*applied sciences*

MDPI

*Article*

# Generative AI-Assisted PID Controller Tuning Under Performance and Actuator Constraints

**Leonas Balasevicius [1,*], Virginijus Baranauskas [1], Olga Strikuliene [2] and Darius Andriukaitis [3]**

[1] Department of Automation, Kaunas University of Technology, Studentu St. 48-320, LT-51367 Kaunas, Lithuania; virginijus.baranauskas@ktu.lt
[2] Technology and Entrepreneurship Competence Center, Panevezys Faculty of Technologies and Business, Kaunas University of Technology, Nemuno St. 33-218, LT-37164 Panevezys, Lithuania; olga.strikuliene@ktu.lt
[3] Department of Electronics Engineering, Kaunas University of Technology, Studentu St. 50-438, LT-51368 Kaunas, Lithuania; darius.andriukaitis@ktu.lt
* Correspondence: leonas.balasevicius@ktu.lt; Tel.: +370-37-300-291

## Featured Application

The proposed AI-assisted PID tuning framework may be relevant to industrial automation, process control, energy and electronics systems, robotics, and mechatronic applications, where controller parameters must be quickly determined from object input-output data, while satisfying predefined performance requirements. However, these application domains were not directly evaluated in this paper, and further validation is required using representative real-world case studies.

## Abstract

Generative artificial intelligence tools are increasingly being used to support engineering tasks; however, their applicability to process identification, controller tuning, and control performance prediction remains insufficiently investigated. This article evaluates the capabilities of GenAI systems to identify dynamic process models, design PI and PID controllers, and predict closed-loop performance metrics. Five benchmark process models representing first-order, delayed, integrated, and second-order dynamics were considered. Open-loop and closed-loop datasets were provided to two AI systems, which were then requested to identify process models, propose controller parameters, and predict overshoot, settling time, integral of time-weighted absolute error, and disturbance rejection time. The generated controller settings were validated using MATLAB/Simulink simulations, and both absolute and relative prediction errors were calculated. The results show that both AI systems could generate feasible controller parameters that produced stable closed-loop responses for most of the processes investigated. However, significant discrepancies were observed between predicted and simulated performance indicators. The integrating process exhibited the highest prediction errors, indicating substantial difficulties in interpreting integrating dynamics. The results suggest that current GenAI systems can support preliminary controller design and process analysis but cannot yet replace simulation-based validation for accurate quantitative performance assessment.

**Keywords:** PID controller tuning; generative artificial intelligence; data-driven control; open-loop and closed-loop process data

Check for updates

Academic Editors: Jesús Sandoval, Eusebio Bugarin Carlos, Jose Gudalupe Romero and Arkadiusz Gola

Received: 10 August 2026
Revised: 7 October 2026
Accepted: 19 October 2026
Published: 21 October 2026

**Copyright:** © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

## 1. Introduction

Proportional–Integral–Derivative (PID) controllers remain the most widely used control algorithms in industrial automation due to their simple structure, robustness, reliability, and ease of implementation. Despite the development of advanced control strategies, PID controllers continue to dominate industrial applications, including process control, manufacturing systems, energy conversion systems, robotics, and mechatronic devices. Recent industrial surveys indicate that PID-based solutions remain the preferred regulatory control technology because they provide acceptable performance with relatively low implementation complexity [1,2]. The quality of a closed-loop control system is strongly influenced by the selection of PID controller parameters. In practical applications, controller tuning involves achieving a balance among several often conflicting objectives, including low overshoot, fast settling time, zero steady-state error, effective disturbance rejection, and limited control effort. In addition, industrial actuators are commonly subject to amplitude, rate, or energy constraints, which requires the controller parameters to be selected not only according to performance criteria, but also according to physical limitations of the controlled system. Consequently, PID tuning continues to attract substantial research interest despite decades of theoretical and practical developments [3]. Historically, PID tuning has been dominated by analytical and empirical approaches. Classical methods, such as the Ziegler–Nichols tuning rules, established the foundations of practical controller design and are still widely used in industrial environments due to their simplicity [4]. Subsequently, model-based approaches, including Internal Model Control (IMC), provided more systematic tuning procedures capable of achieving improved robustness and closed-loop performance [5]. Although these methods remain important milestones in control engineering, they typically require process identification, mathematical modelling, or considerable domain expertise [6].

