Otto-von-Guericke-University Magdeburg
Faculty of Electrical Engineering and Information Technology
Institute for Automation Engineering
Chair for Automation/Modeling

# Automation Lab

## Temperature Control Lab 2
## (TCL 2)
## Experiment Report

submitted: December 15, 2024

by: Jigarkumar Ratilal Panchal
Siddharth Kamal Menon

# Contents

# 1 Introduction

This project report focuses on the implementation and analysis of PI controllers for a Heater-Sensor-System, building upon the transfer function identified in Temperature Control Lab 1 (TCL 1). The objectives include designing PI controllers in the frequency domain, applying standard tuning methods, and validating performance through simulation and practical testing. The report also explores the impact of varying phase margins and crossover frequencies, as well as comparing the effectiveness of different tuning rules such as Ziegler, $T_\Sigma$, and Opelt. Practical experiments include disturbance handling and set-point adjustments to assess controller performance comprehensively.

# 2 Controller Design

The objectives of this lab are to design and analyze PI controllers using the transfer function from TCL 1 report besed on TCL 1 task [1]. The goals include gaining practical experience with standard controllers, applying frequency-domain design methods, and validating controller design strategies without prior model identification. MATLAB/Simulink will be used for implementation and testing.

## 2.1 Preparation Part

### 2.1.1 PI controller design in the frequency domain

#### A) Gain adjustment with phase margin $\varphi_M = 30^\circ$ and $\varphi_M = 70^\circ$.

From Temperature Control Lab 1 report and calculation steps describe in Controller Design in the Frequency Domain [2] it is clear that the Transfer function $G(s)$ and $G_c(s)$ are

$$G(s) = \frac{K}{(1+T_1 s)(1+T_2 s)} 1001[3] \quad and \quad G_c(s) = \frac{K_P(1+T_I s)}{T_I s}, \quad T_1 > T_2 \implies T_I = T_1$$

Where

$$G(s) = \frac{0.5833}{(1+168.7770s)(1+18.7530s)} \quad and \quad G_c(s) = \frac{K_P(1+168.7770s)}{168.7770s}$$

Hence, the open loop transfer function is,

$$G_{ol}(s) = G(s) \times G_c(s) = \frac{0.5833 \cdot K_P(1+168.7770s)}{(1+168.7770s)(1+18.7530s)(168.7770s)}$$

Which gives the $G_{ol}(s)$ ,

$$G_{ol}(s) = \frac{0.5833 \cdot K_p}{(1+18.7530s)(168.7770s)}$$

While keeping the value of $K_p = 1$,

$$G_{ol}(s) = \frac{0.5833}{(1+18.7530s)(168.7770s)}$$

With the utilization of MATLAB Control System Designer(sisotool) and the blow plot obtained according to the gain adjustment with phase margin $\varphi_M = 30°$ and $\varphi_M = 70°$.

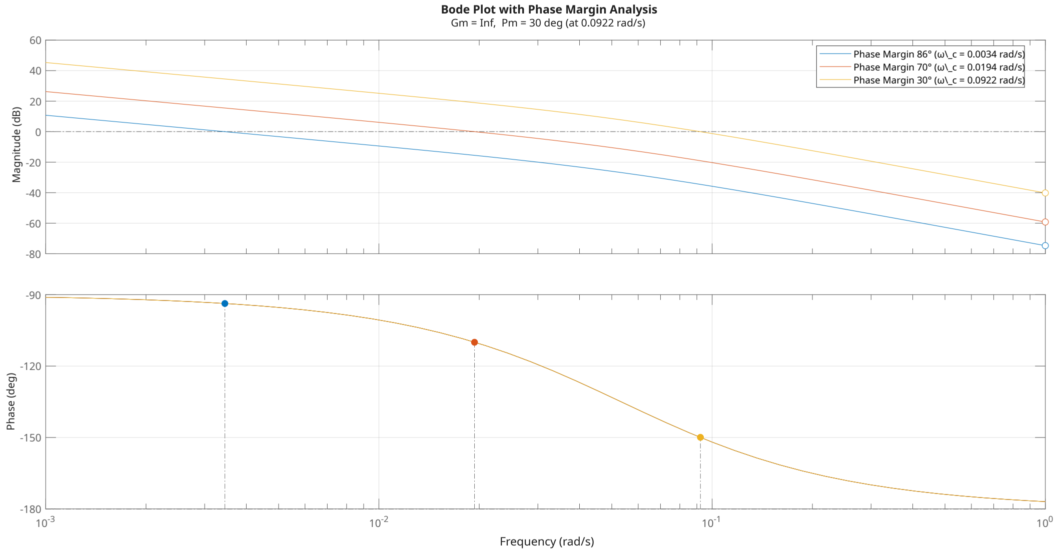


Figure 2.1: Gain adjustment at phase margin $\varphi_M = 30°$ and $\varphi_M = 70°$.

