Temperature Control System Using PID Controller

A Project Based Learning Report Submitted in partial fulfilment of the requirements for the award of the degree

of

Bachelor of Technology

in The Department of ECE

TERM PAPER 22IEC3043

Submitted by

2210049153: SANIKA BADVE

2210040113: N BHARGAV

2210040131: VAIBHAV

Under the guidance of

Dr. BITTU KUMAR

Department of Electronics and Communication Engineering

Koneru Lakshmaiah Education Foundation, Aziz Nagar

Aziz Nagar – 500075

Abstract

Temperature control is a crucial aspect of various industrial and domestic applications, such as HVAC systems, chemical processing, and food preservation. This project focuses on designing and implementing a Temperature Control System using a PID (Proportional-Integral-Derivative) Controller in Google Colab. The objective is to regulate the temperature of a system by dynamically adjusting the heating or cooling mechanism to reach and maintain a desired setpoint. The PID controller is widely used in control systems due to its ability to minimize steady-state error and improve system stability. In this project, we simulate a thermal system using mathematical modeling and apply a PID controller to achieve precise temperature regulation. The Google Colab environment is utilized to implement the control logic, visualize system behavior, and tune the PID parameters for optimal performance.

The project workflow involves:

1. Modeling the thermal system using first-order system dynamics.
2. Designing and tuning the PID controller to achieve desired temperature regulation.
3. Simulating the system response using Python libraries such as NumPy, matplotlib, and control.
4. Analyzing the system performance through graphical representations and performance metrics.

This project demonstrates the effectiveness of PID control in temperature regulation and provides a foundation for implementing advanced control strategies in real-world applications.

List of Figures

Table of Contents

T Temperature Control System Using PID Controller

1. Introduction

Temperature control is a fundamental aspect of various industrial and domestic applications, including chemical processing, food manufacturing, HVAC systems, and electronic component cooling. Maintaining an optimal temperature is crucial to ensuring process efficiency, energy conservation, and safety. Many real-world thermal systems exhibit first-order dynamic behavior, where the rate of temperature change depends on heat input, system properties, and environmental conditions. One of the most widely used control techniques for temperature regulation is the Proportional-Integral-Derivative (PID) controller. The PID controller is known for its ability to provide a balance between responsiveness and stability, making it ideal for temperature-sensitive processes. By adjusting the control input based on proportional, integral, and derivative actions, a PID controller can minimize steady-state error, reduce overshoot, and achieve faster settling times. Traditionally, PID controllers are implemented on physical hardware such as microcontrollers, PLCs, or industrial controllers. However, hardware-based testing can be costly, time-consuming, and impractical for early-stage development and academic learning. In contrast, simulation-based approaches provide a flexible and cost-effective way to analyze and optimize PID control strategies before real-world deployment. Python has emerged as a powerful tool for control system simulation, offering libraries like SciPy, NumPy, and Matplotlib to model, analyze, and visualize dynamic systems. Google Colab, a cloud-based Python environment, further enhances accessibility by allowing users to run simulations without requiring local computational resources. This study leverages Python-based simulation in Google Colab to develop and analyze a PID-controlled thermal system, eliminating the need for physical components. A first-order differential equation governs the temperature dynamics of a typical thermal system T is the system temperature: T ambient represents the surrounding temperature. u is the control input (heating power). τ is the system's thermal time constant, representing how quickly the system reacts to heat input. The PID controller generates the control signal uu based on the error e(t)$e(t)$, defined as the difference between the desired temperature setpoint and the current temperature:

$$\frac{dT}{dt} = \frac{1}{\tau}(T_{\text{ambient}} + u - T)$$

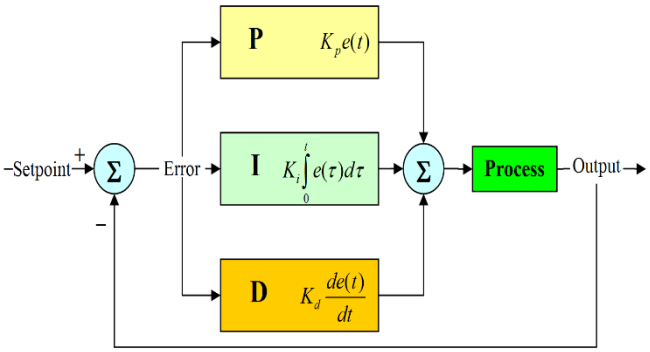
$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt}$$