To address these limitations, researchers have increasingly explored artificial intelligence and computational intelligence techniques for controller tuning. Recent reviews demonstrate a growing interest in fuzzy systems, neural networks, genetic algorithms, particle swarm optimisation, differential evolution, and hybrid intelligent approaches for PID parameter optimisation. These methods often outperform classical tuning rules and are capable of handling non-linearities, uncertainties, and multiple design objectives [7,8]. However, their successful application usually requires dedicated optimisation frameworks, extensive computational resources, algorithm-specific parameter selection, and significant control engineering experience. As a result, their adoption in routine engineering practice remains limited.

In parallel, the rapid development of generative artificial intelligence (GenAI) and large language models (LLMs) has introduced a new paradigm for engineering design and decision support. Unlike conventional AI techniques that are typically developed for a narrowly defined optimisation task, LLMs combine domain knowledge, reasoning capabilities, numerical interpretation, and natural language interaction within a single framework. Recent studies have demonstrated the potential of GenAI for engineering design, knowledge extraction, design space exploration, and decision-making support, suggesting that these technologies may significantly transform engineering workflows [9,10]. The application of large-language models to control engineering has only recently begun to attract scientific attention. Preliminary studies indicate that state-of-the-art models are capable of solving control-related problems, interpreting engineering information, and helping in control system analysis. Furthermore, recent research highlights an emerging convergence between control theory and large language models, where LLMs can serve not only as knowledge assistants but also as active participants in controller design workflows.

However, the practical use of generative AI for determining controller parameters under explicit engineering requirements remains largely unexplored [11,12].

However, the practical use of Generative AI for process identification, controller tuning, and quantitative performance prediction remains largely unexplored. Existing studies on LLM-assisted control engineering focus mainly on control-theory assistance, conceptual controller design guidance, educational support, or qualitative engineering recommendations [13,14]. Comparatively few studies investigate whether Generative AI systems can directly generate PI/PID controller parameters from identified process models and reliably predict resulting closed-loop performance metrics before simulation-based validation [15]. Furthermore, the influence of open-loop and closed-loop process data on GenAI-assisted process identification and controller tuning has received limited attention in the existing literature. Consequently, the capability of current Generative AI systems to support the complete workflow from process data to controller design and closed-loop performance assessment remains insufficiently understood [13].

A particularly important research gap exists in the area of AI-assisted controller tuning under multiple performance and actuator (control signal amplitude) constraints. Existing AI-based PID tuning approaches mainly rely on optimisation algorithms or machine learning models trained for predefined tasks. On the contrary, generative AI systems offer the possibility of directly generating controller parameters from engineering specifications through a natural language interaction process. This capability could significantly simplify controller commissioning, reduce dependence on expert knowledge, shorten design cycles, and improve accessibility of advanced control design techniques for a broader engineering community [12,16].

Therefore, this paper investigates the applicability of generative artificial intelligence to the tuning of the PID controller under predefined performance and actuator (control signal amplitude) constraints. The proposed framework employs a large language model as an engineering assistant responsible for determining the parameters of ideal and parallel PID controllers. The tuning process is guided by explicit control objectives, including a maximum overshoot below a specified limit, a minimum settling time, zero steady-state error, effective disturbance rejection, and compliance with actuator (control signal amplitude) constraints.