Now, as suggested in Controller Design in the Frequency Domain [2] the gain $K_p$ should be obtain with the help of interactive plot by sisotool. Alternatively,to calculate the gain $K_p$ manually as describe in Controller Design in the Frequency Domain:

Gain adjustment with phase margin:

At phase 70°, the values of frequency and magnitude are:

$$\varphi_M = 70' \quad f_{110} = 0.0194\,\text{rad/sec}, \quad M = -15.5\,\text{dB}.$$

Finding the values of $K^*$, and $K_p$:

$$20 \log K^* = 15.5 \implies K^* = 10^{\frac{15.5}{20}} = 5.9635$$

$$K_p = \frac{K^* \times K_p^*}{T_I} = \frac{5.9635 \times 1}{168.7770} = 0.0353$$

Open loop transfer function gain given by $K \times K_p^*$ as $G_{ol}(s) = G(s) \times G_c(s)$.

$$K \times K_p^* = 0.5833 \times 0.0353 = 0.0206$$

Therefore, the open loop transfer function with gain adjustment for phase margin $\varphi_M = 70$ is :

$$G_{ol}(s) = \frac{0.0206}{(1 + 18.75s)s}$$

Same result can be obtained by interactive function sisotool(G,C) in matlab. By using similar calculation it is easy to get open loop transfer function for $\varphi_M = 30$. As function sisotool(G,C) interactive, easy to use and accurate; the function sisotool(G,C) is used to derive controller gain $K_p$ for Open loop transfer function with phase margin $\varphi_M = 30$. Therefore, the open loop transfer function with gain adjustment for phase margin $\varphi_M = 30$ is :

$$G_{ol}(s) = \frac{0.1843}{(1 + 18.75s)s}$$

### B) Improvement in performance by utilizing loop shaping

Loop shaping involves adjusting the loop transfer function to achieve desired characteristics. This process can be carried out by designing compensator to shape the loop transfer function $L(s) = G(s) \cdot G_c(s)$ across different frequencies. Alternatively, Bode plots and their additive properties can serve as essential tools for this purpose. To modify the Bode plot, components such as PID controllers, poles and zeros or lead and lag compensators are introduced to achieve the desired phase and magnitude responses.

Loop shaping for the transfer function obtained from TCL 1 is performed under the assumption of the following required parameters:

$$\Delta h_{\text{rel}} = 10\% \quad \text{and} \quad T_P = 10\,\text{seconds}.$$

The design will aim to meet these specifications while ensuring stability and desired system performance. By following the calculation steps as mentioned in Controller Design in the Frequency Domain [2] it is easy to come up with the lead compensator to fulfill the required requirements. The closed-loop parameters ($PT_2^*$) can be calculated from the time-domain specifications as follows:

$$D = \sqrt{\frac{1}{1 + (\frac{\pi}{\ln \Delta h_{\text{rel}}})^2}} \approx 0.5912, \quad \omega_n = \frac{\pi}{T_P\sqrt{1 - D^2}} \approx 0.3895$$

The phase margin ($\varphi_M$) and corresponding frequency ($\omega_{\varphi_M}$) are calculated as:

$$\varphi_M \approx 100D \approx 59.12^\circ, \quad \omega_{\varphi_M} = \omega_n\sqrt{\sqrt{4D^4 + 1} - 2D^2} \approx 02812$$

To increase the crossover frequency $\omega_{\varphi_M}$ at phase margin $\varphi_M = 30^\circ$ and $\varphi_M = 70^\circ$; a lead element is introduced into the open-loop transfer function:

$$G_{ol}(j\omega) = \frac{0.5833 \cdot K_p}{(1 + 18.7530j\omega)(168.7770j\omega)} \times G_{\text{lead}}(j\omega)$$

where

$$G_{\text{lead}}(j\omega) = \frac{1 + \frac{j\omega}{\omega_{\text{lead}}}}{1 + \frac{j\alpha\omega}{\omega_{\text{lead}}}}.$$

### Calculation of Lead Element Parameters

To achieve the phase shift of $\Delta\varphi_M$ at $\omega = \omega_{\varphi_M}$, the following steps are performed: 1. The phase shift condition: $\Delta\varphi_M = 65°$.

The relationship between the lead element parameters:

$\omega_{\text{lead}}$ and $\alpha$ are determined such that the desired phase shift is achieved at $\omega_{\varphi_M}$.

To calculate the parameters of the lead element for a phase shift of $\Delta\varphi_M$ at the frequency $\omega_{\varphi_M}$, the following steps are used:

$\varphi_{\max} = \Delta\varphi_M$. Calculated $\alpha$ is

$$\alpha = \frac{1 - \sin\varphi_{\max}}{1 + \sin\varphi_{\max}} \approx 0.049$$

keeping $\omega_{\max} = \omega_{\varphi_M}$. Calculated $\omega_{\text{lead}}$ is

$$\omega_{\text{lead}} = \omega_{\max}\sqrt{\alpha} = \omega_{\varphi_M}\sqrt{\alpha} \approx 0.06224$$

from,

$$G_{\text{lead}}(j\omega) = \frac{1 + \frac{j\omega}{\omega_{\text{lead}}}}{1 + \frac{j\alpha\omega}{\omega_{\text{lead}}}} = \frac{1 + 16.0668j\omega}{1 + 0.7873j\omega}$$

which will result in

$$G_{\text{lead}}(s) = \frac{1 + 16.0668s}{1 + 0.7873s}$$

These parameters are used to determine open loop transfer function.

$$G_{ol}(s) = \frac{0.5833 \cdot K_p}{(1 + 18.7530s)(168.7770s)} \times \frac{1 + 16.0668s}{1 + 0.7873s}$$

