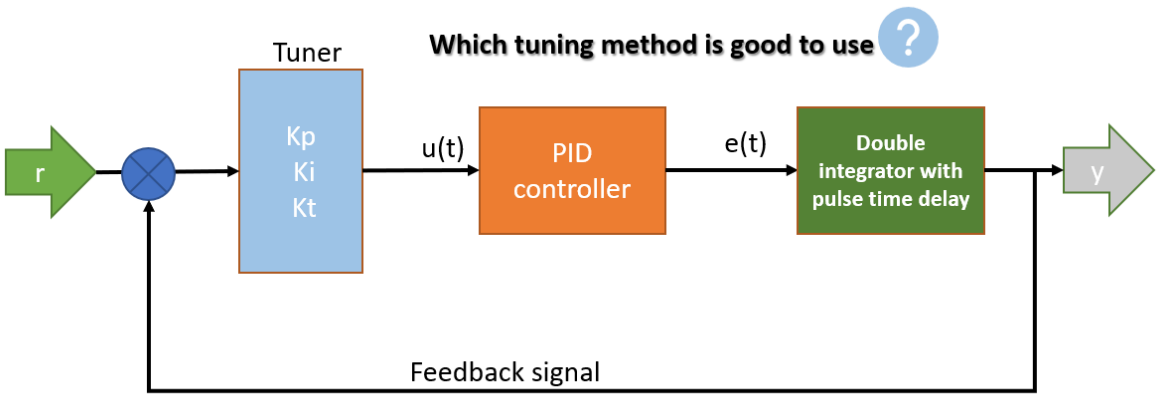
University of South-Eastern Norway

www.usn.no

FMH606 master's Thesis 2023

Industrial IT and Automation

Tuning PID Controllers for Double Integrating Plus Time Delay (DIPTD) processes.



Arash Aminaliabadi

Faculty of Technology, Natural sciences and Maritime Sciences

Campus Porsgrunn

University of South-Eastern Norway

www.usn.no

Course: FMH606 master's thesis, 2023

Title: Tuning PID Controllers for Double Integrating Plus Time Delay (DIPTD) processes

Number of pages: 43 pages

Keywords: PI controller, tuning, integrating system, time delay, maximum time delay error, frequency, PID controller, crucial gain, phase margin, overshot, step response.
analysis

Student: Arash Aminaliabadi

Supervisor: David Di Ruscio

External partner: **Not Available**

Summary:

This report tried to have an introduction on the PID controller and theory beyond this kind of controller, then it is going to discuss some tuning methods which are available in referred literatures in task description for PID controllers in the double integrator with pulse time delay system. Excerpt methods have been discussed in the review section the Modified IMC (Internal Model Control) and DDCD (Di Ruscio and Dalen Controller Design) methods have been chosen to implement on the model of the DIPTD in MATLAB environment. After all, it is concluded that which method can be suggested to use for double integrator with pulse time delay system in desired gain crossover and phase margin. in general, the DDCD method may be a better choice for systems with significant time delays, as it is specifically designed to handle such systems. However, the choice between these methods ultimately depends on the specific requirements and characteristics of the system being controlled.

Preface

A peaceful world and society are one in which individuals and communities live in harmony, free from violence, oppression, and discrimination. It is a world where conflicts are resolved through peaceful means, and where the basic needs of all people are met, including access to education, healthcare, and economic opportunities. Building a peaceful world and society requires a collective effort from individuals, governments, and organizations around the globe. It involves promoting values such as respect, empathy, and kindness, and addressing root causes of violence and injustice, such as poverty, inequality, and prejudice. Only through collaboration and a commitment to peace can we create a world where all individuals can thrive and reach their full potential.

Hope it will be a real matter one day.

Porsgrunn, 15.05.2023

Arash Amin

Contents

1 Introduction 6

1.1 Feedback system 8
1.2 PID Controller 9
1.2.1 Proportional gain 10
1.2.2 Integral gain 10
1.2.3 Derivative gain 10
1.3 PID algorithm 11
1.4 Tuning of the PID controllers 12
1.5 Time-delay 13

2 Literature Review 14

2.1 Tuning PI controller for integrating pulse time delay process. 14
2.1.1 Suggestions 14
2.1.2 Integrating pulse time delay model function process model 15
2.1.3 Tuning of PI controller methods for IPTD systems. 16
2.1.4 Comparison of the results on tuning PI controller for IPTD process 18
2.2 Tuning PID controller for Double integrator pulse time delay 19
2.2.1 Suggestions 19
2.2.2 Double integrating plus time delay (DITD) process model 20
2.2.3 Tuning methods for PD and PID controllers for double integrating plus time delay (DITD) systems 21
2.2.4 Comparison on results 24

3 Simulation and Implementation 26

3.1 Double integrator pulse time delay process model 27
3.2 Implementing double integrator with pulse time delay system impacted by disturbance function. 29
3.3 Implementing double integrator with pulse time delay system with Modified internal model (IMC) tuning method for PID controller. 31
3.4 Implementing double integrator with pulse time delay system with DDCD tuning method for PID controller. 34

4 Conclusion 37

References 38

Appendices 39

Nomenclature

DIPTD	Double integrator pulse time delay
IPTD	Integrator pulse time delay
PID controller	Proportional, Integral, Derivative controller
PI controller	Proportional, Integral controller
PD controller	Proportional, Derivative controller
Modified IMC	Modified internal model control
DDCD	The author’s name Di Ruscio and Dalen Controller Design
ZN	Ziegler-Nichols
ITAE	Integral of Time multiplied by the Absolute Error
IAE	Integral Absolute Error
ISE	Integral of the Squared Error

1 Introduction

Imagine that there is a system that must be under control, it is called plant here, as you can see in below Figure 1, in every plant there is a input signal that is actuating and output signal which is a variable of the system.

Figure 1, The plant is a system that we want to control its behaviors.

In many industries, they call the input and output signals differently; generally speaking, the matter is how to achieve the right output with the input. Exactly here, industries use control engineering to produce the right input signal to get the proper output. Adjusting a system needs to have a feedback signal to compare an input/commanded signal with the controlled variable; by this feedback signal and comparing it with the input/commanded signal, the error term can define in the system. in Figure 2, a basic form of feedback is illustrated, it shows that error signal might be zero when the controlled variable is equal to commanded signal in the system. In this way, an operator can understand that the system is under control.

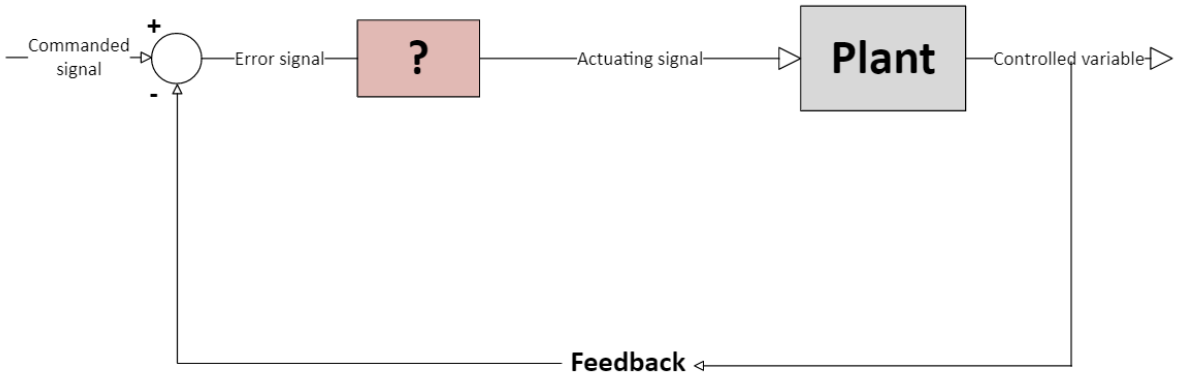


Figure 2, The feedback signal makes the error in controlling the commanded signal.

Reaching the zero error is the thing that is important in the system; this process (getting zero error) needs time to make the error equal to zero, so the question is how it is possible to convert the error signal to a proper actuating signal in terms of time. The answer to this question is that a controller can make a suitable actuating signal for the plant to get a controlled signal in the output. It means that it is possible to drive the error to zero over time by using the proper controller in the system.

Converting the error signal to the actuating signal for the plant needs to consider some actions that are happening during the controlling process; imagine a system with a setpoint in a certain amount to achieve this set point, there is a need to control signal by amplifying the input signal in the meantime by multiplying to a gain number, to achieve the zero error but when it will happen the error would be zero. Still, the commanded signal will get zero, and the output will oscillate like an on/off control system. This control system would be unstable. This part of the controller, called proportional, has a gain to amplify the signal. To improve the system's stability and keep the steady state error, the old information would help control the output signal to keep it closer to the actual output; in this part, an integrator will add to the control system to use the old information to keep the steady-state error, this integrator will keep the total of the inputs overtime then if you there is an increasing at the input there will be increasing in output also. On the other hand, as long as the error is zero, the proportional controller will not gain something in the system; these two controllers help each other to keep the system in a steady-state condition, over the time the integrator will increase the output when a small amount of error has appeared. Then the interesting point in output will be achieved after some oscillation because there will be a negative error on the system also; it might happen many times to get the ideal output in the plant by adding a derivative part to the controller system, the future error would be predicted because it will count the terms of error and checking that how the system is close to the goal in output.

