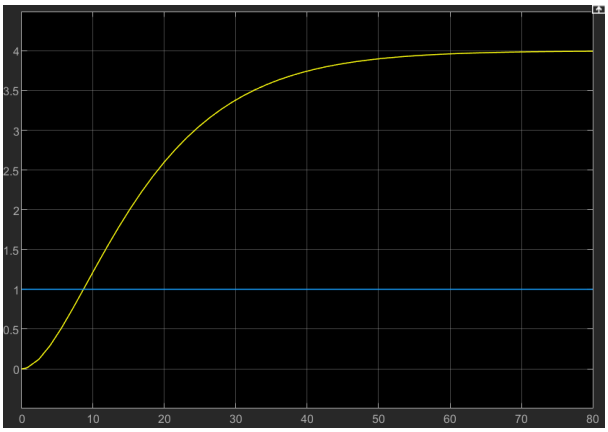

Figure 3: Result of Open Loop Simulink model

As it shown in the Figure 3, the yellow line presents the respond of the transfer function while the blue line indicates the input unit step.

From Figure 3, it can be indicated that when the input jumped from 0 to 1, the overall response begin to increase from 0. The increase of the respond with time firstly become quick and then become slow before approaching 4. Finally, after around time 70, the result was stable at 4.

Theoretically, the expected result is a expression of the convolution of unit-step and plant. In transfer function, it should be the multiple of the transfer function of unit-step and the plant,

as it shown in Equation 2.

$$Y(s) = \frac{K}{T^2 s^3 + 2Ts^2 + s} \tag{2}$$

Equation 2 can be classified into the following form with factorization.

$$Y(s) = \frac{K}{s} - \frac{K}{s + \frac{1}{T}} - \frac{\frac{K}{T}}{(s + \frac{1}{T})^2} \tag{3}$$

In the time domain, Equation 3 can be expressed as Equation 4.

$$y(t) = K - Ke^{-\frac{1}{T}t} - \frac{K}{T}e^{-\frac{1}{T}t}t \tag{4}$$

From Equation 4, it can be inferred that this function is convergence at K. This result was similar to simulation because the convergence value is about 4 according to Figure 3.

2.2 FOPDT Approximation

2.2.1 Method and procedure

From the Equation 4, the derivative and second order derivative can be calculated as Equation 5 and Equation 6.

$$y'(t) = \frac{K}{T^2}e^{-\frac{1}{T}t}t \tag{5}$$

$$y''(t) = \frac{K}{T^2}e^{-\frac{1}{T}t}(1 - \frac{t}{T}) \tag{6}$$

From Equation 6, it can be found that the inflection point is achieved at time T, while the slope is k/Te, in this case, they are 9 and 0.1635. The tangent line pass by (t, 0.2642K), which is (9, 1.057) in this case.

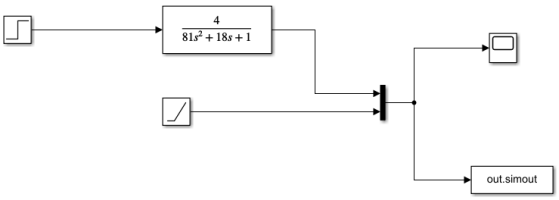

Figure 4: Tangent Line for Response

We use the simulation in Figure 4 to verify the computation of the tangent line. As it shown from the following picture 5, the tangent line can suit the original respond.

The parameters of FOPDT including the process delay time t_d, the process time T_p and the process gain K_p can be then calculated.