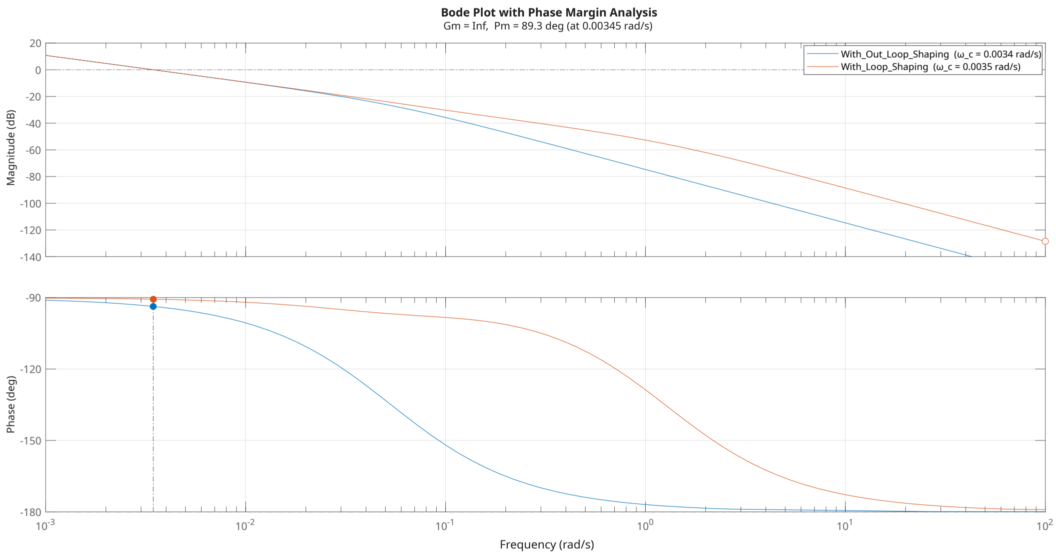


Figure 2.2: Loop Shaping.

It is easy to achieve the desired increased crossover frequency by repeating the same step of gain adjustment on open loop transfer function with lead compensator while kipping $K_p$=1 as before with the help of the Matlab function sisotool(G,C) at phase margin $\varphi_M = 30°$ and $\varphi_M = 70°$.

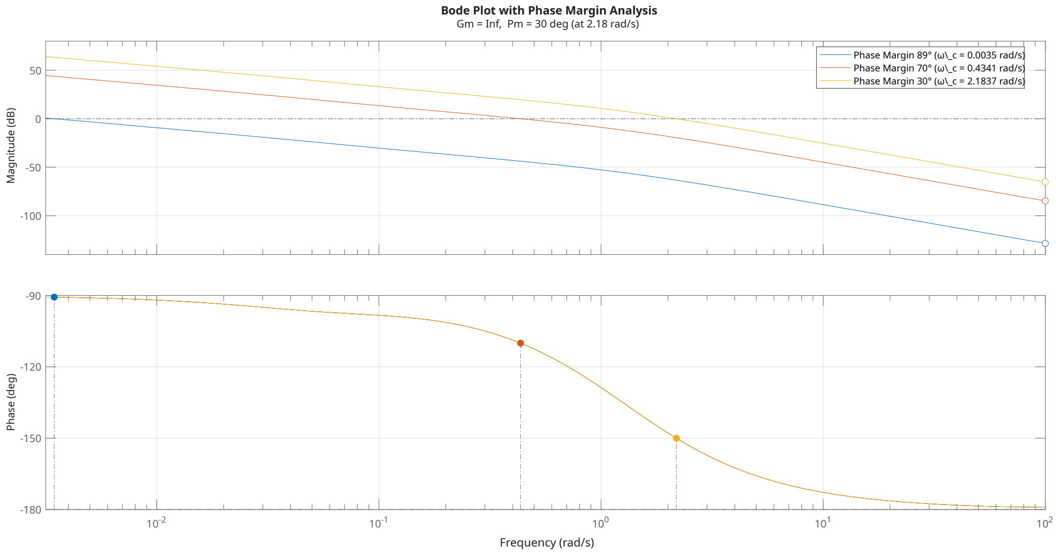


Figure 2.3: Gain adjustment at phase margin $\varphi_M = 30°$ and $\varphi_M = 70°$ with loop Shaping.

The open loop transfer function after gain adjustment for phase margin $\varphi_M = 70$ is :

$$G_{ol}(s) = \frac{8.581s + 0.5341}{14.76s^3 + 19.54s^2 + s}$$

And the open loop transfer function after gain adjustment for phase margin $\varphi_M = 30$ is :

$$G_{ol}(s) = \frac{81.44s + 4.069}{14.76s^3 + 19.54s^2 + s}$$

### 2.1.2 Comparison of crossover frequency with and without lead compressor

Main aim is to increase crossover frequency at phase margin $\varphi_M = 30°$ and $\varphi_M = 70°$ [4]to increase the response time of the system.

From the figure it is clear that due to lead compensator the crossover frequency $\omega_c$ increased significantly at the same phase margin of $\varphi_M = 30°$ and $\varphi_M = 70°$.

- crossover frequency at at phase margin $\varphi_M = 30°$ with out lead compensator is $\omega_c = 0.0992 rad/s$ and with lead compensator increased is $\omega_c = 2.1837 rad/s$.
- crossover frequency at at phase margin $\varphi_M = 70°$ with out lead compensator is $\omega_c = 0.0194 rad/s$ and with lead compensator increased is $\omega_c = 0.4341 rad/s$.

The step response of the obtain transfer function is given below in two separate figures, one with lead compensator(loop shaping) and other without lead compensator.