Using the present error, the past error and an anticipation of the future error controller would calculate the suitable actuator command to achieve the ideal output command. The PID controller will be placed in the question block in the figure below.

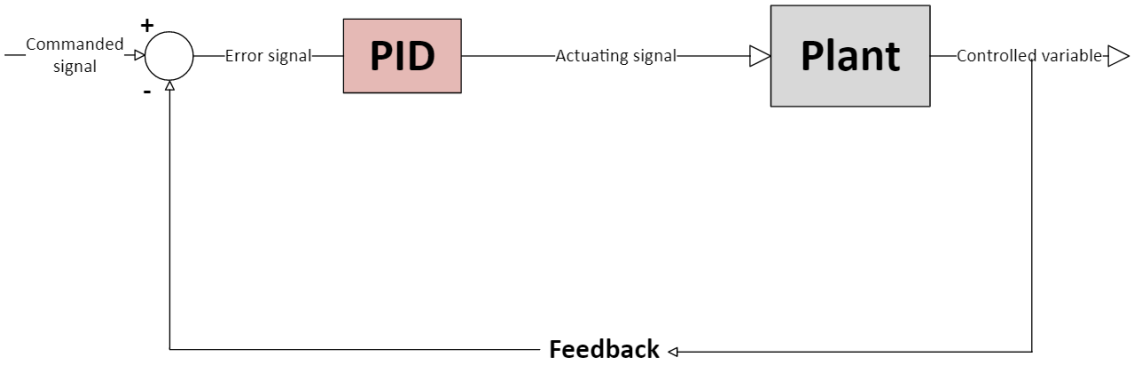


Figure 3, PID controller includes the proportional, integrator and derivative parts.

In this report, the theory of modelling and calculation of the controller will be discussed and reviewed for digging into the tuning of the PID controller with a double integrating pulse time delay system. The primary purpose is to compare a plant's tunning method and analysis the outputs.[2]

1.1 Feedback system

Industries are attempting to achieve the desired output with high accuracy; for this reason, the feedback system will add to the control loop to enhance the error as much as possible.

In figure 4, there is a block called feedback system, this system usually includes some equipment to acquire the value of output and send it to the controller unit.

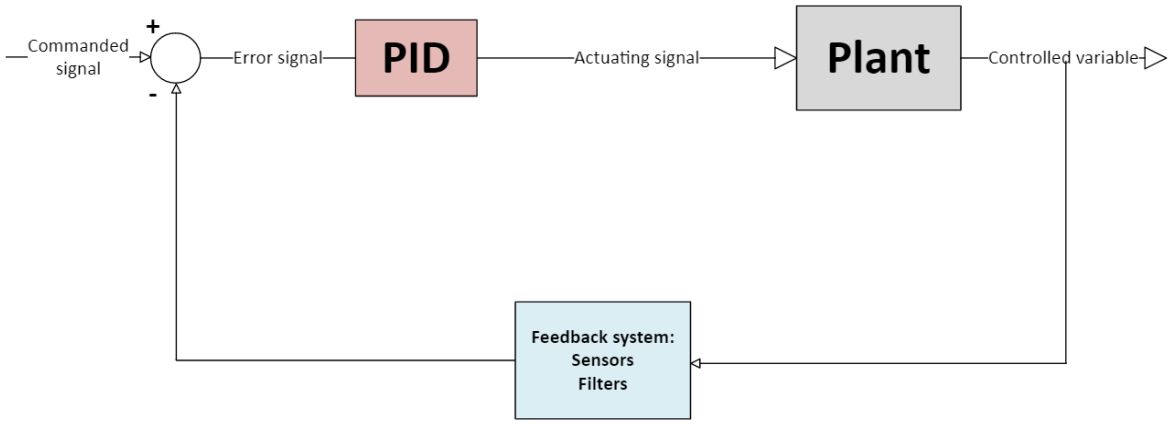


Figure 4, The control loop is completed by adding the feedback system.

The controller unit will process the feedback and prepare the suitable command for the actuator.to reach the target output, there is a need to run an automatic control loop. By implementing the feedback system and proper tuning, it would be reachable for the process. The automatic control of a plant needs to have a model of the control system in the next step; the mathematical model of the PID controller defines simulate by coding on any environment.

In the control loop, some essential parts, such as the measurement filter, will be discussed in the future. As it is necessary for the sensor to acquis data from the analogue signal and turn it into the digital signal, there would be a measurement system in the feedback loop; the most crucial part of the measurement system is a filter. The duty of the filter is to remove or thin random measurement noise from the measurement signal; this measurement signal will go to the controller to calculate the output according to the setpoint; in this way, there will be an automatic computer-based controller in the process.[3]

Simple notation for the measurement signal of discrete time t_k is defined below, which is filtered by the discrete time.

$$y_{mf} = y_{mf}(t_k)$$

Then the error signal e_k will be like Equation 1, where the feedback loop calculates the error signal as the difference between the setpoint value $y_{sp,k}$ and $y_{mf,k}$.

$$e_k = y_{sp,k} - y_{mf,k} \qquad (1)$$

This error signal will be processed by PID discrete time controller then controller provides the actuator signal for plant to reach the desired output.

1.2 PID Controller

As it is discussed above the PID controller includes three terms which noted here like u_p for the proportional part, u_d assigned to the derivative part and the integrator part signed as u_i.

By definition of the above terms, there would be a model of the PID controller, which is ready to code on computer programs. The model includes all terms and a parameter for manual controlling u_{man}, this parameter is the nominal value of the control variable [3].

This model is presented for a continuous-time PID controller, which is still based on the discrete-time PID controller. This model is presented in the Automatic Control book by Finn Akre Haugen [3].

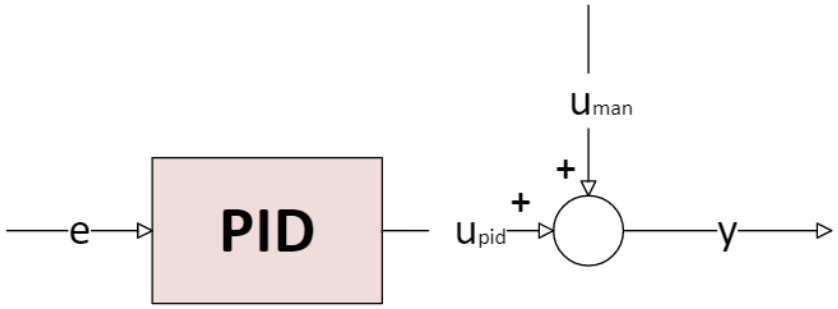


Figure 5, A standard PID controller with manual adjustment parameter.

Imagine a standard PID control system like the figure 5, the model of this controller consists of all terms, which Equation (2) shows.

$$y_o = u_p + u_i + u_d + u_{man} \qquad (2)$$

Inside the PID controller box, there are similar terms; these terms are driving with the continuous-time which means that if the other types of controllers are desired, by putting each term equal to Zero, it would be achievable; for example, if the operator wants to make a PI controller, they can put the derivative time to infinity time then they will have a PI controller, an explanation of the terms would be three terms that will be defined in this notation Equation (3).

$$y_o = K_p e + \frac{K_p}{T_i}\int_0^t e\, d\tau + K_p T_d \dot{e} + u_{man} \qquad (3)$$

Here the e is defined as equation (4) and it is controlling error.

$$e = y_{sp} - y_{mf} \qquad (4)$$

There are three types of gain here, more than manual gain, which will be discussed in the next step. But the focus for all of them is on the essential parameters for tuning the PID controllers. Where the operator/control engineer can set up the tuning based on these parameters.

1.2.1 Proportional gain

K_p is the proportional gain that is directly multiplied by the controller input, as explained before. Most of the PID controllers in the market use the proportional band P_B; this parameter is given as a stead proportional gain by:

$$P_B = \frac{100\%}{K_p} \qquad (5)$$

Where the explanation of the P_B meaning actually is a range of change in proportional gain K_p.

1.2.2 Integral gain

The integral time is T_i , which counts by time units, e.g. minutes, seconds. This time is the duration of the repeating of the proportional gain K_p in the control system, the alternatively the integral gain K_i would be:

$$K_i = \frac{K_p}{T_i} \qquad (6)$$

1.2.3 Derivative gain

By multiplying the T_d with the proportional gain K_p there will be the derivative gain like Equation (7):

$$K_d = K_p T_d \qquad (7)$$

1.3 PID algorithm

Based on continuous time PID controller, it will be possible to develop a discrete time PID algorithm ready for programming here [3], in this report there is no discussions about the definition of the mathematical model so just summarize of the model would be under consideration for rest of the work. As mentioned above the summary of the PID algorithm will be in these parameters below.

- Control error:

$$e_k = y_{sp,k} - y_{mf,k} \quad (8)$$

- Manual control signal:

$$u_{man,k} = constant \quad (9)$$

- P term:

$$u_{p,k} = K_p e_k \quad (10)$$

- I term:
  This term is given by Euler Backward method in Equation (11).

$$u_{i,k} = u_{i,k-1} + \frac{K_p T_s}{T_i} e_k \quad (11)$$

- D term:
  This term is given by Euler Backward method in Equation (12).

$$u_{d,k} = K_p T_d \frac{e_k - e_{k-1}}{T_s} \quad (12)$$

- Total control signal:

$$u_k = u_{man} + u_{p,k} + u_{i,k} + u_{d,k} \quad (13)$$

For these algorithms in Equation 8 and Equation 13, there are idealized PID algorithms. To have more real algorithm, practical modifications like below are needed.