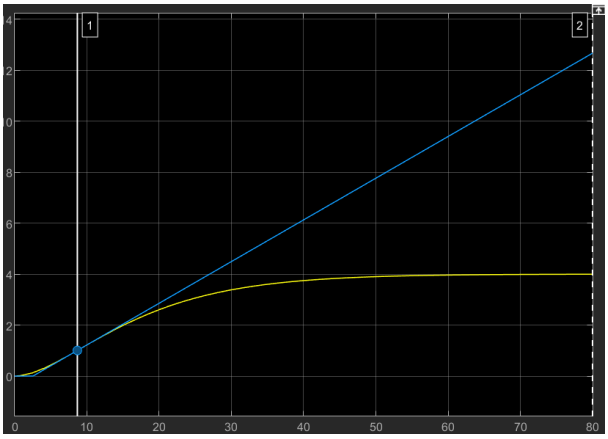

Figure 5: Tangent Line for Response Result

Table 1: Parameters of FOPDT

t_d	T_p	K_p
2.535	24.465	4

$$G(s) = \frac{K_p e^{-t_d S}}{T_p s + 1} \tag{7}$$

With these parameters and Equation 7, we built the FOPDT model to approximate the result. The build method was shown in the following Figure 6.

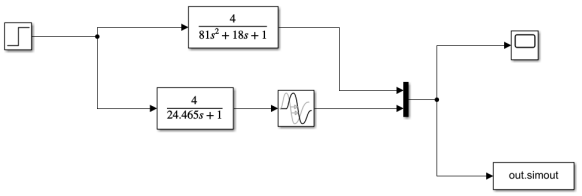

Figure 6: Simulink of FOPDT

As it shown in Figure 6, a trans fcn module was firstly used to simulate the G(s) and then a delay module was used to simulate the delay part. After that, these two signal were sent to one scope to display.

2.2.2 Result and Comment

Figure 7 shows the result of FOPDT model, while the yellow line shows the direct simulation of the plant and the blue line shows the FOPDT approximation. It can be indicated that these two result are nearly the same so that the FOPDT seems to be effective in simulate high order system.

From Figure 3 and Figure 7, the steady-state error, %OS, settling time T_s, and the rise time T_r can be classified in to Table 2.

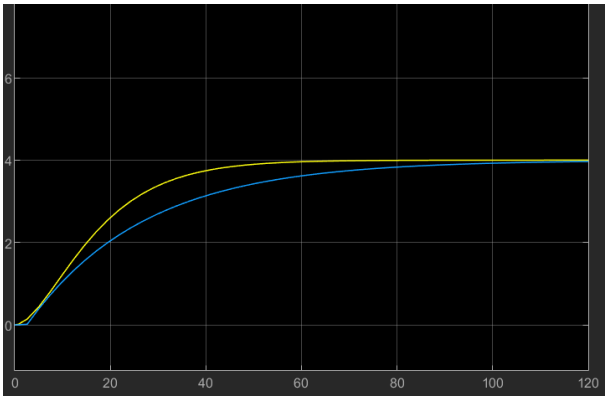

Figure 7: Simulink Result of FOPDT

Table 2: Parameters of FOPDT

Model	OS%	T_r	T_s	steady error
Open-loop	0	30.5	52.8	0
FOPDT	0	53.5	98.5	0

Table 2 displays the comparison between the effect of Open-loop situation and FOPDT. It can be further indicated that the T_r and T_s of FOPDT are large than actual. But in generally, it is suitable for approximation.

3 Part 2

3.1 Method and procedure

To design a P controller, the most important thing to do is to settle the gain of the controller. According to FOPDT model, the recommended value of gain is determined by $\frac{T_p}{t_d}$. From the information of Table 1, the value can be calculated to be 9.65. In this experiment, as we also need to compare the result of different controller parameters, two other P controllers were set with 7.65 and 11.65 to investigate the effect of changing this parameters of the P controller.

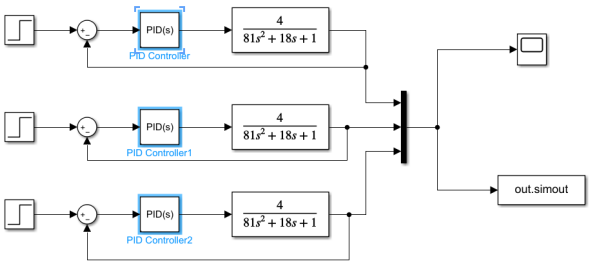

Figure 8: Simulink Model of P Controllers

Table 3 displays the P value of these three controller, while Figure 8 shows the structure of how these three controllers being built. Similar to the above process, an unit-step and the

Table 3: Controller Gain

PID Controller	PID Controller 1	PID Controller 2
9.65	7.65	11.65

result was import in to the P controller before the plant. The result was exports to scope to display.