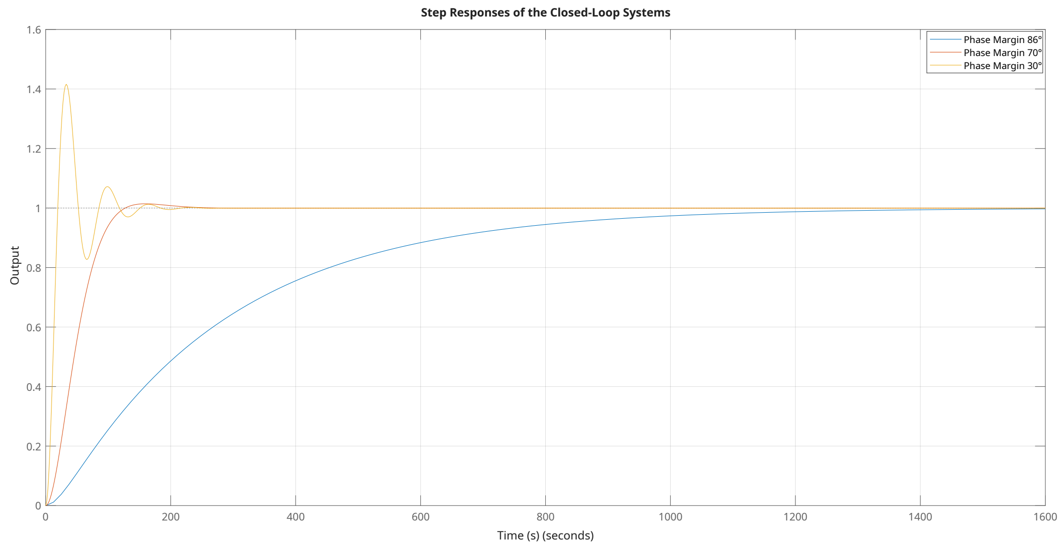


Figure 2.4: Step response without loop Shaping.

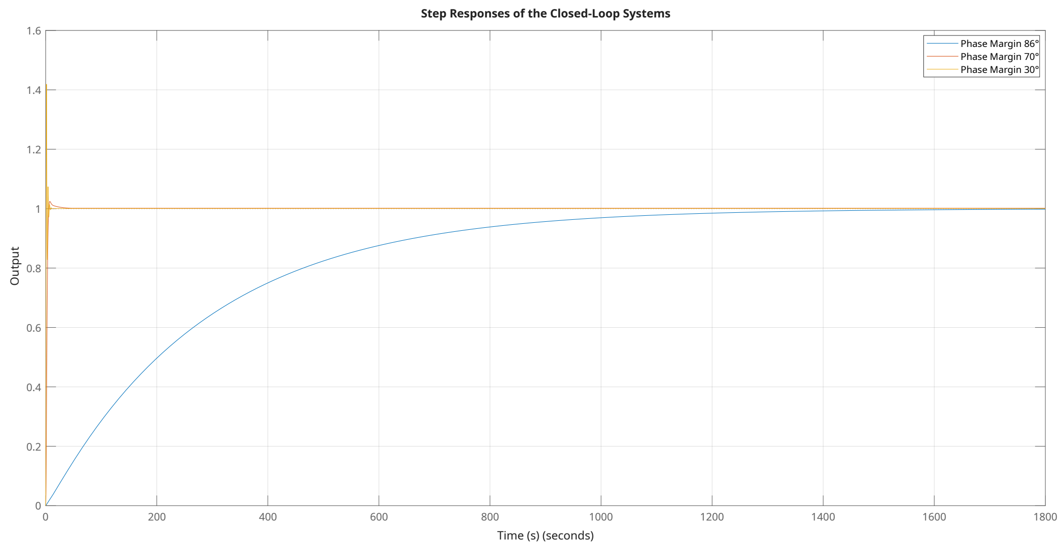


Figure 2.5: Step response with loop Shaping.

### 2.1.3 Design of the PI Controller

The design of the PI controller is performed using three tuning methods Ziegler Nicolas Tuning Rules, Σ Tuning Rules and Opelt Tuning Rules.

The transfer function values obtained from TCL 1 are used to design the PI controller. The transfer function for the PI controller in the frequency domain is given by:

$$G_c(s) = K_R\left(1 + \frac{1}{T_I s}\right) = \frac{K_R(1 + T_I s)}{T_I s} \quad [2]$$

Using the transfer function parameters from TCL 1, we set $T_a = 222.0356$, $T_u = 4.4532$, $K_s = 0.5833$ and $T\Sigma = T_a + T_u = 222.0356 + 4.4532 = 226.4888$ to design the PI controller according to the three rules mentioned above.

#### a) Ziegler Tuning Rules

Using Table 1 from the TCL 2 handout [5], the PI controller is given by:

$$K_R = 0.9 \cdot \frac{T_a}{K_s \cdot T_u} = 76.9309, \quad T_I = 3.33 \cdot T_u = 14.8292$$

$$G_c(s) = 76.9309\left(1 + \frac{1}{14.8292s}\right) = \frac{1141s + 76.9309}{14.8292s}$$

#### b) $T\Sigma$ Tuning Rules

Using Table 2 from the TCL 2 handout [5], the PI controller is:

$$K_R = \frac{1}{K_s} = 1.7144, \quad T_I = 0.7 \cdot T\Sigma = 158.5422$$

$$G_c(s) = 1.7144\left(1 + \frac{1}{158.5422s}\right) = \frac{271.8047s + 1.7144}{158.5422s}$$

#### c) Opelt Tuning Rules

Using Table 3 from the TCL 2 handout [5], the PI controller is:

$$K_R = 0.8 \cdot \frac{T_a}{K_s \cdot T_u} = 68.3830, \quad T_I = 3 \cdot T_u = 13.3596$$

$$G_c(s) = 68.3830\left(1 + \frac{1}{13.3596s}\right) = \frac{913.5695s + 68.3830}{13.3596s}$$

### 2.1.4 Implementation of the designed PI Controller

The performance of three types of PI controllers with the help of simulation is given below.