- Low pass filtering of D term
- Integral anti windup
- Reducing P kick and D kick
- Bump less transfer between manual and automatic modes.

These adjustments are needed when the PID controller is going to implement on the computer to control the plant's features. But in this report the tunning of PID controller is more focused to get the target point in the output of the controller, so in next step this would be discussed in more details.

1.4 Tuning of the PID controllers

Some parameters of the PID controllers such as K_p, T_i and T_d are important to have a suitable value as a setting to tune the controller, so the specification of the control system could be passed with these parameters adjustment. There are two items in the specification of the control system like:

- Stability, which is defined in the ZN method.
- Speed

But before attempting to tune a controller. It is important to set the PID controller to work reverse or in direct action. Therefore, these two will be described briefly here.

It is vital for controlling a plant that setup the PID controller for having the reverse action or direct action means that if the controller give the command in a opposite direction of control signal, the control system is going to be unstable, unsafe, and defective at all. It means when there is negative K_p or positive K_p. After checking the behavior of the control system, the result will show that the system is working in a proper way there is need to have a tuning or adjustment on the parameters settings. There are famous methods for tuning of PID controllers and their settings here in the below list [3]:

- **The Ziegler-Nichols closed loop.**

  This setting is related to tune the PID controller that is presented in the reference, P_u is always a constant number as an ultimate period [3]

  Proportional Gain: $0.6K_{p,u}$

  Integral Time: $\frac{P_u}{2}$

  Derivative Time: $\frac{T_u}{8} = \frac{T_i}{4}$

- **Relaxed Ziegler-Nichols**
  This setting actually is resulting of combination of ZN and Skogestad methods and it would apply on PI controllers.

  Proportional gain: $0.25K_{p,u}$

  Integral Time: $\frac{P_u}{1.2}$

- **Skogestad Controller tuning method**
  This method is based on transfer function of the control process parameters, it will discuss in the next chapter through the literature reviews. In a short expression,

regarding the formulation in Equation (14), the transfer function is structed the principles of Skogestad method for tuning of the PID controller.
Regarding the below equation the Time-constant T_c must modify by user and the τ is the time delay that is given on the process model.

$$\mathrm{T(s)} = \frac{1}{T_c s+1} e^{-\tau s} \qquad (14)$$

1.5 Time-delay

Signal flow in several system has a time delay or dead time, to have a simple example imagine the conveyor belt in below figure 6, here the relation between input variable $K_{p,u}$ and output variable is the transportation time delay with the time-delay as τ in the system.

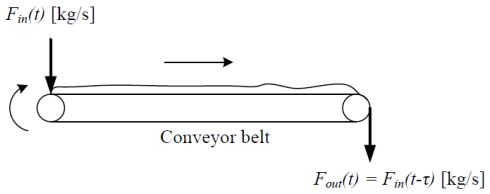


Figure 6, Time-delay on the blet [3].

Equation 15 indicates the relation between input and output on this belt.

$$F_{out}(t) = F_{in}(t-\tau) \qquad (15)$$

In this expression the time delay will be presented by transfer function in Equation (16).

$$H(s) = e^{-\tau s} \qquad (16)$$

This type of delay is a 1st order transfer function modeling, there are different types of delay at 2nd order in the process that will be discussed further in this report and tuning method of the PID controllers for integrating and double integrating pulse time delay systems in practical tuning examples.

2 Literature Review

In this chapter, there is a brief review of science papers which are included some PID controller tuning methods settings, they worked on analytical and results from tuning methods on integrator and double integrator pulse time delay systems. Then they covered a lot of information related to the models based on time delays.

2.1 Tuning PI controller for integrating pulse time delay process.

There is a paper "On Tuning PI Controllers for Integrating Plus Time Delay Systems" by David Di Ruscio, 2010 [4] that discusses the process of tuning PI controllers for integrating plus time delay systems in industrial control applications. Integrating plus time delay systems are common in industrial processes and can be challenging to control due to their tendency towards instability and oscillation. The use of a PI controller is a common method to improve control performance by adjusting the process output based on the difference between the setpoint and the actual output. The article emphasizes that proper tuning of the PI controller is critical to achieving stable and responsive control. The tuning process involves selecting appropriate tuning parameters and adjusting them to optimize control performance. The article outlines a step-by-step approach to tuning the PI controller for integrating plus time delay systems, which includes identifying the system's characteristics, selecting a suitable tuning method, and adjusting the tuning parameters based on the desired performance specifications. The tuning process also involves evaluating the system's response using simulations and adjusting the tuning parameters accordingly. The article provides several examples and simulations to illustrate the tuning process and its impact on control performance. It also discusses the limitations of the tuning process and the need for ongoing monitoring and adjustment of the control system.

2.1.1 Suggestions

The article does not identify a specific tuning method as the best for all integrating plus time delay systems. Instead, the author suggests that different tuning methods may be appropriate depending on the characteristics of the specific system being controlled and the desired performance specifications.
The article provides several examples of tuning methods and compares their performance in simulations. However, the results indicate that the best tuning method depends on the specific system being controlled and the performance specifications. The article emphasizes that the tuning process should be iterative, with adjustments made based on the system's response to the controller, until the desired performance is achieved.

2.1.2 Integrating pulse time delay model function process model

Referred integrating plus time delay (IPTD) process model in this paper, which can be described by the following equation (17):

$$G(s) = \frac{K}{(1+T_{ds})\mathrm{e}^{-Ls}} \qquad (17)$$

where $G(s)$ is the transfer function of the process, K is the process gain, T_d is the process time constant, L is the process dead time, and s is the Laplace transform variable.

This model is used in the paper to illustrate the tuning process for the PI controller. The author explains how to determine the values of K, T_d, and L for a given process and how to use these values to select appropriate tuning parameters for the PI controller.

The IPTD process model is a common mathematical representation of industrial processes that have both integrating (or integral) and time delay characteristics. Integrating processes have a cumulative effect on the process output, meaning that any deviation from the setpoint accumulates over time. Time delay refers to the time lag between a change in the process input and the resulting change in the process output.

The IPTD process model equation includes three components: the gain, the time constant, and the dead time.

- The gain, K, represents the steady-state output response of the process to a change in the input. A higher gain indicates a more responsive process.
- The time constant, T_d, represents the time it takes for the process output to reach 63.2% of its steady-state value in response to a step change in the input. A longer time constant indicates a slower process.
- The dead time, L, represents the time lag between a change in the input and the corresponding change in the output. Dead time can arise from a variety of sources, such as physical transport delays or measurement delays.

The transfer function of the IPTD process model, $G(s)$, is a mathematical representation of the relationship between the process input and output in the frequency domain. The Laplace transform variable, s, allows for analysis of the system's response to different inputs over time.

Above, the IPTD process model is used to illustrate the tuning process for the PI controller. By understanding the characteristics of the process and its transfer function, the author provides a practical guide for selecting appropriate tuning parameters to achieve stable and responsive control.

2.1.3 Tuning of PI controller methods for IPTD systems.

The article discusses several tuning methods for PI controllers in integrating plus time delay (IPTD) processes. The tuning methods include:

1- Ziegler-Nichols method: This method involves step testing the process to determine the ultimate gain and ultimate period of the system. Based on these values, the proportional gain and integral time constant of the PI controller can be calculated.

2- Cohen-Coon method: This method also involves step testing the process, but it uses a different approach to calculate the tuning parameters for the PI controller. This method is known for its simplicity and ease of implementation.

3- Internal Model Control (IMC) method: This method uses a model of the process dynamics to calculate the tuning parameters for the PI controller. The model incorporates the effects of the process dead time, which can improve control performance in systems with significant dead time.

4- Direct Synthesis method: This method involves designing the PI controller using a model of the process transfer function in the frequency domain. This approach can result in better control performance but requires more advanced mathematical knowledge and tools.

The author provides examples and simulations to compare the performance of these tuning methods for different IPTD processes. The author emphasizes that the most appropriate tuning method will depend on the specific characteristics of the process and the desired control performance.

In summary, the article discusses several tuning methods for PI controllers in IPTD processes, including Ziegler-Nichols, Cohen-Coon, IMC, and Direct Synthesis. The article provides guidance on how to select an appropriate tuning method based on the specific system being controlled and the desired control performance.

2.1.3.1 Suggested setting for each tuning method

Here are the equations for each tuning method discussed by author:

1- Ziegler-Nichols's method:

The tuning parameters for the PI controller using the Ziegler-Nichols method are calculated as follows:

Proportional gain:

$$K_p = 0.9\ \frac{K_u}{L} \qquad (18)$$

Integral time constant:

$$T_d = 0.3\ L \qquad (19)$$

where K_u is the ultimate gain of the system and L is the dead time of the system. The ultimate gain and dead time can be determined by step testing the system.

2- Cohen-Coon method:

The tuning parameters for the PI controller using the Cohen-Coon method are calculated as follows:

Proportional gain:

$$T_d = \frac{(1.35L)}{(K\left(1+0.35\frac{T_d}{L}\right))} \qquad (20)$$

Integral time constant:

$$T_i = \frac{(2.5L)(1+0.35\frac{T_d}{L})}{(K\left(1+0.35\frac{T_d}{L}\right))} \qquad (21)$$

where K, T_d, and L are the gain, time constant, and dead time of the system, respectively. These values can be determined by fitting the IPTD model to the process data.