3.2 Result and Comment

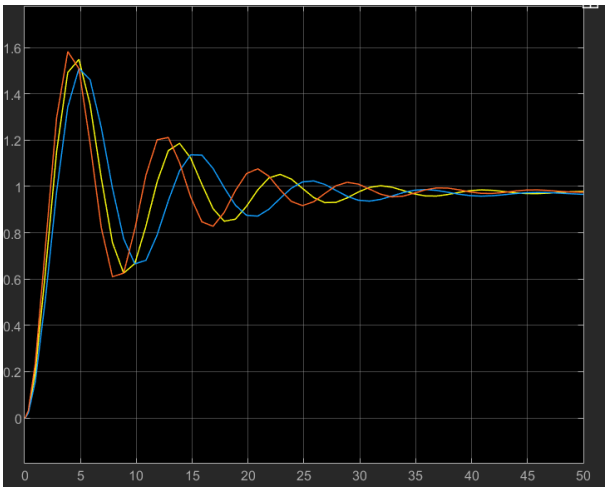

Figure 9: Simulink Result of P Controllers

Figure 9 presents the result from the simuLink model in Figure 8, while the yellow line represents the PID Controller, blue line represents the PID Controller 1, and the red line represents the PID Controller 2. It can be indicated from the figure that the red line acts quick than the other two lines and it achieve the stable condition much earlier than the other two lines. This may indicate that the larger the P gain, the more quick (strong) the controller can adjust the plant. Additionally, it can be noticed that the amplitude of the red line is greater than the other two cases. This may means that the larger the P value, the more shake would be generated before stable.

Table 4: Parameters for different Kp

Gain	OS%	T_r	T_s	steady error
7.65	50%	2.0	25.7	0.032
9.65	54%	1.8	28.3	0.025
11.65	58%	1.6	28.7	0.021

From Table 4, it can be shown that, with the increasing of P parameters, the OS% and T_s increases, while the T_r and steady error decreases. The increase of OS% indicates that the more gain, the more shaking amplitude. The decreasing T_r indicates that the greater the gain, the quick the quicker the system can recover from effect. The increasing T_s means that Greater

gain can bring the system earlier to get into a stable condition. However, the decreasing steady error shows that the final state can be not equal to the ideal one.

4 Part 3

4.1 Method and procedure

From the FOPDT part, it can be conducted that one possible value for the gain K_p and K_I in PI case, while K_p equals $0.9\frac{T_p}{t_d}$ and K_I equals $0.27\frac{T_p}{t_d^2}$. Therefore, the value of K_p and K_I in our case can be calculated to be 8.69 and 1.03.

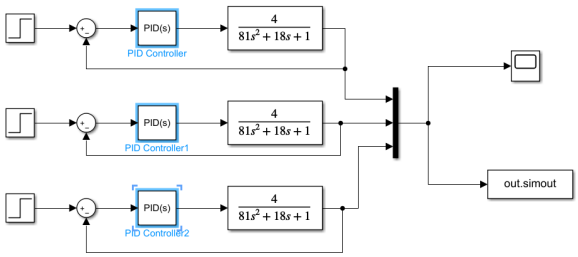

Figure 10: Simulink Model for PI controller

Table 5: K_I with K_p equals 8.69

PID Controller	PID Controller 1	PID Controller 2
0.53	1.03	1.53

Table 6: K_p with K_I equals 1.03

PID Controller	PID Controller 1	PID Controller 2
7.69	8.69	9.69

Figure 10 shows the method one simulink to simulate a PI controller. To investigate the properties of the PI controller, we firstly kept the value of K_p at 8.69 and change the value of K_I as shown in Table 5. Then the value of K_p was kept at 1.03 and the value of K_I varied as shown in Table 6.