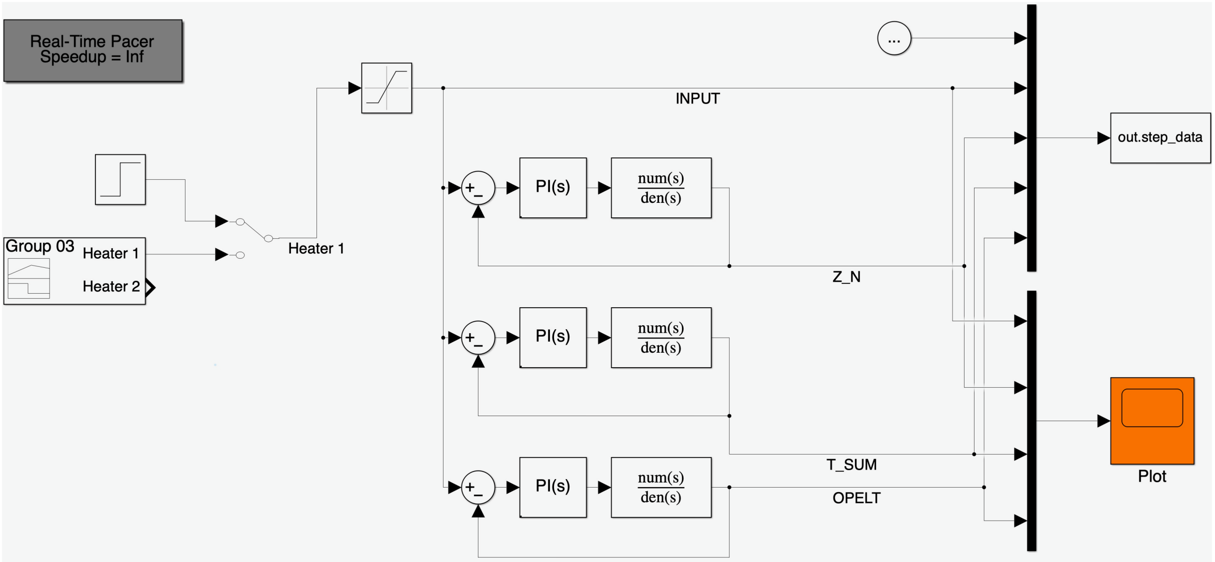


Figure 2.6: PI controller implementation with transfer function from TCL1 in Simulink.

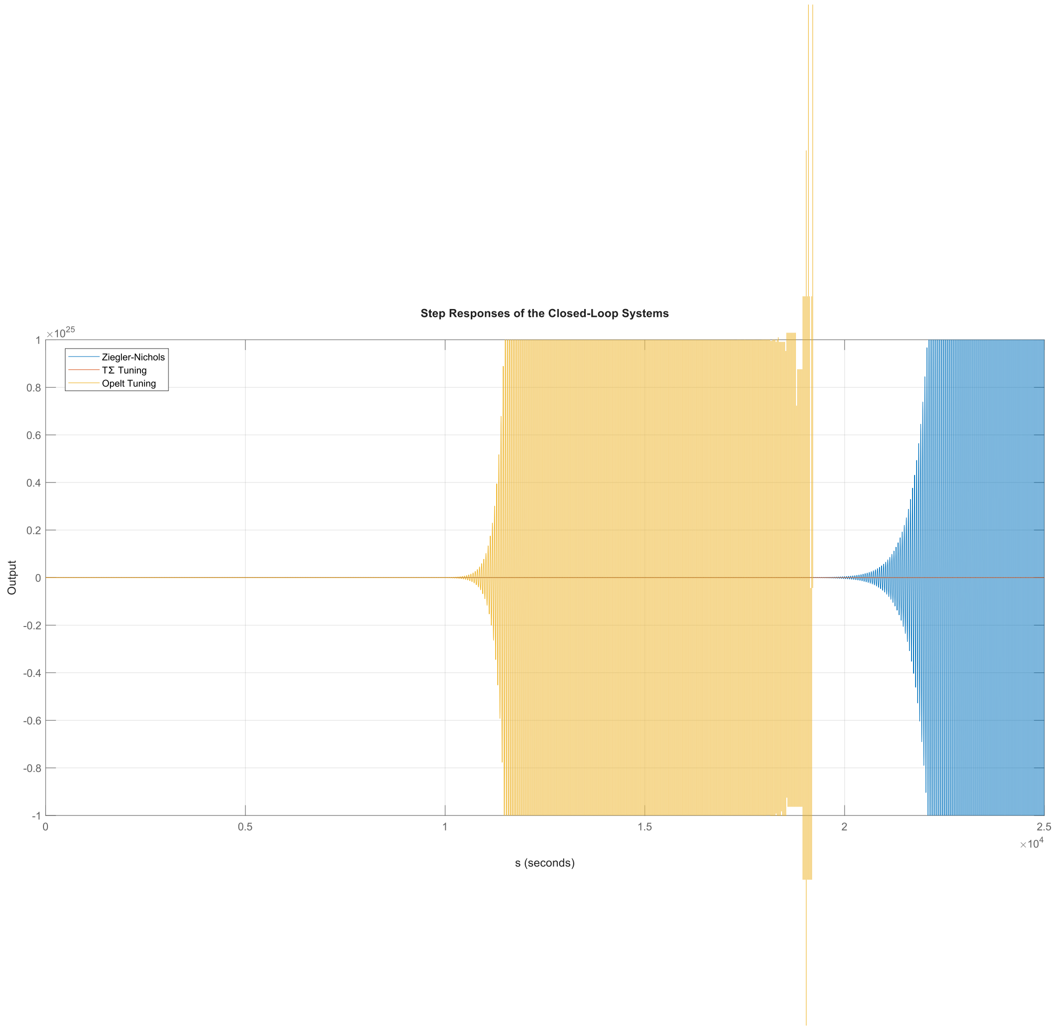


Figure 2.7: Implementation of the designed PI Controller.