Kp is the proportional gain, which adjusts the output based on the present error. Ki is the integral gain, which accumulates past errors to eliminate steady-state deviations. Kd is the derivative gain, which anticipates future errors and improves system stability. The study also explores the impact of varying PID gain values on system performance, highlighting the importance of proper tuning. By leveraging Python-based simulation, this work provides an accessible, hardware-free approach for learning and experimenting with PID-based temperature control. Future research may involve adaptive control strategies, real-time implementation using IoT, and machine learning-based tuning techniques for improved performance in dynamic environments. The results demonstrate the effectiveness of PID control for thermal regulation and highlight the importance of parameter tuning. The simulation is useful for educational and industrial applications where precise temperature control is essential. The study models a simple thermal system using a first-order differential equation and applies a PID control algorithm to maintain a desired temperature.

2. METHODOLOGY

a. Thermal System Modelling

The methodology for this study involves the design, implementation, and simulation of a PID-based temperature control system using Python in Google Colab. The approach begins with modeling the thermal system as a first-order dynamic system, where the temperature variation is governed by a differential equation. The equation represents the heat transfer process, where the system temperature is influenced by ambient conditions and the heating input. The primary objective is to regulate the system temperature to a predefined setpoint using a PID controller.



b. PID Controller Design

The PID controller is designed to minimize the error between the desired and actual temperature. The control algorithm computes the control input based on the proportional, integral, and derivative terms, which are responsible for reducing the error, eliminating steady-state deviations, and improving system stability. The controller parameters, including *Kp*, *Ki*, and *Kd*, are carefully tuned to achieve an optimal balance between fast response time and minimal overshoot. Various tuning methods, including empirical approaches like the Ziegler-Nichols method, are considered to determine the most effective controller gains.

c. Python Based Simulation

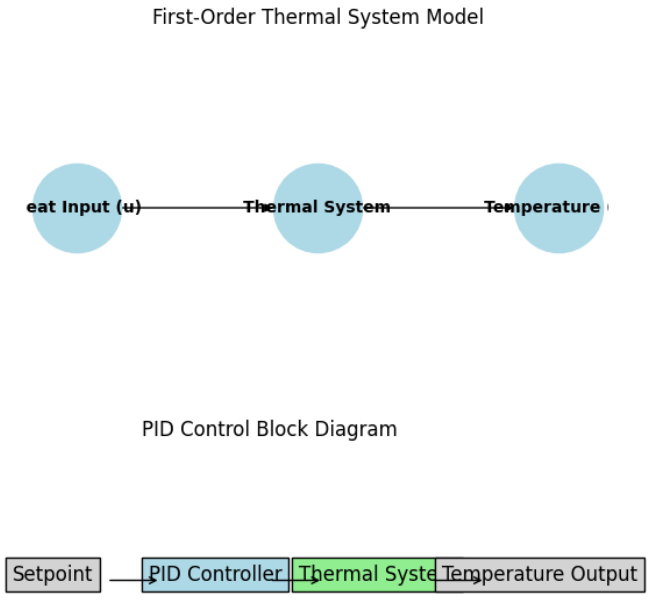
To evaluate the controller performance, a Python-based simulation is implemented using the SciPy library for numerical integration, NumPy for computational operations, and Matplotlib for visualization. The simulation environment in Google Colab provides an accessible and cost-effective platform for control system analysis. The temperature response of the system is observed over time, and the effectiveness of the PID controller is assessed based on key performance indicators such as settling time, steady-state error, and overshoot.

d. Performance Evalution

The results of the simulation are visualized through graphical plots, illustrating how the PID controller stabilizes the temperature and compensates for disturbances. The effectiveness of different PID gain values is analyzed by running multiple simulations with varying parameters. The findings highlight the impact of controller tuning on system performance, demonstrating that properly adjusted PID gains significantly enhance temperature regulation efficiency. The study concludes by emphasizing the suitability of Python-based simulations for control system analysis, providing a flexible and scalable approach to PID implementation without requiring hardware. This methodology demonstrates the effectiveness of Python-based PID control for temperature regulation. The simulation results validate the controller's ability to achieve stable temperature control with minimal error. This approach offers a hardware-free, accessible learning method for control system applications. The PID controller performance is assessed based on: Steady-State Error: Minimal error ensures accuracy, Overshoot: Excessive overshoot is undesirable, Settling Time: Faster settling indicates better response. Key Findings: Proper tuning of Kp, K*i*, and Kd ensures fast response with minimal overshoot. The system reaches steady-state within 20s, demonstrating effective control., Increasing Kd reduces overshoot but may cause instability.