4.2 Result and Comment

4.2.1 Change K_I

Figure 11 shows the result of keeping K_p and changing K_I. It can be noticed that the larger the K_I, the stronger shaking would be. Besides, the system with a larger K_I tends to take longer time to make the system back to the stable condition.

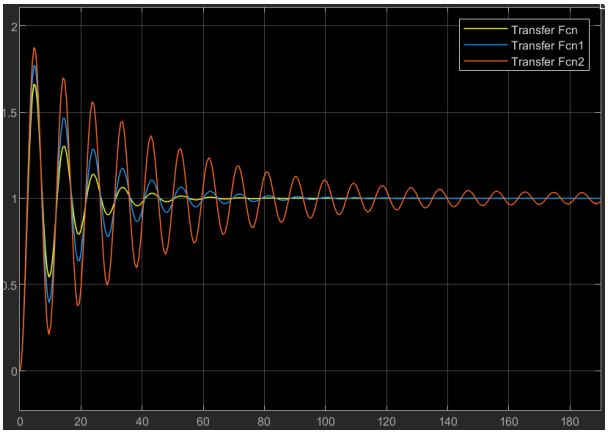

Figure 11: Result from K_p equals 8.69

Table 7: Result with different K_I

K_I	OS%	T_r	T_s	steady error
0.53	65.83%	1.6	68.61	0
1.03	77.68%	1.6	105.9	0
1.53	95.01%	1.6	∞	0

From Table 7, it can be found that with the increasing of K_I, the OS% increases, the T_r remains, T_s increases and the steady errors are all zero. The increasing of OS% indicated a stronger shake for the overall system. However, the steady error was effectively removed by adding the Integral part. The increasing of T_s means that the adding K_I can result in long time to recover to the stable state. If the value of K_I is set for too large, this may result in a always shaking behavior.

4.2.2 Change $\mathbf{K}_p$

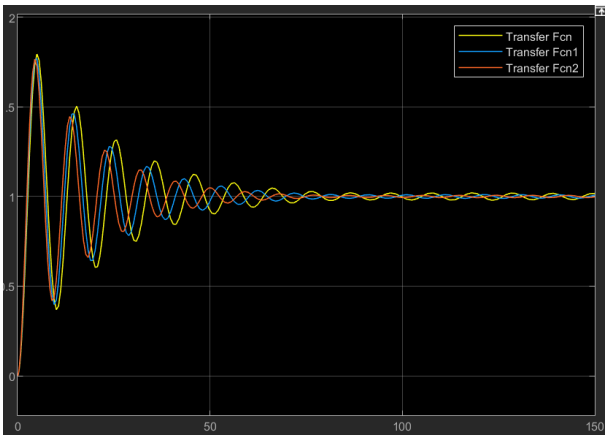

Figure 12: Result from K_I equals 1.03

From Figure 8 and Table 6, it can be inferred that the changing of K_p does not have

Table 8: Result with different K_p

K_p	OS%	T_r	T_s	steady error
7.69	77.7%	1.6	108	0
8.69	77.7%	1.6	70	0
9.69	77.7%	1.6	60	0

significant effect on OS%, T_r and steady error. The only thing it can impact is that with the increasing of K_p, the time to achieve the stable state can be shorten.

Comparing with the P model in Section Part 3, the PI model have higher OS% but with nearly removed steady error.