From the figure it is noticeable that controllers designed with Ziegler-Nicholas and Opelt tuning method provides fast response with similar overshoot, whereas three some tuning method provides a controller with slow response with no overshoot.

## 2.2 Practical Part

### 2.2.1 Implementation of PI controller with TCL

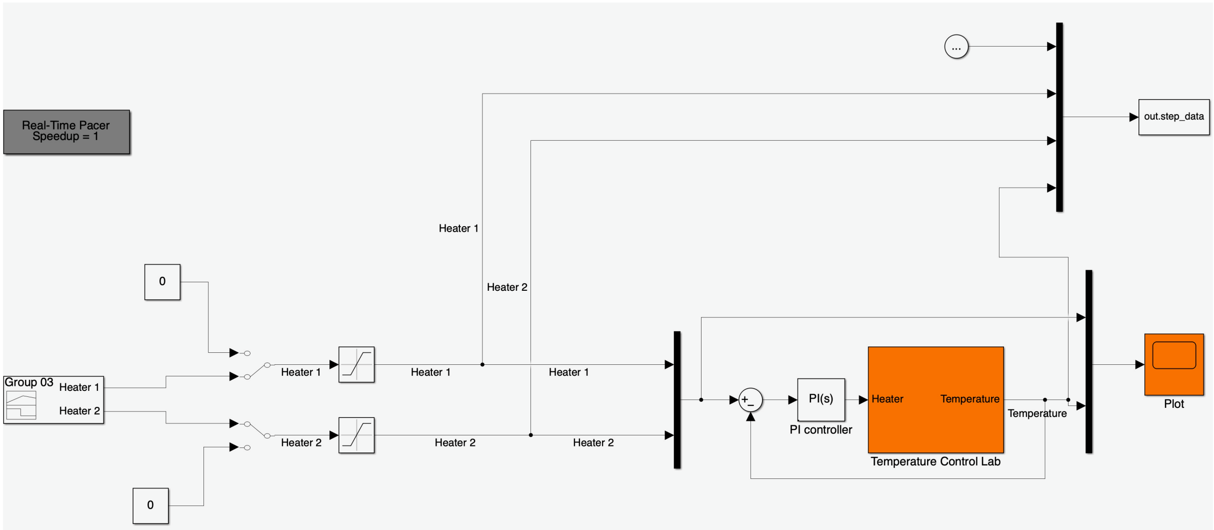


Figure 2.8: PI controller implementation with transfer function from TCL1 in Simulink.

### 2.2.2 Implement and test the controller

The controller will be implemented and tested by introducing disturbances through another heater and altering the set point.

Where,

- Set point of heater 1 system will be 30°C from 0 to 300 seconds and 60°C from 300 to 2100 seconds as set point for close-loop temperature control system.
- Set point of heater 2 system will be 27°C from 0 to 900 seconds, step of 27°C to 40°C at 900 second, remp of 40°C to 70°C from 900 to 1400, constant value of 70°C from 1400 to 1600 seconds, step of 70°C to 0°C at 1600 second and constant at 0°C from 1600 to 2100 seconds as set point for disturbance.

a) Controller performance with Ziegler-Nicholas tuning rules

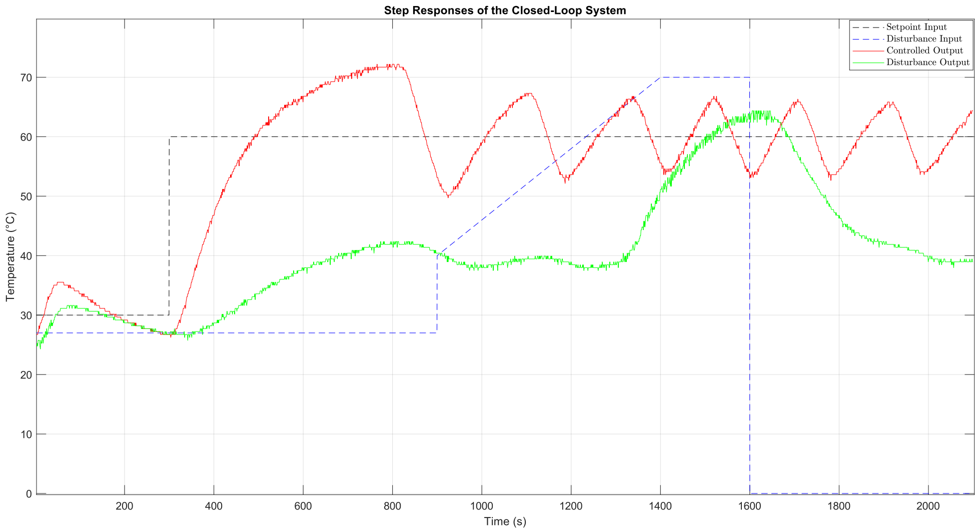


Figure 2.9: Controller performance with Ziegler-Nicholas tuning rules.