Unlike previous studies that mainly investigate conversational assistance and qualitative controller design support, the present work evaluates the complete workflow from process identification and controller parameter generation to closed-loop performance prediction and simulation-based validation using both open-loop and closed-loop datasets. The present work contributes:

1. A unified evaluation framework covering process identification, controller tuning, and closed-loop performance prediction from process data;
2. An assessment of direct ideal and parallel PI/PID controller parameter generation by Generative AI systems;
3. A comparison of GenAI performance using both open-loop and closed-loop process data;
4. A quantitative evaluation of AI-generated closed-loop performance predictions;
5. A benchmark comparison with a conventional IMC-based tuning method.

The remainder of the paper is organised as follows. Section 2 presents the proposed generative AI-assisted tuning methodology. Section 3 describes the benchmark systems, controller structures, and evaluation procedures. Section 4 reports the results obtained and the comparative performance analysis. Section 5 discusses the advantages, limitations, and practical implications of the proposed approach and outlines the direction for future research. Finally, Section 6 summarises the conclusions.

## 2. Generative AI-Assisted PID Tuning Methodology

### 2.1. General Framework

The proposed methodology investigates whether a generative AI system can determine suitable controller parameters directly from simulation-generated process data, without explicit access to the analytical process model. The underlying hypothesis is that the AI model can infer relevant dynamic characteristics of the process from observed input-output behaviour and subsequently generate controller parameters that satisfy predefined performance requirements.

The proposed tuning workflow is illustrated in Figure 1 and consists of the following stages:

1. Acquisition of process data;
2. Construction of a structured engineering prompt;
3. Interpretation of process dynamics by the LLM;
4. Generation of candidate PI and PID controller parameters;
5. Closed-loop validation;
6. Performance evaluation;
7. Iterative refinement when necessary.

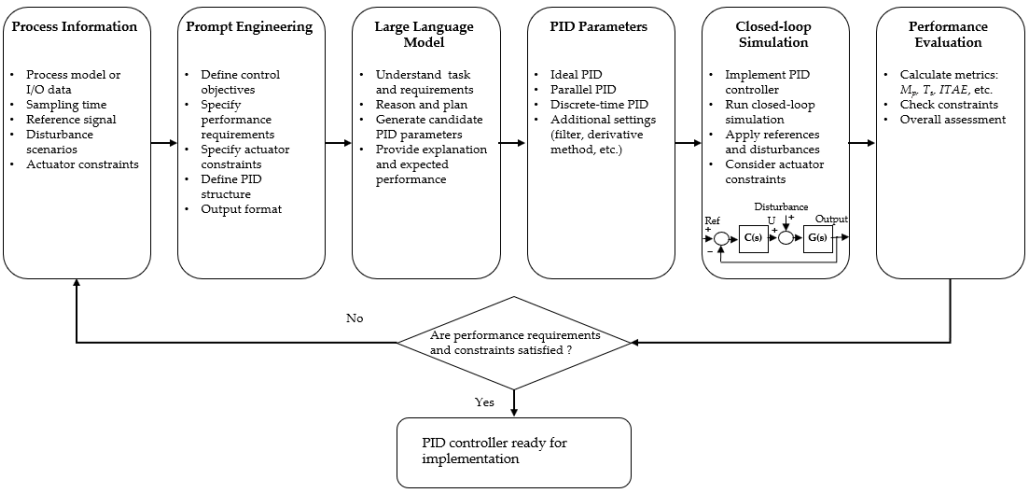



**Figure 1.** Generative AI-assisted PID tuning framework.

The controller design requirements provided to the GenAI systems may be interpreted as a multiobjective tuning problem involving stability, overshoot, settling time, control effort, and disturbance-rejection considerations. However, the present study does not attempt to investigate the internal optimisation mechanisms of the evaluated large language models. Instead, the generated controller parameters are treated as outputs of black-box AI systems and assessed solely through their resulting closed-loop performance. When a response was incomplete or failed to provide controller parameters that satisfied performance requirements and constraints, an additional clarification prompt could be issued. No formal iterative optimisation procedure was implemented.