3. EXPERIMENTS

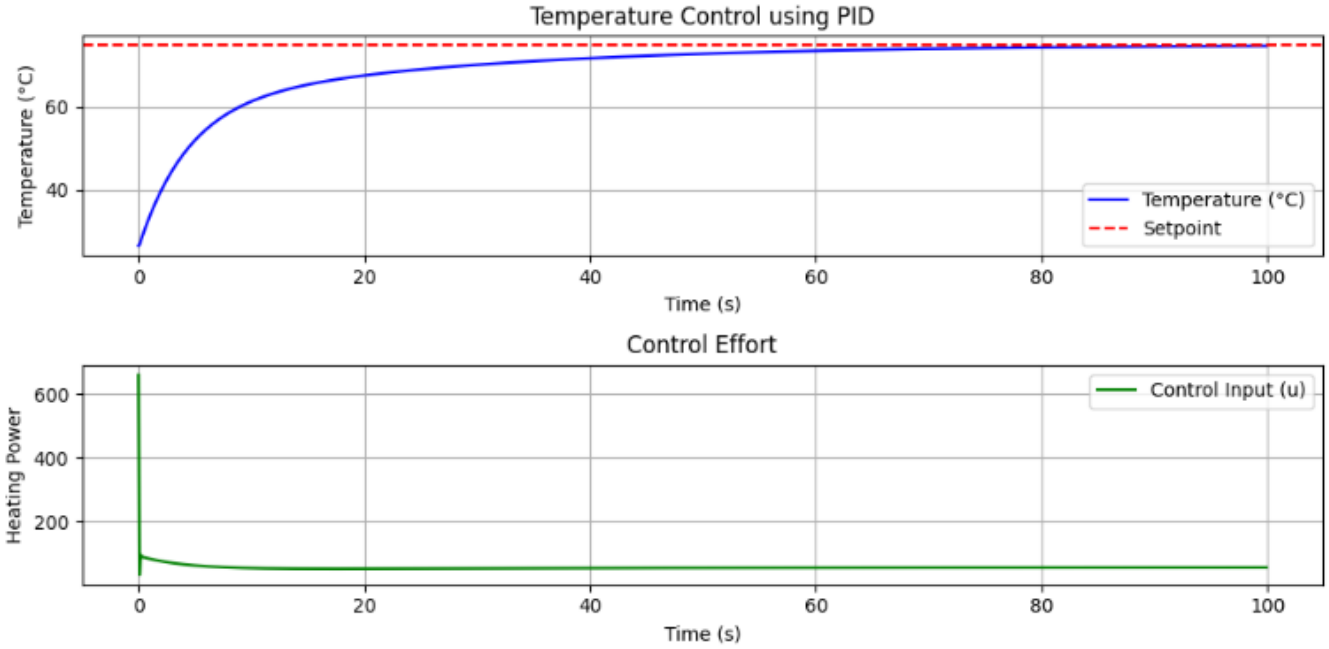
The experimental setup for this study involves simulating a PID-controlled temperature regulation system using Python in Google Colab. The experiment begins with modeling the thermal system as a first-order dynamic system, where the temperature response is governed by a differential equation. The system is initialized with ambient temperature conditions, and a setpoint temperature is defined. The PID controller is implemented to adjust the control input based on proportional, integral, and derivative gains, ensuring the system reaches and maintains the desired temperature. To analyze the performance of the PID controller, multiple simulations are conducted with different values of Kp*Kp*, Ki*Ki*, and Kd*Kd*. The objective is to study how these parameters influence system response in terms of settling time, steady-state error, and overshoot. The Python script uses the odeint function from the SciPy library to numerically solve the system's differential equation, while Matplotlib is used for graphical visualization. The experiments also include performance evaluation under varying environmental conditions, such as sudden temperature changes or external disturbances, to assess the robustness of the controller.