5 Part 4

5.1 Method and procedure

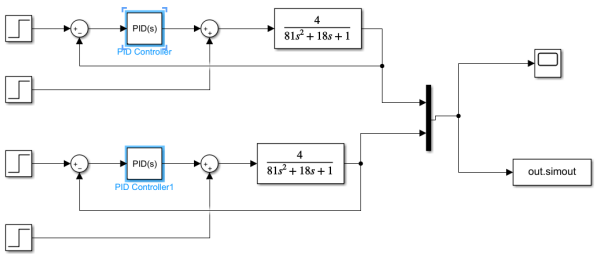

Figure 13: Simulink Model with 5 disturbance at 70s

Figure 13 shows the simulate model of P and PI model with additional disturbance. In this case, for the P model, K_p was set to be 9.65 while the K_p and K_I of PI model was set to be 8.69 and 1.03. After the controller, a disturbance was implemented with a unit-step with amplitude of 5 at 70s. The result was export to the scope for display.

5.2 Result and Comment

Table 9: Disturbance Response of P and PI Model

Model	Max	T_s	steady error
P	2.0	102	0.47
PI	1.7	199	0

Figure 14 and Table 9 shows the result of this part. As it can be inferred from Figure 14, at 70s, a disturbance was implemented and the system need to recover itself through the P and PI controller. It can be found that comparing with P Model, the PI Model has less Max value, nearly zero steady error while the P Model has a shorter recover time. However, the PI

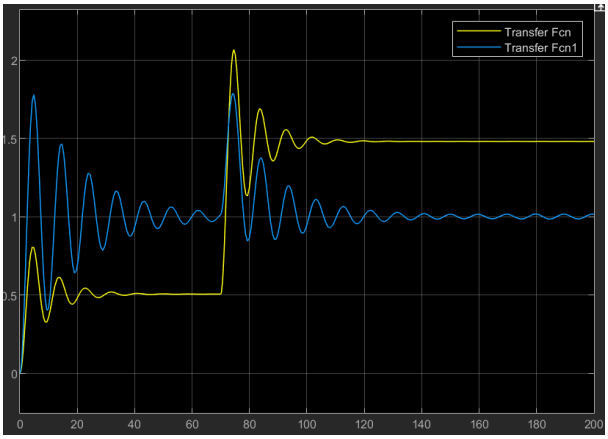

Figure 14: Simulink Result with 5 disturbance at 70s

Model is thought to be more suitable for remove the effect of disturbance because its overall performance is better.

6 Bonus

6.1 Method and procedure

According to the FOPDT case, for a PID controller, the parameters K_p, K_I and K_D can be designed as $1.2\frac{T_p}{t_d}$, $0.6\frac{T_p}{t_d^2}$ and $0.6T_p$ accordingly. In our case, they are 11.52, 2.28 and 14.68.

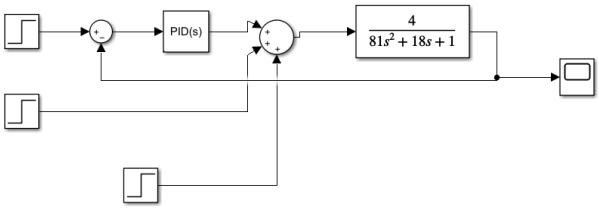

Figure 15: Simulink with 2 disturbance at 70s and 140s

To test the system's ability to against disturbance, 2 unit-step disturbance with amplitude 1 and 5 were implemented at 70s and 140s. The simulink model can be shown in Figure 15.

6.2 Result and Comment

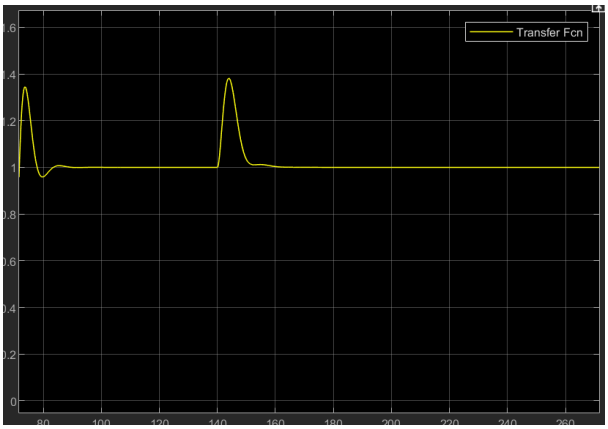

Figure 16: Simulink Result with 2 disturbance at 70s and 140s

From Figure 16, it can be seen that these two disturbance were eliminated successfully. With around 40% OS% and T_s within 10s, this response was expected.