The framework can be applicable to both continuous-time and discrete-time PID controller implementations.

### 2.2. Problem Formulation

The objective of the proposed methodology is to determine the PI and PID controller parameters that satisfy predefined control performance and actuator (control signal amplitude) constraints. For a process characterised by experimental observations, the controller parameters are selected to satisfy

$$M_p < M_{p,max}, \tag{1}$$

where $M_p$ is an overshoot;

$$e_{ss} = 0, \tag{2}$$

where $e_{ss}$ is a steady-state error;

$$|u(t)| < U_{max}, \tag{3}$$

where $u(t)$ is the controller output, while simultaneously minimising the settling time

$$T_s \rightarrow min. \tag{4}$$

To obtain a quantitative measure of control performance, the Integral Time Weighted Absolute Error (ITAE) criterion was adopted, as it is one of the most widely used performance indices for controller evaluation and tuning [17]:

$$ITAE = \int_0^\infty t|e(t)|dt \tag{5}$$

where *e(t)* denotes the tracking error.

The overall tuning problem can therefore be expressed as minimising *ITAE* subject to (1–3). The selected formulation is consistent with contemporary multi-objective PID tuning methodologies, where controller quality is assessed by simultaneously considering transient performance, steady-state accuracy, and actuator limitations [7,8].

### 2.3. PID Type Controller Structures

The proposed framework considers both the PI and PID controller structures in ideal and parallel form [18].

The continuous-time ideal PI controller is defined as

$$C_{PI}(s) = K_c(1 + \frac{1}{T_i s}), \tag{6}$$

where $K_c$ is the controller gain and $T_i$ is the integral time constant.

The continuous-time parallel PI controller is represented as

$$C_{PI}(s) = K_p + \frac{K_i}{s}, \tag{7}$$

where $K_p$ and $K_i$ denote proportional and integral gains, respectively.

The continuous-time ideal PID controller is defined as

$$C_{PID}(s) = K_c(1 + \frac{1}{T_i s} + T_d s), \tag{8}$$

where $T_d$ is the derivative time constant.

The continuous-time parallel PID controller is represented as

$$C_{PID}(s) = K_p + \frac{K_i}{s} + K_d s, \tag{9}$$

where $K_d$ denotes the derivative gain.

Although the ideal and parallel controller forms are mathematically related through parameter conversion, they were intentionally evaluated as separate prompting scenarios. The objective was to investigate whether the representation of the controller design problem affected the responses generated by ChatGPT and Microsoft Copilot. Therefore, the con-

troller parameters produced for the ideal and parallel forms were treated as independent AI-generated designs rather than converted versions of the same controller.

The PID controllers were implemented in their ideal form according to Equation (8). No derivative filtering was applied, and the derivative term was calculated using the control error signal. The derivative term may improve transient response characteristics such as overshoot and settling time. However, it should not be interpreted as a direct compensation mechanism for dead time. Condition $|u(t)| \leq 5$ was treated as a design requirement and used to evaluate the resulting controller performance.

### 2.4. Experimental Scenarios

To evaluate the capabilities of generative AI in controller synthesis, two experimental scenarios were investigated.

#### 2.4.1. Scenario A: Open-Loop Step Response Data

In the first scenario, the AI model receives only the measured process step-response data. The supplied dataset contains

$$n, Time, Z, PV, \tag{10}$$

where $n$ represents the row number; $Time$ represents time in seconds; $Z$ represents a unit step input signal, and $PV$ is the corresponding process response.

No analytical process model or transfer function is provided to the LLM. Based solely on these experimental observations, the GenAI model is required to determine:

- Continuous-time ideal PI parameters;
- Continuous-time parallel PI parameters;
- Continuous-time ideal PID parameters;
- Continuous-time parallel PID parameters.

This scenario evaluates whether the generative AI model can infer process dynamics directly from open-loop simulated data.