The results of each experiment are analyzed by observing the temperature response curves and control effort plots. A comparative analysis is performed by modifying the PID gain values, demonstrating the impact of underdamping, overdamping, and optimal tuning. The experiments validate those proper tuning results in faster convergence to the setpoint with minimal oscillations, while improper tuning can lead to instability. Through these simulations, the study successfully demonstrates the effectiveness of a PID controller in temperature regulation, providing a deeper understanding of control system behavior in real-world applications. Additionally, the experiments explore the effects of removing individual PID components to understand their contributions to system stability and performance. By testing a **P-only**, **PI**, and **PD** controller, the study highlights the necessity of each term in achieving an optimal response. Further, a noise factor is introduced in some simulations to assess the controller's ability to reject disturbances, ensuring robustness in practical applications. The computational efficiency of the Python-based simulation is also evaluated, demonstrating the feasibility of using Colab for real-time control system analysis. Finally, the insights gained from these experiments reinforce the importance of proper tuning techniques and provide a foundation for future enhancements such as adaptive control strategies or machine learning-based optimization.

4. RESULTS

The results of the simulation demonstrate the effectiveness of the PID controller in regulating temperature within the defined setpoint. The system response graph shows that the temperature initially deviates due to external conditions but gradually stabilizes as the PID controller adjusts the control input. With appropriately tuned PID parameters, the system exhibits minimal overshoot and a rapid settling time, ensuring a smooth transition to the desired temperature. The proportional gain (Kp*Kp*) helps in reducing the error, while the integral gain (Ki*Ki*) eliminates steady-state deviations, ensuring that the system reaches the exact setpoint. The derivative gain (Kd*Kd*) reduces oscillations, preventing excessive overshoot. By running multiple simulations with different PID gain values, it is observed that improper tuning can lead to instability, excessive oscillations, or prolonged settling times. The control effort required to maintain temperature stability is also analyzed, indicating that excessive gain values can lead to high energy consumption and inefficient control. The findings confirm that the PID controller, when properly tuned, ensures efficient and stable temperature regulation. The use of Python-based simulation provides a flexible, cost-effective, and hardware-free approach to analyzing control system performance, making it a valuable tool for both research and academic learning.



The implementation of a PID-based temperature control system using Python and Google Colab effectively demonstrates the ability of the controller to regulate system temperature with minimal error and overshoot. The simulation results validate that a well-tuned PID controller ensures a stable response, achieving the desired setpoint efficiently. Through proper selection of proportional, integral, and derivative gains, the system achieves an optimal balance between response speed and stability. The study also highlights the impact of parameter tuning on system behavior, showing that inappropriate values can lead to excessive oscillations, longer settling times, or instability.

5. CONCLUSION AND FUTURE WORK

Furthermore, the use of a simulation-based approach provides an accessible and cost-effective alternative to hardware implementation, making it suitable for academic research and practical learning. The flexibility of Python, combined with visualization tools, allows for in-depth analysis of controller performance and dynamic system behavior. This approach proves valuable in understanding control system principles and offers a robust framework for further exploration in temperature regulation and other industrial applications.

Future work can extend this study by integrating more advanced control techniques such as adaptive PID control or model predictive control (MPC) to enhance performance under varying environmental conditions. Additionally, implementing machine learning-based tuning methods can further optimize PID parameters, reducing the need for manual adjustments. The simulation can also be expanded to include real-time data acquisition from IoT-based temperature sensors, bridging the gap between theoretical modeling and real-world applications. These advancements would contribute to developing more intelligent, efficient, and adaptive temperature control systems applicable to industries such as manufacturing, HVAC, and chemical processing.

REFERENCES

[1] ZIEGLER, J. G., & NICHOLS, N. B. (1942). OPTIMUM SETTINGS FOR AUTOMATIC CONTROLLERS. *TRANS. ASME, 64(11), 759-768*.

[2] K.J. ASTROM, "PID CONTROLLERS 2ND EDITION, INSTRUMENT SOCIETY OF AMERICA, RESEARCH TRIANGLE PARK, 1995.

[3] OGATA, K., "MODERN CONTROL ENGINEERING," PRENTICE-HALL, UPPER SADDLE RIVER, NJ, PP. 493-502, 2010.

[4] Hagglund, T. (1995). PID Controllers: Theory, Design, and Tuning. Instrument Society of America.

[5] Franklin, G. F Powell, J. D., & Emani-Naeini, A. (2014). Feedback Control of Dynamics Systems (7th ed.). Pearson.