3- Internal Model Control (IMC) method:

The tuning parameters for the PI controller using the IMC method are calculated as follows:

Proportional gain:

$$T_d = \frac{(\frac{T_d}{K})(1+0.5\frac{T_d}{T_i})}{(1+\frac{T_d}{T_i})} \qquad (22)$$

Integral time constant:

$$T_i = \frac{T_d(1+0.5\frac{T_d}{T_i})}{2(1+\frac{T_d}{T_i})} \quad (23)$$

where K, T_d, and T_i are the gain, dead time, and integral time constant of the process, respectively. The IMC method uses a model of the process dynamics to calculate these values.

4- Direct Synthesis method:

The tuning parameters for the PI controller using the Direct Synthesis method are calculated based on the desired closed-loop transfer function of the system. The transfer function can be designed based on the process model and the desired closed-loop performance specifications.

The article does not provide a specific equation for the Direct Synthesis method, as it is a more advanced tuning method that requires knowledge of control system design and advanced mathematics.

These equations provide a starting point for selecting appropriate tuning parameters for the PI controller in IPTD processes. The author notes that the tuning process may require iterative adjustments based on the system's response to achieve the desired control performance.

2.1.4 Comparison of the results on tuning PI controller for IPTD process

The article provides simulations to compare the performance of different tuning methods for PI controllers in IPTD processes. The simulations evaluate the control performance based on several metrics, including the Integral of Time multiplied by the Absolute Error (ITAE), the Settling Time (T_s), and the Percent Overshoot (%OS).

The results of the simulations show that the performance of each tuning method varies depending on the specific IPTD process being controlled. For example, the Ziegler-Nichols method performed well in systems with smaller dead times but showed poor performance in systems with longer dead times. The Cohen-Coon method performed well in systems with longer dead times but showed poor performance in systems with smaller dead times.

The IMC and Direct Synthesis methods showed better overall performance across a wider range of systems. The article also provides an evaluation of the simulation results using a performance index based on the ITAE and the settling time. This performance index allows for a quantitative comparison of the different tuning methods. The results of the evaluation show that the IMC method and Direct Synthesis method generally outperform the Ziegler-Nichols and Cohen-Coon methods in terms of the performance index. However, the author notes that the performance index is just one metric and that other metrics, such as the %OS, may be more important for certain applications.

Overall, the simulations and evaluations in the experiments suggest that the most appropriate tuning method for a given IPTD process will depend on the specific characteristics of the system and the desired control performance. The article emphasizes the importance of an iterative tuning process that involves adjusting the controller based on the system's response until the desired performance is achieved.

2.2 Tuning PID controller for Double integrator pulse time delay

The paper "Tuning PD and PID Controllers for Double Integrating Plus Time Delay Systems" by David Di Ruscio and Christer Dalen, published in 2017 [5] addresses the problem of tuning PD and PID controllers for double integrating plus time delay (DITD) systems, which are commonly encountered in process control applications. The authors provide a comprehensive review of the existing literature on tuning methods for PD and PID controllers in DITD systems, including both analytical and empirical methods.
The authors note that many of the existing methods have limitations, such as being computationally expensive, requiring an exact mathematical model of the system, or not being able to handle model uncertainties. They propose a new method for tuning PD and PID controllers based on a simplified transfer function model of the DITD system.
The proposed method involves selecting a suitable set of tuning parameters for the PD or PID controller based on the desired closed-loop performance specifications, such as settling time, overshoot, and steady-state error. The authors demonstrate the effectiveness of their method through simulation examples and compare it to other well-known tuning methods.
Overall, the article provides a useful contribution to the field of control engineering, particularly for the tuning of controllers for DITD systems. The authors review the limitations of the existing methods and propose a new method that is simple, effective, and robust to model uncertainties and external disturbances. The article is relevant to researchers and practitioners working in the area of process control and provides a starting point for future research in this area.

2.2.1 Suggestions

Regarding the best method author design anew method in this article, According to the results reported in the article "Tuning PD and PID Controllers for Double Integrating Plus Time Delay Systems" by Di Ruscio and Dalen[5], the DDCD method performed the best among the tuning methods considered. The authors compared the performance of the DDCD method with several other tuning methods, including the Ziegler-Nichols (ZN) method, the Cohen-Coon (CC) method, the internal model control (IMC) method, and the modified IMC (MIMC) method. The comparison was based on closed-loop system performance metrics

such as the integral of the squared error (ISE), the integral of the absolute error (IAE), and the integral of the time-weighted absolute error (ITAE). The results showed that the DDCD method provided the best overall closed-loop performance, with the lowest ISE, IAE, and ITAE values among all the methods tested. The DDCD method also provided good robustness to changes in the process dynamics, such as changes in the process gain or time delay.
However, it should be noted that the best tuning method may depend on the specific characteristics of the process being controlled, and the performance of the tuning methods may vary in different applications.

2.2.2 Double integrating plus time delay (DITD) process model

The double integrating plus time delay (DITD) process model equation presented in the article "Tuning PD and PID Controllers for Double Integrating Plus Time Delay Systems" by David Di Ruscio and Christer Dalen [5] is:

$$Gp(s) = \frac{K_p}{s^2(1+T_d s)} \qquad (24)$$

Where $G_p(s)$ is the Laplace transform of the process output to the process input, K_p is the process gain, T_d is the process time delay, and s is the Laplace variable.

This model equation represents a second-order process with two integrators and a time delay. The double integration in the process model arises from the presence of two poles at the origin (s=0). The time delay is represented by the term $(1 + T_d)$ in the denominator, where T_d is the process time delay.

The process gain, K_p, represents the steady-state gain of the process and can be determined experimentally or through system identification methods. The time delay, T_d, represents the amount of time it takes for the process output to respond to a change in the process input.

The DITD process model is commonly encountered in process control applications, and its control presents unique challenges due to the presence of the double integrators and time delay. Effective tuning of PD and PID controllers for this type of process is essential for achieving desired closed-loop performance specifications, such as fast settling time, minimal overshoot, and low steady-state error.

2.2.3 Tuning methods for PD and PID controllers for double integrating plus time delay (DITD) systems

The author proposes a new tuning method for PD and PID controllers for double integrating plus time delay (DITD) systems, based on a simplified transfer function model. The authors also provide a literature review of other tuning methods for DITD systems.
In the literature review, the authors summarize several existing methods, including:

- Ziegler-Nichols method: A classic empirical method that involves finding the critical gain and period of the system and using these values to determine the controller gains.
- Cohen-Coon method: Another empirical method that involves finding the process time constant and the time delay and using these values to determine the controller gains.
- Internal model control (IMC) method: An analytical method that involves modeling the process dynamics and using this model to design the controller.
- Smith predictor method: A method that involves using a model of the time delay to design a controller that compensates for the delay.
- Pole placement method: An analytical method that involves placing the closed-loop poles of the system at desired locations to achieve a desired response.

The authors note that these methods have limitations, such as being computationally expensive, requiring an exact mathematical model of the system, or not being able to handle model uncertainties. They propose a new method based on a simplified transfer function model that addresses these limitations and is robust to model uncertainties and external disturbances.
The proposed method involves selecting a suitable set of tuning parameters for the PD or PID controller based on the desired closed-loop performance specifications, such as settling time, overshoot, and steady-state error. The authors demonstrate the effectiveness of their method through simulation examples and compare it to other well-known tuning methods.

2.2.3.1 Tuning setting for each method.

The article provides a literature review of other tuning methods for DITD systems, including the Ziegler-Nichols, Cohen-Coon, Internal Model Control (IMC), Smith Predictor, and Pole Placement methods. Here are the equations for each of these methods:

- Ziegler-Nichols method:

The ultimate gain, Ku, and ultimate period, Pu, are determined experimentally by increasing the controller gain until the system oscillates at a sustained amplitude and period. The controller gains are then calculated as:

$$K_p = 0.6K_u \quad (25)$$

$$T_d = 0.5P_u \quad (26)$$

- Cohen-Coon method:

The process time constant, T_p, and the time delay, T_d, are determined experimentally, and the controller gains are then calculated as:

$$K_p = \frac{1.35T_p}{K_p + T_d} \quad (27)$$

$$T_d = 0.35T_p + T_d \quad (28)$$

- Internal Model Control (IMC) method:

The IMC tuning method involves designing a controller based on a mathematical model of the process dynamics. The controller gains are determined as:

$$K_p = \frac{1}{K_p}\sqrt{\frac{T_d}{T_p}} \quad (29)$$

$$T_d = T_d \quad (30)$$

- Smith Predictor method:

The Smith Predictor method involves designing a controller that compensates for the time delay in the process. The controller gains are determined as:

$$K_p = K_p \quad (31)$$

$$T_d = \frac{T_d}{1-\alpha} \quad (32)$$

Where α is a parameter that depends on the process time constant and the time delay.

- Pole placement method:

The pole placement method involves selecting the closed-loop pole locations to achieve a desired response. The controller gains are then calculated based on the pole locations.