#### 2.4.2. Scenario B: Closed-Loop Operational Data

In the second scenario, the generative AI model receives closed-loop operational data generated during known controller operation. The supplied dataset consists of

$$n, Time, PV, Setpoint, CO, \tag{11}$$

where $Setpoint$ represents the reference signal from the closed-loop system, and $CO$ is the corresponding known controller output.

In addition, the known PI or PID parameters used during the experiment are provided. The process transfer function remains unknown to the AI model.

Based solely on these experimental observations, the generative AI model is required to determine:

- Continuous-time ideal PI parameters;
- Continuous-time parallel PI parameters;
- Continuous-time ideal PID parameters;
- Continuous-time parallel PID parameters.

The objective is to investigate whether generative AI can redesign and improve existing controllers directly from operational closed-loop data. Such AI-assisted engineering workflows represent an emerging research direction in both engineering design and control applications.

### 2.5. Prompt Templates

Prompt engineering is recognised as a critical factor that influences the quality, consistency, and reproducibility of LLM-generated results [10,16]. Therefore, fixed prompt templates were used throughout the paper.

#### 2.5.1. Prompt Template for Scenario A

The controller synthesis procedure started with the determination of the ideal PI controller parameters using the following prompt.

*In the attached file „Time" column represents time; „Z" column represents process input; „PV" column represents process output.*

*Determine continuous-time ideal PI type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PI parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

The interaction with the generative AI model was performed sequentially. After obtaining the continuous-time ideal PI controller parameters, the following prompt was submitted.

*Determine continuous-time parallel PI type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PI parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

After the parallel PI controller parameters had been obtained, the AI model was requested to determine the parameters of the continuous-time ideal PID controller.

*Determine continuous-time ideal PID type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PID parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

After the ideal PID controller parameters had been obtained, the AI model was requested to determine the parameters of the continuous-time parallel PID controller.

*Determine continuous-time parallel PID type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PID parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

#### 2.5.2. Prompt Template for Scenario B

The controller synthesis procedure started with the determination of continuous-time ideal PI controller parameters using the following prompt

*In the attached file „Time" column represents time; „PV" column represents process output in the closed loop; „Setpoint" column represents setpoint; "CO" column represents controller output-process input. Data was obtained with PI type controller which transfer function 1(1 + 1/s).*

*Determine continuous-time ideal PI type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PI parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

The interaction with the generative AI model was performed sequentially. After obtaining the continuous-time ideal PI controller parameters, the following prompt was submitted.

*Determine continuous-time parallel PI type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PI parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

Next, the generative AI model was asked to determine the parameters of the continuous-time ideal PID controller.

*Determine continuous-time ideal PID type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PID parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

Subsequently, the parameters of the continuous-time parallel PID controller were requested.

*Determine continuous-time parallel PID type controller parameters which: guarantee closed-loop stability; guarantee zero steady-state error; enforce the raw controller output $|u(t)| < 5$; guarantee <5% overshoot; minimal settling time; improve disturbance rejection; minimal ITAE. For the designed controller provide: PID parameters; predicted overshoot; predicted settling time; predicted disturbance rejection time; stability assessment; ITAE value and justification of parameter selection.*

### *2.6. Experimental Configuration of GenAI Systems*

The experiments were carried out using ChatGPT (OpenAI, GPT-5.5) and Microsoft 365 Copilot (version number: bizchat.20260701.51.3). For each benchmark process, both systems received identical input information, including the process dataset, task descriptions, and evaluation instructions, to ensure a fair comparison.

For every benchmark process and dataset scenario (Scenario A and Scenario B), a new conversation session was started. Consequently, the open-loop and closed-loop experiments were performed independently, and no information from previous interactions was available in the conversational context. This procedure ensured that both ChatGPT and Microsoft Copilot processed exactly the same information under equivalent experimental conditions.