b)Controller performance $T\Sigma$ tuning rules

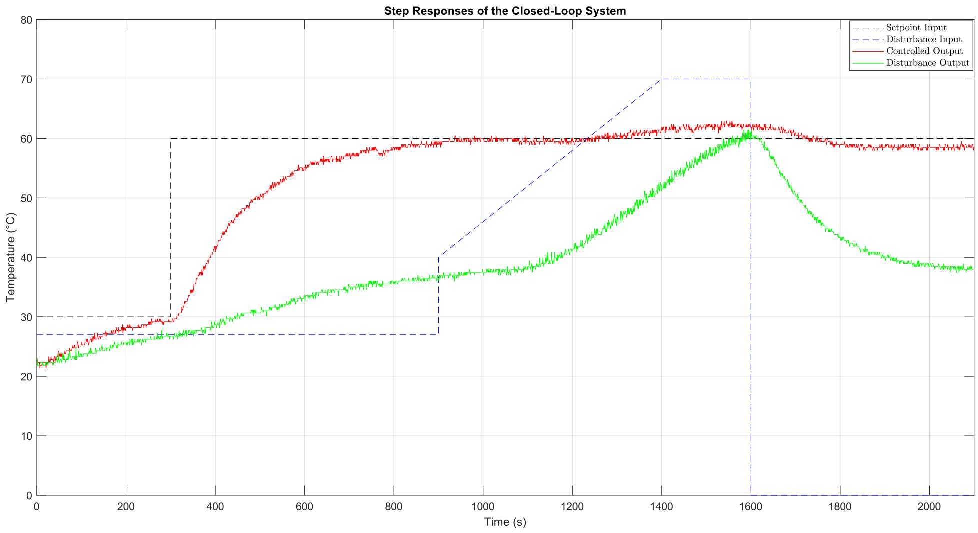


Figure 2.10: Controller performance with $T\Sigma$ tuning rules.

c) Opelt tuning rules

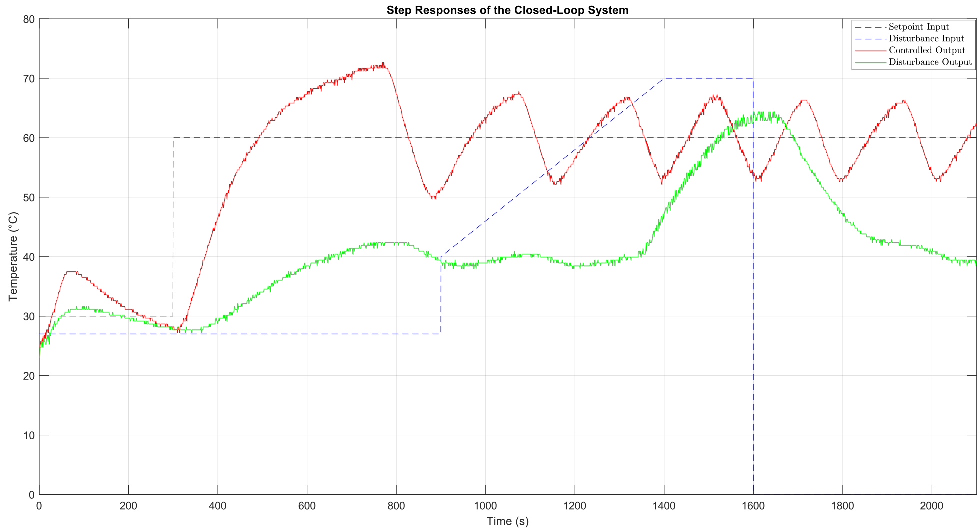


Figure 2.11: Controller performance with Opelt tuning rules.

### 2.2.3 Interpretation and comparison of the controller performance and results

The controller with Ziegler-Nicholas tuning rules and Opelt tuning rules perform very similarly both the controllers are good with disturbance rejection and response time on the other hand they give comparatively high overshoot, where controller tuned with $T\Sigma$ tuning rules give almost no overshoot but have very high response time due to which it can not reject high frequency disturbances easily.

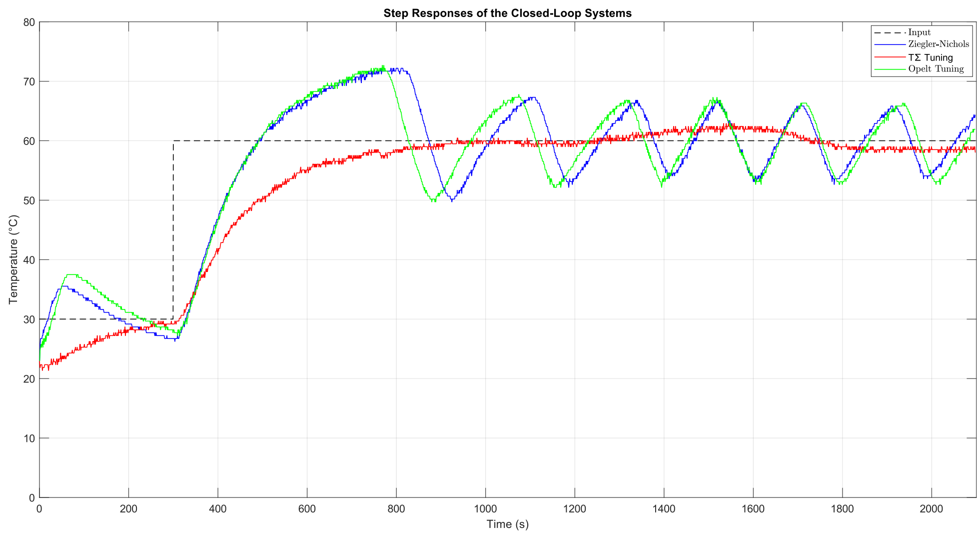


Figure 2.12: Controller performance with Ziegler-Nicholas tuning rules.