- There is a new method of tuning PID controller for DIPTD systems in the article that is presented. The model named like the author's name, The DDCD (Di Ruscio and Dalen Controller Design) method involves selecting the controller gains based on a simplified transfer function model. The method provides a set of guidelines for selecting the controller gains, rather than explicit equations.

Here are the steps involved in the DDCD method:

- Determine the process gain, K_p, and time delay, T_d, through experimental identification or system analysis.
- Calculate the critical gain, K_c, using the following equation (19):

$$K_c = \frac{4T_d}{K_p\pi} \qquad (33)$$

- Select the controller gains based on the desired closed-loop damping ratio, ζ, and natural frequency, ωn. These values can be chosen based on the desired closed-loop performance specifications.

The controller gains, K_p and K_d, can be calculated as follows:

$$K_p = \frac{2\zeta\omega n K_c}{K_p\sqrt{1-\zeta^2}} \qquad (34)$$

$$K_d = \frac{2\omega n^2 K_c}{K_p} \qquad (35)$$

The DDCD method also provides a set of recommended settings for the closed-loop damping ratio and natural frequency, based on the process time constant, T_p:

For T_p < 0.1 T_d, use ζ = 0.4 and ωn = $\frac{1.2}{T_d}$

For T_p > 10 T_d, use ζ = 0.7 and ωn = $\frac{1.2}{T_p}$

For 0.1 $T_d \leq T_p \leq$ 10 T_d, use ζ = 0.5 and ωn = $\frac{1.2}{T_d}$

It should be noted that the DDCD method is intended for tuning PD and PID controllers for double integrating plus time delay (DITD) systems with a dominant double integrator response. The method is not suitable for other types of processes or systems.

2.2.4 Comparison on results

The authors compared the performance of several tuning methods for double integrating plus time delay (DITD) systems, including the Ziegler-Nichols (ZN) method, the Cohen-Coon (CC) method, the internal model control (IMC) method, the modified IMC (MIMC) method, and the DDCD method.

The comparison was based on closed-loop system performance metrics such as the integral of the squared error (ISE), the integral of the absolute error (IAE), and the integral of the time-weighted absolute error (ITAE).

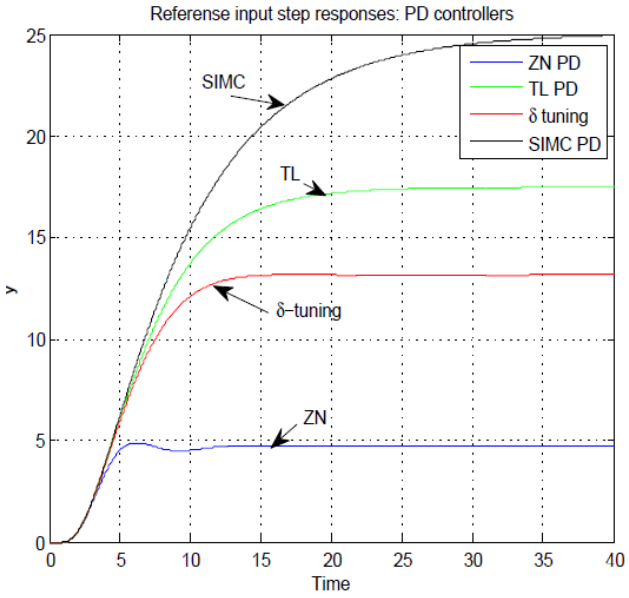


Figure 7, Reference input unit [5].

Figure 7 in the article shows the closed-loop step responses for the different tuning methods, using a set of common tuning parameters. The results show that the DDCD method provided the fastest rise time and settling time, with minimal overshoot and steady-state error. The ZN

and CC methods had similar performance, with moderate overshot and settling time. The IMC and MIMC methods had slower response times and larger overshot.

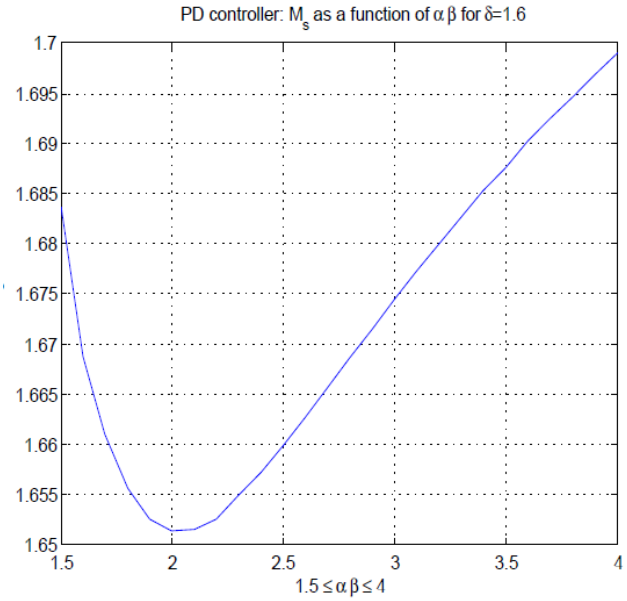


Figure 8, Picture of sensitivity index as function of the tuning parameter.[5]

Figure 8 in the article illustrates the closed-loop frequency response for the different tuning methods, using the same set of common tuning parameters. The results show that the DDCD method provided the most stable closed-loop response, with a larger phase margin and smaller gain margin compared to the other methods. The ZN and CC methods had smaller phase margins and larger gain margins, indicating lower stability margins. The IMC and MIMC methods had similar stability margins, but with larger overshot and slower response times.

Overall, the results of the evaluation based on the figures suggest that the DDCD method provided the best closed-loop performance and stability among the tuning methods considered in the article. However, it should be noted that the specific performance of the tuning methods may depend on the specific characteristics of the process being controlled, and the results may vary in different applications.

3 Simulation and Implementation

Regarding the task description on this research topic, to compare the different tuning methods for double integrating with plus time delay (DITD) systems using simulation experiments in MATLAB, we can generate random state-space models (SSMs) using the "rss" or "drss" functions in MATLAB. These models can then be used to tune PI/PID controllers using the different tuning methods and evaluate their performance based on closed-loop system response. Both the "drss" and "rss" functions in MATLAB are used to generate random state-space models with specified dimensions. However, the main difference between the two functions is in how they generate the state-space matrices.

The "drss" function generates random stable state-space models, which means that all the eigenvalues of the system matrix A have negative real parts. The function also ensures that the system is controllable and observable.

On the other hand, the "rss" function generates state-space models with random eigenvalues, which may result in a non-stable system. The function does not guarantee controllability or observability, and the resulting system may have some poorly conditioned modes.

Therefore, the "drss" function is recommended when you need a stable and well-conditioned system. However, if you need to test your control algorithm or observer design under different system conditions, you can use the "rss" function to generate a variety of random systems with different eigenvalues.

Appendix A is an example MATLAB code for generating a random state-space model using the "rss" function. In this code, we first define the system parameters such as the number of states, inputs, and outputs. We then use the "rss" function to generate a random state-space model with the specified dimensions. The function returns the state-space matrices A, B, C, and D. We then create a ss object using these matrices and display it using the "disp" function. By modification of the values of n, m, and p it is possible to generate a state-space model with the desired dimensions. The "rss" function generates random matrices with random eigenvalues, which may result in a non-stable system. But we mainly will focus on the double integrator pulse with pulse time delay system to analysis some of the PID controller tuning methods and compare their behavior.

In the next step, there are some experiment implementations of some tuning methods.

3.1 Double integrator pulse time delay process model

A commonly used double integrator with time delay process for tuning PD/PID controllers is the following the Equation (36): [6]

$$G(s) = \frac{K}{s(s+a)\mathrm{e}^{-\tau s}} \qquad (36)$$

where K is the process gain, a is the time constant of the process, τ is the time delay, and s is the Laplace variable.

Regarding the suggestions in task description, there is research on finding the model for a vessel movement application, we can model the system as a double integrator process. The vessel's position is the integral of its velocity, which is itself the integral of the acceleration. Thus, the transfer function of a vessel's position can be represented as Equation (37).

$$G(s) = \frac{1}{s^2} \qquad (37)$$

To incorporate time delay into the model, we can use a time delay block with a time constant T :

$$G(s) = \frac{\mathrm{e}^{-\tau s}}{s^2} \qquad (38)$$

This suggested transfer function can be used to tune PD/PID controllers for a vessel's position control.

To implement the double integrator with time delay process model in Equation (38) in MATLAB, Appendix B is a code with a controller for getting the step response of the process after 250 seconds and can reach the setpoint after this time with minimum overshooting.

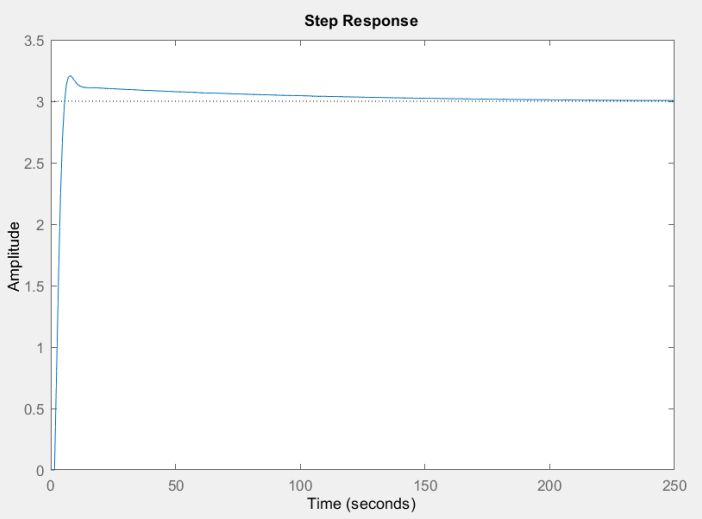


Figure 9, the step response of the double integrator pulse time delay process in 250 s.