The experiments were carried out between June 2026 and August 2026 using the publicly available versions of the respective services. Since Generative AI systems are continuously updated by their providers, complete reproducibility of the responses obtained cannot be guaranteed, and future versions of the models may produce different results under identical prompting conditions.

To minimise user-induced variability, the same prompt templates, datasets, and instructions were applied to both AI systems throughout the study. Each prompt was submitted only once, and the first response generated by the system was used for further analysis. Multiple executions of identical prompts were not performed; therefore, response variability and repeatability were not investigated and remain outside the scope of the present work.

The prompts used in the experiments, together with the datasets and Supplementary Materials required to reproduce the study, are provided in the Supplementary Materials.

### 2.7. Validation Procedure

The transfer functions of the benchmark systems were intentionally hidden from the Generative AI systems and used exclusively for validation purposes. Consequently, the AI models performed process identification and controller design solely on the basis of the provided datasets, without access to the actual benchmark process models.

After the LLM-generated controller parameters were obtained, each controller together with the benchmark model was implemented in MATLAB/Simulink (version R2019a) and evaluated using closed-loop simulations. The validation procedure consists of the following:

1. Implementation of the generated PI or PID controllers;
2. Closed-loop simulation using benchmark processes;
3. Evaluation of transient and steady-state performance indicators;
4. Comparison of LLM-predicted and simulated performance indices.

This validation framework enables an objective assessment of Generative AI systems as engineering assistants for process identification, controller tuning, and closed-loop performance prediction.

## 3. Benchmark Processes and Experimental Setup

### 3.1. Benchmark Processes

The proposed Generative AI-assisted framework was evaluated using five benchmark processes representing several classes of industrial dynamics commonly encountered in PID control studies, including first-order, second-order, integrating, and time-delay systems. Such benchmark models are widely used in the process-control literature for the development and evaluation of process identification and PID tuning methods, as they provide representative yet well-understood dynamic behaviours while enabling reproducible comparisons between control-design approaches [19].

The benchmark processes considered in this paper are defined as follows:

$$\text{Process P1} \qquad G_1(s) = \frac{1}{5s + 1}, \tag{12}$$

$$\text{Process P2} \qquad G_2(s) = \frac{e^{-5s}}{5s + 1}, \tag{13}$$

$$\text{Process P3} \qquad G_3(s) = \frac{1}{s(5s + 1)}, \tag{14}$$

$$\text{Process P4} \qquad G_4(s) = \frac{1}{s^2 + 2s + 1}, \tag{15}$$

$$\text{Process P5} \qquad G_5(s) = \frac{e^{-5s}}{s^2 + 2s + 1}. \tag{16}$$

The transfer functions presented above were not provided to the Generative AI systems and were used exclusively during the validation stage. Consequently, AI systems were required to infer the process dynamics solely from the supplied datasets. The original benchmark models were subsequently used to evaluate the quality of the identified process representations and the performance of the generated controllers.

It should be emphasised that the benchmark processes do not represent specific physical plants. Therefore, the control signal $u(t)$ is dimensionless and normalised. The constraint $|u(t)| < 5$ was introduced as a normalised control-signal amplitude limit to generate closed-loop benchmark datasets used in the controller design and prediction tasks. The same limit was applied to all benchmark processes to ensure a consistent and comparable evaluation framework. The selected bound does not correspond to any particular actuator limitation and was used solely for benchmarking purposes.

### 3.2. Benchmark Datasets

The benchmark datasets were generated using numerical simulation of the processes defined in Section 3.1. Consequently, the datasets represent synthetic process data rather than measurements obtained from physical systems. The simulations were performed under idealised conditions and did not include measurement noise, parameter uncertainty, sensor quantisation effects, actuator non-linearities, or other disturbances commonly encountered in industrial environments.

As described in Section 2.4, two independent datasets were created for each benchmark process:

- An open-loop dataset corresponding to Scenario A;
- A closed-loop dataset corresponding to Scenario B.

Consequently, a total of ten datasets were generated and used during the experiments.