7 Discussion and Conclusion

In this experiment, controller design method has been designed utilizing P model, PI model and PID model. FOPDT method was used to approximate the original response and to find the suitable parameters. It was investigated that comparing with P model, PI model seems to have higher OS% but with nearly removed steady error. PI model shows more expected ability to against disturbance because of less Max value, nearly zero steady error compared with P model. Finally, a PID model was designed to make the anti-disturbance ability of the system to be as high as possible.

It was discovered that the P component can slightly decrease the system stability and generate fast transient response while resulting in higher steady-state error and overshoot. The I component can improve the stability, decrease the steady-state error and overshoot while giving rise to a longer transient response. The D component can control the overshoot to be smaller but will result in low stability, slow transient response and more steady-state error.

References

[1] A. Ataby, "Experiment 81 - design of a feedback control system," https://vital.liv.ac.uk/bbcswebdav/pid-2008634-dt-content-rid-13991637_1/courses/ELEC273-201920/Exp%2081%20%28Feedback%20control%20system%29.pdf, UoL, 2011.

Appendices

A Part 1 Screenshot

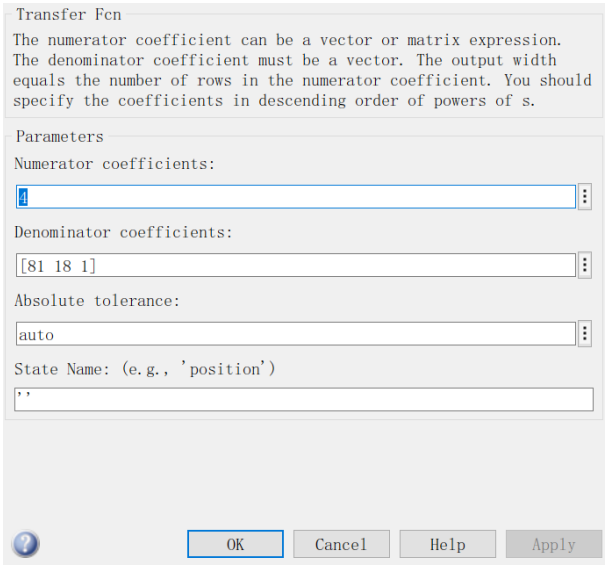

Figure 17: Part 1 trans func

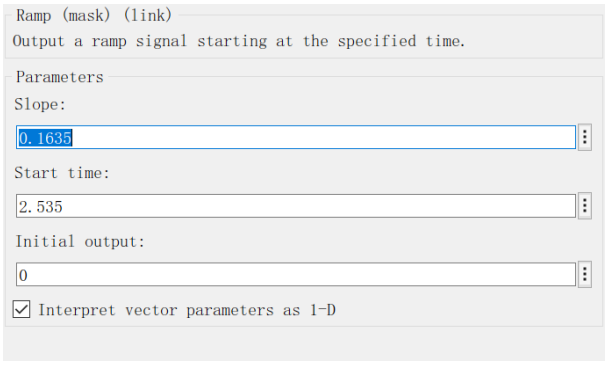

Figure 18: Part 1 tangent line

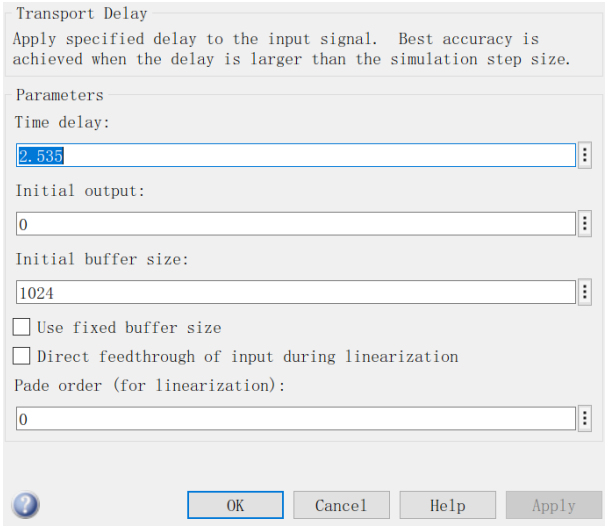

Figure 19: Part 1 FOPDT delay module

B Part 2 Screenshot

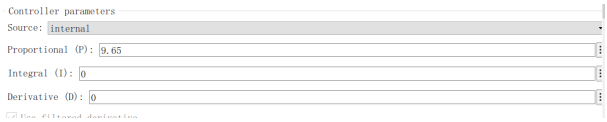

Figure 20: Part 2 P model

C Part 3 Screenshot

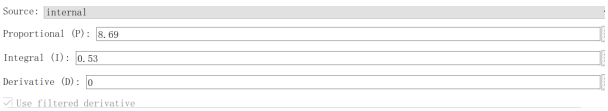

Figure 21: Part 3 PI Model