In this code, we define the process parameters $K = 1$ as a constant number because in the Equation (38) it is considered equal to 1, and τ as time delay of the system to create the transfer function G using the tf function. We then define the controller gains K_p, K_d, and K_i and create the controller transfer function C using the "pidtune" function automatically. We combine the process and controller transfer functions to create the closed-loop system "T_total", which we use to plot the step response using the step function in figure 9. This system is implementation of Equation (38). Time delay in the double integrator with pulse time delay system can be fluctuated sometimes, to show that the system would be stable in other change of time delay parameter for an example we decreased the time delay form 1.5 s in figure 9 to 0.05 second, then the system reaches the setpoint much earlier than longer time delay. It means the tuner cooperates with time delay to tune the PID controller parameters. step response for this example is illustrated in figure 10 below.

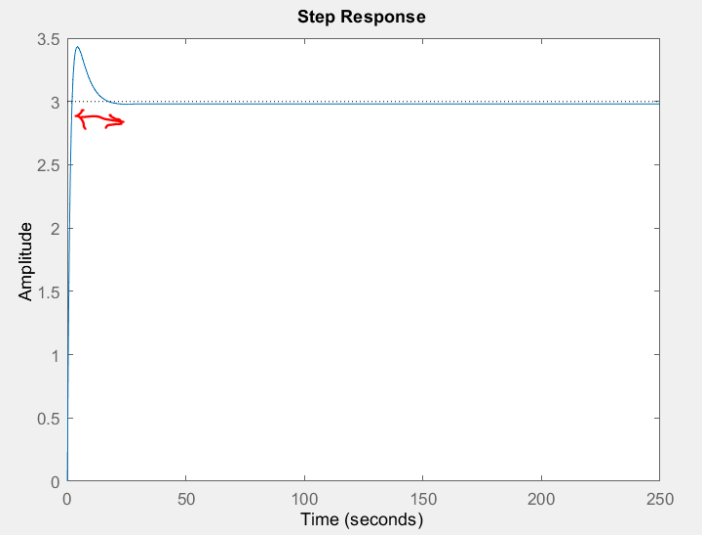


Figure 10, Step response for changing the time delay in the double integrator with pulse time delay system for auto tune PID controller.

3.2 Implementing double integrator with pulse time delay system impacted by disturbance function.

There is a suggestion for using the pidtune.m in MATLAB in simulation of the double integrator with pulse time delay system. In this code Appendix C. This code sets up a control system with a PID controller that is automatically tuned and simulates the closed-loop response to a setpoint value, while also considering a random disturbance signal. Like below Figure 11.

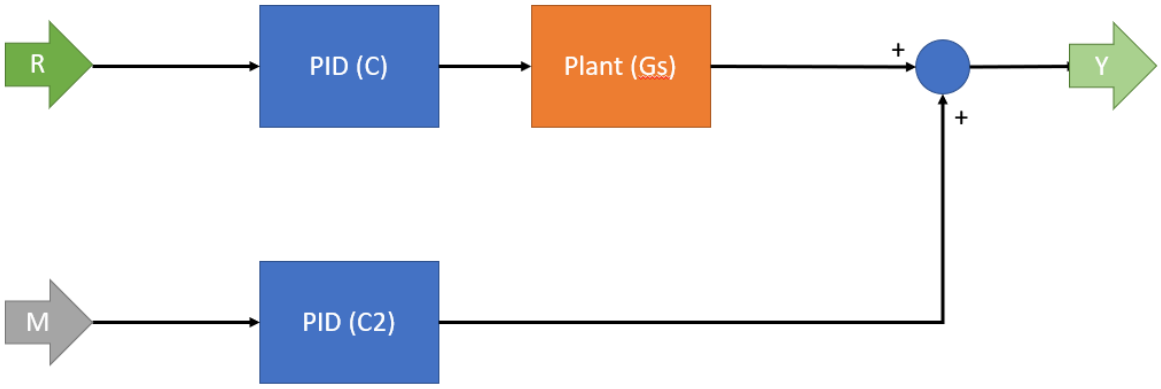


Figure 11, The example for adding the disturbance model into the system with two PID controller.

Here, R represents the setpoint signal and D represents the disturbance signal. The first PID controller receives the error signal between the setpoint and the output of the plant G(s) and adjusts the input to the plant accordingly. The second PID controller receives the disturbance

signal and adjusts the input to the plant to counteract the effect of the disturbance on the output. The resulting output of the closed-loop system is the sum of the plant output and the disturbance.

Appendix C`s code defines the time delay constant tau and sets up the Laplace variable s for later use in defining the transfer function. It defines the transfer function G of the system in Laplace domain. The transfer function is a mathematical representation of the relationship between the input and output of a system. In this case, G represents a second-order system with an exponential decay due to the time delay tau and sets up a PID controller C for the system with automatic tuning. A PID controller is a feedback controller that uses a proportional, integral, and derivative term to adjust the system's output based on the difference between the desired setpoint and the actual output. The pidtune function automatically tunes the PID controller parameters to achieve a desired response, which in this case is a good balance between speed and stability. There is additional to study the result with this disturbance model also, the code generates a random disturbance signal M that will be applied to the system. The rss function generates a random state-space model with specified damping and natural frequency, which in this case is 1. Then it sets up another PID controller C2 for the system with automatic tuning, but this time to compensate for the disturbance signal M. The pidtune function again automatically tunes the PID controller parameters to minimize the effect of the disturbance signal. Finally, we calculate the closed-loop response of the system to the setpoint value and disturbance signal and plots it using the step function. The feedback function combines the plant G (transfer function of the system) with the two PID controllers C and C2, and the 1 argument specifies that the feedback loop is closed. The resulting closed-loop transfer function T_total is then multiplied by the setpoint value setpoint, and the step function simulates the system's response to this input. The plot shows how the system's output responds over time to changes in the input setpoint value and the random disturbance signal M. step response of this system plotted below. Time delay is equal to 1.5 second.

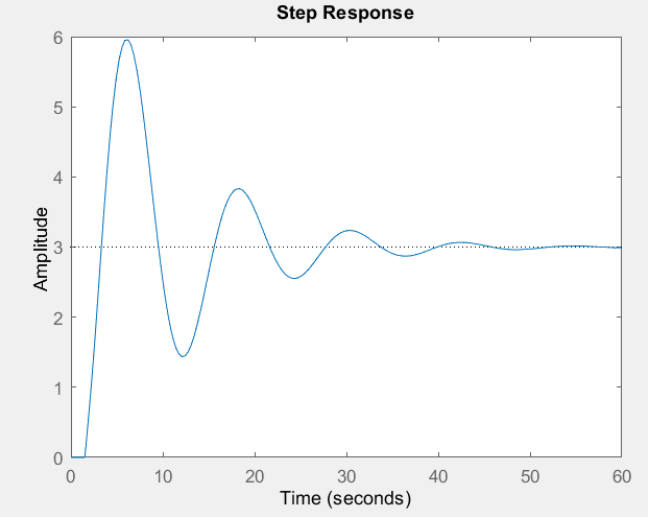


Figure 12, Step response of the system with disturbance model.

As you can see in figure 12 there is a increasing od amplitude value and system is oscillating much before reaching the set point, this example is just for showing that the pidtune.m can be used for two different input signal and make a close loop system with disturbance. The tuning method here is automatic tune in pidtune function in MTLAB.

3.3 Implementing double integrator with pulse time delay system with Modified internal model (IMC) tuning method for PID controller.

To flashback some information about IMC method, it comes that the modified IMC method involves first calculating the ultimate gain and period of the ideal controller for the time-delay-free process using the standard IMC equations. Then, the authors propose a modification to the ultimate period to account for the time delay by adding a term proportional to the time delay. Finally, the modified ultimate period is used to calculate the tuning parameters for the actual controller. In the article "Tuning PD and PID Controllers for Double Integrating Plus Time Delay Systems" by David Di Ruscio and Christer Dalen there is table that summarizes the modified IMC tuning method for different types of controllers (P, PI, PD, PID) and provides the necessary equations for calculating the tuning parameters. Here we will have some analysis on the modification of our model in Equation (38) which is double integrator with pulse time delay systems. It will try to explain step by step the experiment on this tuning method. It is called G_p in the code in Appendix D.