All datasets were stored in comma-separated values (CSV) format and provided directly to Generative AI systems. This approach was intended to emulate practical industrial situations in which historical operating data is available, while an accurate analytical process model is unavailable or difficult to obtain.

To ensure transparency and reproducibility of the proposed methodology, all datasets are provided as Supplementary Materials. Separate files are available for the open-loop and closed-loop datasets of the benchmark processes P1–P5, allowing independent verification and replication of the reported experiments.

The benchmark datasets were the only source of process information available to Generative AI systems during the identification and controller design tasks.

#### 3.2.1. Open-Loop Dataset

For Scenario A, each benchmark process was excited by a unit-step signal applied at the process input. The resulting output response was recorded throughout the simulation interval.

The dataset supplied to the GenAI model consisted of the variables provided in (10). A representative data structure is shown in Table 1.

**Table 1.** Example structure of the open-loop dataset supplied to the GenAI model.

| *n* | *Time*, s | *Z* | *PV* |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0.1 | 1 | 0.02 |
| 2 | 0.2 | 1 | 0.04 |
| ... | ... | ... | ... |

Using this information only, the GenAI model was requested to determine the controller parameters according to the procedure described in Section 2.5.1.

#### 3.2.2. Closed-Loop Operational Dataset

The second type corresponds to the closed-loop scenario described in Section 2.4. For each benchmark process, an initial ideal PI or PID controller was implemented and operated under reference tracking conditions. During the experiment, the time, reference signal *Setpoint*, controller output *CO*, and process output *PV* were recorded throughout the simulation interval.

The dataset supplied to the GenAI model consisted of the variables provided in (11). Furthermore, the controller parameters shown in Table 2 and used during data acquisition were provided to the GenAI model.

**Table 2.** The continuous-time ideal PID controller (8) parameters used during data acquisition.

| Process | $K_c$ | $T_i$, s | $T_d$, s |
|---|---|---|---|
| P1 | 1 | 1 | 0 |
| P2 | 1 | 10 | 0 |
| P3 | 1 | 20 | 0 |
| P4 | 1 | 0.5 | 0 |
| P5 | 0.4 | 2.5 | 1 |

Unlike the open-loop scenario, this experiment evaluates whether generative AI can improve an already existing control loop solely on the basis of operational process data.

### 3.3. Validation and Comparative Evaluation

The performance of the controllers generated by the generative AI system was assessed using the benchmark process models introduced in Section 3.1.

Since the analytical process models were intentionally hidden from the GenAI models during the identification tuning stage, the benchmark transfer functions were employed exclusively during the validation phase. Their role was twofold: (i) to assess the quality of the generated controller parameters and (ii) to evaluate the accuracy of the closed-loop performance indicators predicted by the AI systems.

For each benchmark process and tuning scenario, ChatGPT and Microsoft Copilot were requested to provide the following:

- Controller parameters;
- Predicted maximum overshoot;
- Predicted settling time;
- Predicted disturbance rejection time;
- Predicted steady-state error;
- Predicted ITAE.

To provide a conventional benchmark for the evaluation of controller tuning quality, additional PI and PID controllers were designed using the Internal Model Control (IMC) tuning method. The IMC controllers were calculated separately using the process models identified by ChatGPT and Microsoft Copilot. Consequently, the comparison focused on the quality of the generated controller settings while avoiding confounding effects caused by differences in the identified process models.

For each benchmark process, the IMC controller parameters were determined from the corresponding identified model using standard IMC tuning rules. The resulting closed-loop responses were subsequently evaluated using the same performance criteria as those applied to GenAI-generated controllers.

The generated controller parameters were implemented in MATLAB R2019a and validated using MATLAB/Simulink. Simulation-based validation was used because it is one of the most widely used approaches to assess controller performance and compare alternative control strategies [20]. Ideal PI and PID controllers were implemented in Simulink according to the controller structures defined in Section 2.3. The control signal was limited using a saturation block with upper and lower bounds of +5 and −5, respectively. No derivative filtering was applied. Anti-windup compensation was not implemented.