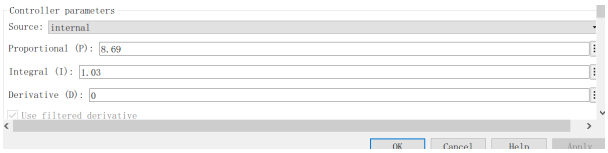

Figure 22: Part 3 PI Model

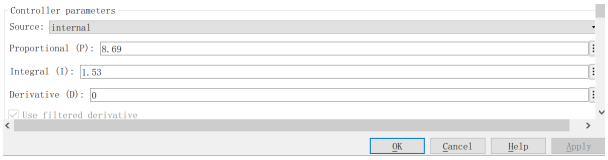

Figure 23: Part 3 PI Model

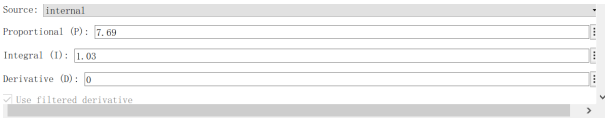

Figure 24: Part 3 PI Model

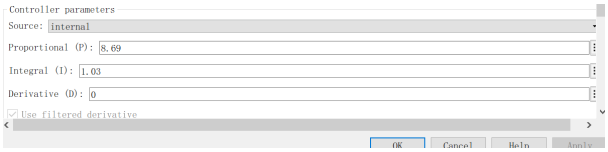

Figure 25: Part 3 PI Model

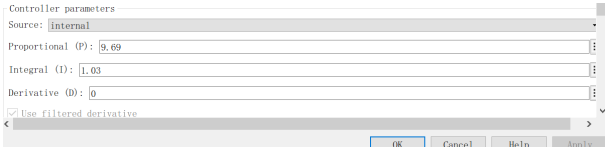

Figure 26: Part 3 PI Model

D Part 4 Screenshot

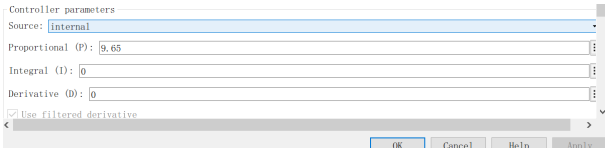

Figure 27: Part 4 PI Model

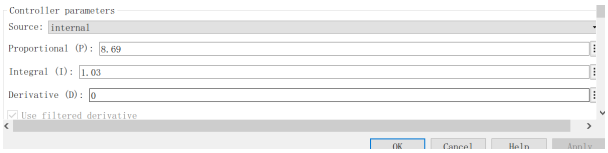

Figure 28: Part 4 PI Model

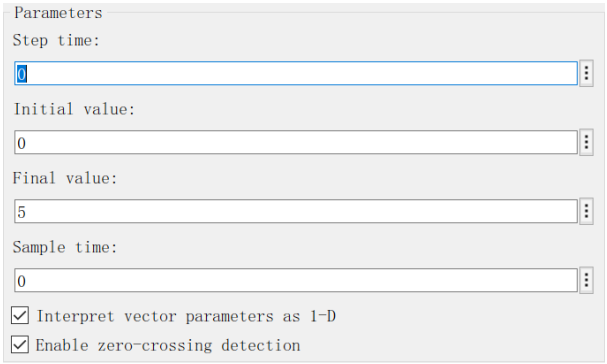

Figure 29: Part 4 disturbance

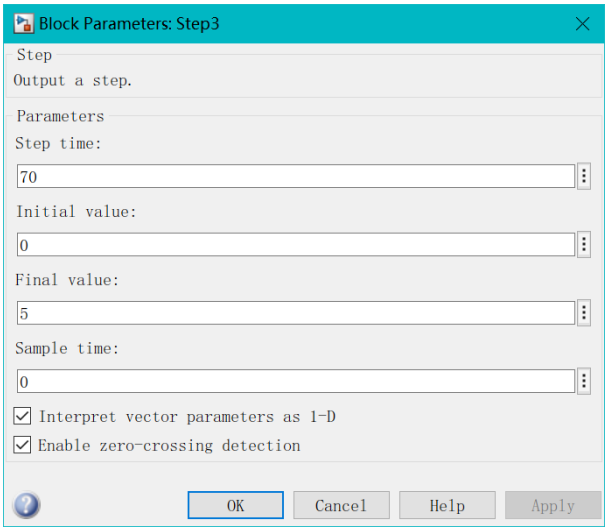

Figure 30: Part 4 disturbance

E Part 5 Screenshot

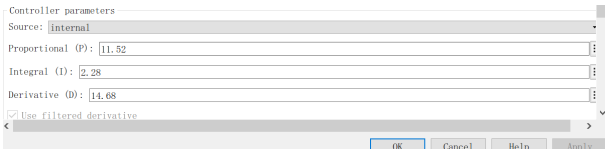

Figure 31: Part 5 PID Model

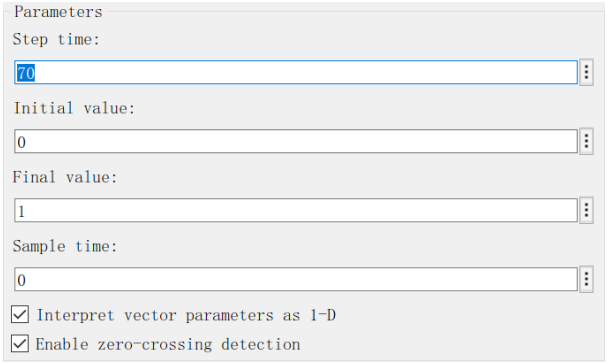

Figure 32: Part 5 disturbance

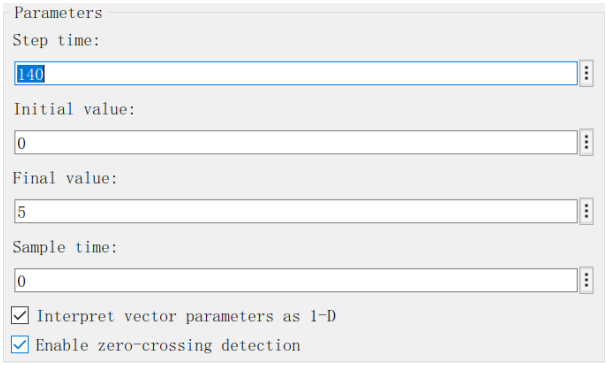

Figure 33: Part 5 disturbance