To calculate the ultimate gain and period of the ideal controller for the time-delay-free process using the standard IMC equations like.

$$K_c = \frac{1}{\tau\sqrt{2}} \tag{39}$$

$$P_u = \sqrt{\frac{\tau^2}{2\sqrt{2}}} \tag{40}$$

There are two parameters in the IMC model that would be modified, to justify the ultimate period $K_c \; and \; P_u$ to account for the time delay by adding a term proportional to the time delay. The Equation 41 make a P_{um}, this parameter is modification of the ultimate gain in the system.

$$P_{um} = P_u\left(1 + 0.2\left(\frac{\tau}{P_u}\right)\right) \tag{41}$$

There are estimation rolls in the modified IMC method to calculate the tuning parameters for the actual controller. The equations below are turning the PID controller parameters such as proportional gain, integral time and derivative time in the system, the out of PID controller would be the input of the double integrator with pulse time delay system.

$$K_p = 0.6 * K_c \quad (42)$$

$$T_i = 0.5 * P_{um} \quad (43)$$

$$T_d = 0.125 * P_{um} \quad (44)$$

In this way the PID controller transfer function would define Equation (45) to implement in MATLAB environment. This can Calculate the closed-loop transfer function and plot the step response.

$$G_c = K_p * \left(1 + \left(\frac{1}{T_i * s}\right) + T_d * S\right) \quad (45)$$

To have perfect feedback in control loop G_{cl} we have a definition to complete this loop with Equation (46) where:

$$G_{cl} = \left(\frac{G_c * G_p}{1 + (G_c * G_p)}\right) \quad (46)$$

The equation (46) calculates the closed-loop transfer function of the system when the PID controller transfer function is G_c and the process transfer function is G_p. The closed-loop transfer function represents the output of the system in response to a given input, and it is obtained by connecting the plant and the controller in a feedback loop as shown in Figure 13:

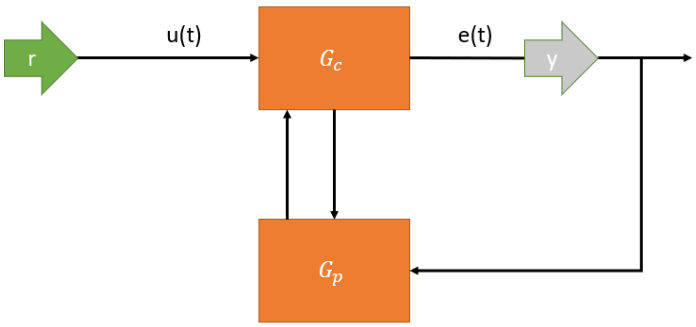


Figure 13, control loop in the system.

In this feedback loop, $u_{(t)}$ represents the output of the controller, r represents the set-point, $e_{(t)}$ represents the error between the set-point and the system output $y_{(t)}$, and G_c and G_p are

the controller and plant transfer functions, respectively. The closed-loop transfer function G_{cl} is defined as $\frac{y_{(t)}}{r}$, which is the output of the system $y_{(t)}$ in response to the set-point input r.

The closed-loop transfer function G_{cl} can be calculated in MATLAB using the feedback function, which takes the plant transfer function G_p and the controller transfer function Gc as input arguments and returns the closed-loop transfer function. The denominator of the closed-loop transfer function is set to $1 + (G_c * G_p)$ to ensure that the feedback loop is closed.

In this code on Appendix D, time delay is equal to 0.0015 second to reach the desired response, setpoint also selected as 5. Figure 14 indicates the step response of the control loop system by desired time delay.

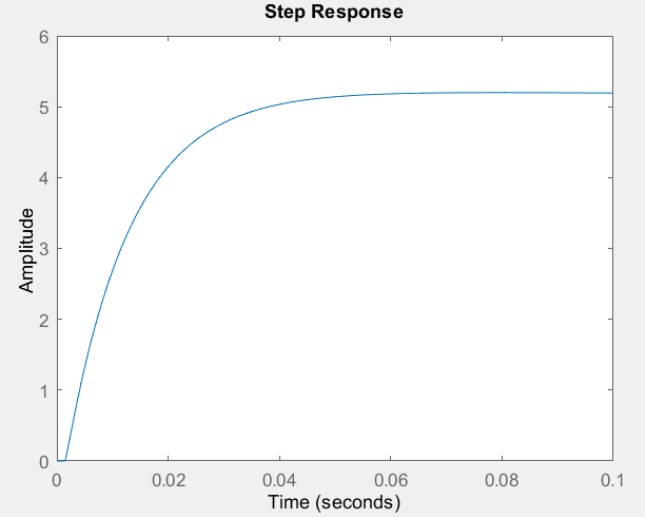


Figure 14, pure step response of the double integrator with pulse time system for IMC method for tuning PID controller.

As it is clear in figure 14 the system is quick fast to reach the set-point in less than 0.1 second. It seems that this method is very sensitive on the time delay, The tuning method modifies the ultimate period of the ideal controller to account for the time delay, and then calculates the tuning parameters for the actual controller based on the modified ultimate period. This ensures that the controller can compensate for the effects of the time delay and provide a stable and robust response. In general, the response of the system with the modified IMC tuning method to corporate time delay depends on the specific values of the tuning parameters, as well as the magnitude and characteristics of the time delay. However, the use of the modified IMC tuning method is expected to reduce the magnitude of the sustained oscillations and improve the stability and transient response of the system compared to a standard PID tuning method, especially for systems with large time delays.

3.4 Implementing double integrator with pulse time delay system with DDCD tuning method for PID controller.

The tuning method named like the author's name, The DDCD (Di Ruscio and Dalen Controller Design) method involves selecting the controller gains based on a simplified transfer function model in the article "Tuning PD and PID Controllers for Double Integrating Plus Time Delay Systems" by David Di Ruscio and Christer Dalen, is studied in this report to tune the PID controller for DIPTD system, here there some feature of this method which mentioned below before implementation:

The DDCD method is a systematic approach to tuning PD and PID controllers for double integrating plus time delay systems. It is based on a simplified transfer function model that captures the dominant dynamics of the system. This method focuses on tuning the controller gains based on the desired phase margin and gain crossover frequency of the open-loop system. This ensures that the closed-loop system is stable and has good performance in terms of settling time, overshoot, and steady-state error. The method uses a specific form of the transfer function model that separates the time delay from the rest of the dynamics. This allows the time delay to be handled separately using a Smith predictor or other time delay compensation techniques. Also, it provides explicit formulas for calculating the proportional gain K_p, derivative gain K_d,, and integral time constant T_i, based on the desired phase margin and gain crossover frequency of the open-loop system. This makes it easy to implement in practice. At the end it is applicable to a wide range of double integrating plus time delay systems, including those with small- or large-time delays. It has been shown to outperform other popular tuning methods, such as the Ziegler-Nichols method and the Cohen-Coon method, for this class of systems.

To evaluate these features, this report presents some technical information about implementation of the DDCD method for tuning a PD controller for the transfer function model in MATLAB environment to get the step response of the system.

The code in Appendix E, the Equation (38) is considered as plant included double integrator pulse with pulse time delay.

In this implementation first we used the same code for define the transfer function then for calculating the crucial gain K_c , there are some parameters like α is based on desired phase margin and δ is the gain crossover frequency in the close loop system, here we just used the desired phase margin in 63.4 degrees, thus the crucial gain is considered where $K_c = 0.707$.

We then design the PD controller using the DDCD method by setting the proportional gain K_p to 0.6 times K_c, the derivative time constant Td to 0.5 times the sum of the time delay tau and 0.1 times tau, and the derivative K_d gain to $\frac{T_d * K_p}{0.1}$.

The settings that we used to code in MATLAB environment is referred from the reference article for DDCD method there are defined like equations below:

$$K_p = 0.6 * K_c \quad (47)$$

$$T_d = 0.5 * (\tau + 0.1 * \tau) \quad (48)$$

To finalize the calculation of PD controller parameters in the system based on desired phase margin derivative gain K_d also define like

$$K_d = \frac{T_d * K_p}{0.1} \quad (46)$$

I didn't use the recommended settings for the closed-loop damping ratio and natural frequency based on the process time constant T_p in the MATLAB code. The DDCD method does not explicitly specify the closed-loop damping ratio and natural frequency; instead, it focuses on tuning the controller gains based on the desired phase margin and gain crossover frequency of the open-loop system.
By implementation of DDCD method for tuning PD controller, the step response of the system would be like figure 15.

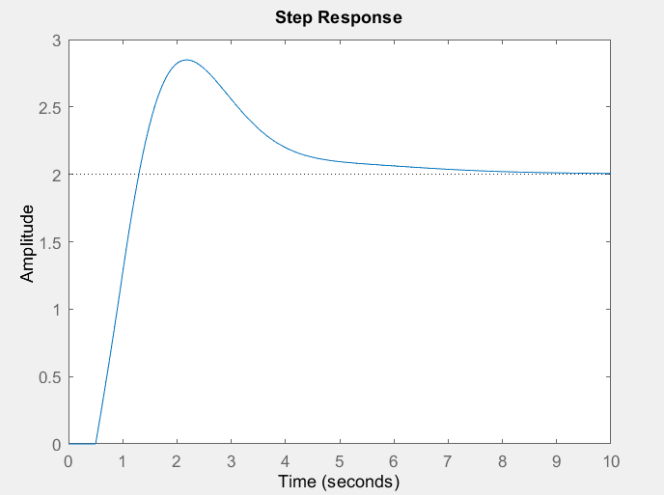


Figure 15, Step response of the DDCD tuning method for PD controller in DIPTD system.

Where τ is equal to 0.5 second the signal`s amplitude increasing to reach the setpoint which is equal to 2, there is a little overshoot on the response and without any other noises it is coming to manipulate the output signal to setpoint.

To check the wide rang covering of the delay value we executed the code with lower delay to see the effects on the step response, by decline the τ equal to 0.05 second step response has oscillation around the setpoint and after few seconds is coming to reach the setpoint the step response in figure 16 is that pure like lager time delay but it is applicable for implementing in such a system.