## 2.3 Conclusion

The tasks involved designing a PI controller for the Heater-Sensor-System using experimental data and tuning methods such as Ziegler-Nicholas, $\Sigma$ tuning, and Opelt rules. The implemented controllers were tested under disturbances introduced by another heater and set point changes. The controllers designed using Ziegler-Nicholas and Opelt tuning rules exhibited similar performance, excelling in disturbance rejection and response time. However, both controllers resulted in a comparatively high overshoot. In contrast, the controller tuned with $\Sigma$ rules achieved almost no overshoot but suffered from a significantly higher response time, limiting its ability to effectively reject high-frequency(small time duration) disturbances. These findings highlight the trade-offs between overshoot and response time when selecting a tuning method for specific applications.

## 2.4 Appeal

Respected faculty,

We are writing to formally address a significant issue encountered during the control design process, which is based on data from the TCL 1 report. The data in question appears to contain an unidentified error, likely attributable to an environmental factor or hardware malfunction. This error caused the sensor in TCL 1 to exhibit an unexpected time delay, leading to a time constant, $T_u = 4.4532$, that is substantially lower than anticipated. Based on reasonable estimates, $T_u$ should range between 9 and 11 seconds; however, this is not accurately reflected in the provided data by TCL 1 report.

As we were explicitly instructed to rely solely on TCL 1 report data for conducting the TCL 2 experiments, the erroneous data has severely compromised the validity of the results and the accuracy of the derived controller. Despite our best efforts to optimize the controller, we were unable to achieve improved performance due to the inherent faults in the data. This has directly impacted the reliability and effectiveness of the produced control system.

To support our claims, we have attached simulation results illustrating system behavior with a corrected time constant of $T_u = 11$ seconds. These results demonstrate performance metrics that align with expectations and highlight the inaccuracies introduced by the faulty data.

If deemed necessary, we are willing to redo the experiment with new, accurate data to ensure the integrity and effectiveness of the control design. However, as the initial instructions explicitly mandated the use of TCL 1 data, we were unable to deviate from this requirement despite recognizing its deficiencies.

Given the outlined circumstances, we respectfully request a review of the data requirements or consideration for alternative approaches to address this issue. We are confident that a resolution will enable a more robust and reliable control design. Your understanding and support in this matter are greatly appreciated.

Sincerely,

Group 3

## Attachment

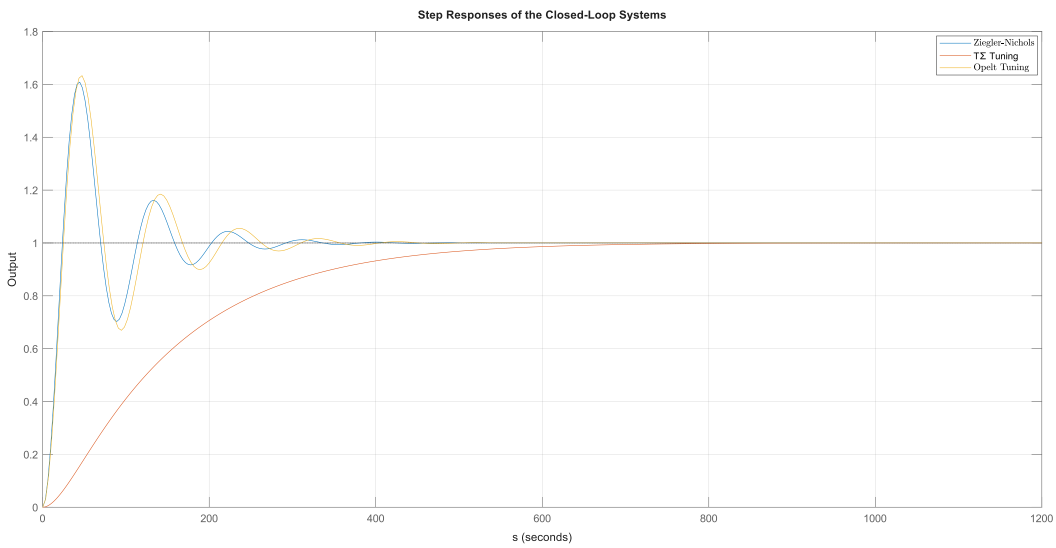


Figure 2.13: The PI Controller designed with $T_u = 11$.

# Bibliography


[1] "Material TCL 1," 2024, course: [Automation Lab], University: [Institute for Automation Engineering OVGU].

[2] "Controller design in the frequency domain," 2024, course: [Systems and Control], University: [Institute for Automation Engineering OVGU].

[3] "Handout TCL 1," 2024, course: [Automation Lab], University: [Institute for Automation Engineering OVGU].

[4] "Material TCL 2," 2024, course: [Automation Lab], University: [Institute for Automation Engineering OVGU].

[5] "Handout TCL 2," 2024, course: [Automation Lab], University: [Institute for Automation Engineering OVGU].