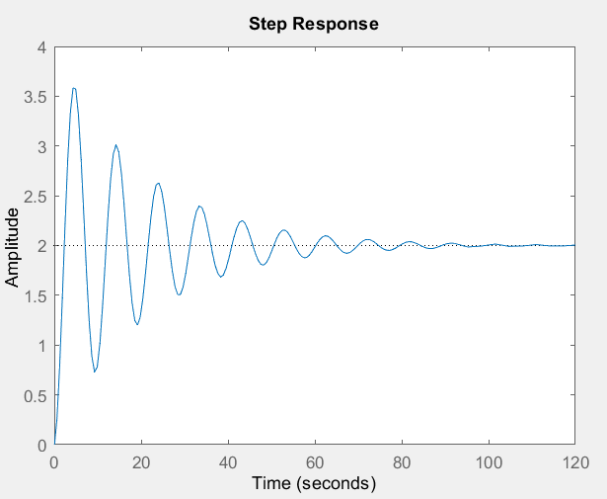


Figure 16, Step response for smaller time delay in the DIPTD system.

There main different between figure 15 and figure 16 is that the output signal reaches the setpoint quicker and without significant oscillation in less than 10 seconds where τ is larger.

4 Conclusion

Both the DDCD method and the Modified IMC (Internal Model Control) tuning method are used for tuning PD and PID controllers for dynamic systems. However, there are some differences between these two methods that can be highlighted:

- **Model representation**: The DDCD method selects controller gains based on a simplified transfer function model of the system, while the Modified IMC tuning method uses a more detailed model of the system.
- **Complexity:** The DDCD method is a simpler method compared to the Modified IMC tuning method, as it involves fewer steps and calculations.
- **Tuning parameters:** The DDCD method tunes the proportional and derivative gains separately, while the Modified IMC tuning method uses a single tuning parameter, the lambda parameter, to adjust both gains simultaneously.
- **Time delay:** The DDCD method can handle time delays in the system, while the Modified IMC tuning method assumes that there is no time delay.

These two tuning methods can deal with time delays in the system. However, the DDCD method is specifically designed to handle systems with time delays, while the Modified IMC tuning method assumes that there is no time delay. In the DDCD method, the controller gains are selected based on a simplified transfer function model that includes a time delay term. The method uses a heuristic approach to determine the controller gains based on the system's time delay and other parameters such as the desired closed-loop response speed, phase margin and derivative time delay. This approach can lead to reasonably good control performance even in the presence of time delays. On the other hand, the Modified IMC tuning method assumes that the time delay can be eliminated by using an internal model of the system. While this approach can work well for systems with very small-time delays which is 0.0015 seconds here, it may not be effective for systems with larger time delays or for systems where the time delay varies significantly with operating conditions like positioning of vessels. Therefore, in general, the DDCD method may be a better choice for systems with significant time delays, as it is specifically designed to handle such systems. However, the choice between these methods ultimately depends on the specific requirements and characteristics of the system being controlled.

References

[1] SYSTEM THEORY STATE SPACE ANALYSIS AND CONTROL THEORY, PhD (Dr. Ing.) David Di Ruscio, Lecture Notes in Control Theory

[2] MATLAB control theory YouTube channel, https://www.youtube.com/@MATLAB

[3] Automatic Control book, Finn Aakre Haugen, 6th September 2019

[4] On Tuning PI Controllers for Integrating Plus Time Delay Systems, David Di Ruscio, Vol. 31, No. 4, 2010, pp. 145-164, ISSN 1890-1328

[5] Tuning PD and PID Controllers for Double Integrating Plus Time Delay Systems, David Di Ruscio, Christer Dalen,Vol. 38, No. 2, 2017, pp. 95-110, ISSN 1890-1328

[6] "Feedback Control of Dynamic Systems" by Gene F. Franklin, J. Da Powell, and Abbas Emami-Naeini.

[7] "Modern Control Engineering" by Katsuhiko Ogata.

Appendices

Appendix A, Example code for “rss” function in MATLAB.

```
% Define random disturbance
rng(0);           % set seed for reproducibility
M = rss(3,1,1);   % generate 3x1 random matrix with damping=1 and natural
frequency=1
```

Appendix B, Implementing the DIPD process.

```
% Define the time delay and Laplace variable
tau = 0.05;
s = tf('s');

% Define the transfer function
G = exp(-tau*s)/(s^2);

% Set up the PID controller with automatic tuning
C = pidtune(G,'PID');

% Define the setpoint value
setpoint = 3;


% Plot the closed-loop response with setpoint
T_total = feedback(G*C,1);
step(setpoint*T_total);
```

Appendix C, double integrator with pulse time delay transfer function with random disturbance.

```
% Define the time delay and Laplace variable
tau = 1.5;
s = tf('s');

% Define the transfer function
G = exp(-tau*s)/(s^2);

% Set up the PID controller with automatic tuning
C = pidtune(G,'PID');

% Define the setpoint value
setpoint = 3;

% Define random disturbance
rng(0);           % set seed for reproducibility
M = rss(3,1,1);   % generate 3x1 random matrix with damping=1 and natural
frequency=1

% Set up the PID controller with automatic tuning to get the disturbance signal
C2 = pidtune(M,'PID');


% Plot the closed-loop response with setpoint
T_total = feedback(G*C*C2,1);
step(setpoint*T_total);
```

Appendix D, Modified IMC method for tuning the PID controller in double integrator with pulse time delay system.

```
%Define the process transfer function
tau = 0.0015; % time delay
s = tf('s');
Gp = exp(-tau*s)/s^2;

% Calculate the ultimate gain and period of the ideal controller for the time-
delay-free process using the standard IMC equations
Kc = 1/(tau*sqrt(2));
Pu = pi*sqrt(tau^2+4)/(2*sqrt(2));


% Modify the ultimate period to account for the time delay by adding a term
proportional to the time delay
Pum = Pu*(1 + 0.2*tau/Pu);


%  Calculate the tuning parameters for the actual controller
Kp = 0.6*Kc;
Ti = 0.5*Pum;
Td = 0.125*Pum;

%  Define the PID controller transfer function
Gc = Kp*(1 + 1/(Ti*s) + Td*s);


%  Calculate the closed-loop transfer function with set point and plot the step
response
setpoint = 5; % set-point value
Gcl = (Gc*Gp)/(1+Gc*Gp); % feedback(Gc*Gp, 1);
Gcl_sp = setpoint*Gcl;
step(Gcl_sp);
```

Appendix E, DDCD tuning method with Kc parameter from ZN recommended value.

```
% Define the transfer function model
tau = 0.5;  % time delay parameter
s = tf('s');
Gp = exp(-tau*s)/s^2; % transfer function G


% Calculate the crucial gain Kc
%phi_des = 63.4;  % desired phase margin in degrees
%wgc = 1/tau;  % gain crossover frequency
%alpha = (1+sin(deg2rad(phi_des)))/(1-sin(deg2rad(phi_des)));
% Kc = alpha*wgc^2;
Kc = 0.707;


% Design the controller using the DDCD method
Kp = 0.6*Kc;  % proportional gain
Td = 0.5*(tau+0.1*tau);  % derivative time constant
Kd = (Td*Kp)/0.1;  % derivative gain
C = pid(Kp,0,Kd);  % PD controller

setpoint = 2; % define a setpoint for the controller.

% Compute the closed-loop transfer function
Gcl = feedback(C*Gp,1);
Gcl_sp = setpoint*Gcl;


% Plot the step response of the closed-loop system with setpoint
step(Gcl_sp);
```

Appendix F, Task description of this report.

University of South-Eastern Norway

Faculty of Technology, Natural Sciences and Maritime Sciences, Campus Porsgrunn

FMH606 Master's Thesis

Title: Tuning PID Controllers for Double Integrating Plus Time Delay (DIPTD) processes

USN supervisor: David Di Ruscio

External partner: None

Task background:
Recently methods for tuning PI/PID and PD/PID for Integrating and Double Integrating Plus Time Delay (IPTD and DIPTD) processes have been published. Se also. There also is a new PID tuning method in MATLAB, the pidtune.m, algorithm. It would be of grate interest to compare these algorithms.
As an DIPTD example process for tuning PD/PID controllers, a double integrating process for describing a vessel movement may be used. This may be used to formulate a Dynamic Positioning system for a marine vessel. The OSP simulator may also be used for testing the controllers.

Task description:

1. Give a literature survey of methods for tuning PID controllers for DIPTD plants, in particular the DIPTD and the pidtune.m methods.
2. Compare the different methods by simulation experiments using MATLAB or similar (Octave/Python). Random SSMs generated from the MATLAB rss.m or drss.m functions may be used in order to generate models for tuning the PI/PID controllers. A laboratory physical process may also be used for the experiments.

Student category: IIA students

Practical arrangements:

None.

Supervision:

As a general rule, the student is entitled to 15-20 hours of supervision. This includes necessary time for the supervisor to prepare for supervision meetings (reading material to be discussed, etc).

Signatures:

Supervisor (date and signature): 12/14-23

Student (write clearly in all capitalized letters): Arash Amin Ali Abadi, 230757

Student (date and signature):

Address: Kjølnes ring 56, NO-3918 Porsgrunn, Norway. **Phone:** 35 57 50 00. **Fax:** 35 55 75 47.