All closed-loop experiments were performed under identical simulation conditions. The simulations were executed using the ode45 solver with default tolerances. The simulation horizon was 200 s and a disturbance of amplitude 0.5 was introduced at $t$ = 100 s. No measurement noise, parameter uncertainty, sensor quantisation, actuator non-linearities, or other nonideal industrial effects were included.

Controller performance was evaluated using the criteria defined in Section 2.2, including maximum overshoot $M_p$, settling time $T_s$, steady-state error $e_{ss}$, integral time-weighted absolute error $ITAE$, disturbance rejection time $T_{DR}$ and the maximum control-signal magnitude (max $u(t)$).

The validation procedure, summarised in Figure 1, enabled the evaluation of two complementary aspects of the design of a Generative AI-assisted controller:

1. The ability of the AI model to generate controller parameters satisfying the specified performance requirements;
2. The accuracy of the AI model in predicting the resulting closed-loop behaviour.

In addition to controller performance evaluation, the discrepancies between the performance indicators predicted by the AI systems and those obtained from MATLAB/Simulink simulations were analysed. This analysis provides insight into the reliability of Generative AI as an engineering assistant capable not only of generating controller parameters but also of estimating expected closed-loop performance prior to implementation.

## 4. Results and Discussion

### *4.1. Process Dynamics Interpretation by Generative AI*

The first stage of the evaluation was focused on the ability of generative AI to interpret process dynamics from the supplied datasets. For each benchmark process, the AI model was requested to identify the dominant dynamic characteristics and justify the proposed controller parameters.

Although analytical transfer functions were not provided to the AI model, the generated explanations indicated that the model could recognise essential process properties, including dominant time constants, dead-time effects, integrating behaviour, and potential oscillatory dynamics.

Two widely used generative AI systems, ChatGPT (OpenAI, GPT-5.5) and Microsoft Copilot, were selected for analysis. Both systems are based on large language model technology and have recently attracted attention as potential tools for engineering design, modelling, and control system applications [11,12,21,22].

Subsequently, the inferred process characteristics were compared with the actual benchmark models introduced in Section 3.1.

#### 4.1.1. Open-Loop Dataset Scenario

The first experiment investigated the ability of generative AI systems to infer process dynamics directly from open-loop step-response data. For each benchmark process, only the measured time response was supplied to the AI models, while the analytical transfer functions remained unknown.

The transfer functions estimated by ChatGPT and Copilot from the open-loop step response datasets are summarised in Table 3.

Both models successfully captured the dominant dynamics of the first-order process P1 and the first-order plus dead-time process P2, producing transfer functions that closely matched the benchmark models. Greater deviations were observed for the integrating process P3 and the second-order processes P4 and P5, where AI systems generally reduced the description of the process to lower-order approximations. Such simplifications are expected, since multiple low-order models may reproduce similar step-response characteristics over a finite observation interval.

**Table 3.** Benchmark process models and transfer functions inferred from open-loop data.

| Process | Actual Transfer Function | ChatGPT Estimated Transfer Function | Copilot Estimated Transfer Function |
|---|---|---|---|
| P1 | $1/(5s + 1)$ | $1/(5.01s + 1)$ | $1/(5s + 1)$ |
| P2 | $e^{-5s}/(5s + 1)$ | $e^{-4s}/(6s + 1)$ | $e^{-5s}/(5s + 1)$ |
| P3 | $1/(s(5s + 1))$ | $e^{-0.1s}/s$ | $0.98e^{-0.12s}/s$ |
| P4 | $1/(s^2 + 2s + 1)$ | $1.001e^{-0.342s}/(1.584s + 1)$ | $1.001e^{-0.442s}/(1.584s + 1)$ |
| P5 | $e^{-5s}/(s^2 + 2s + 1)$ | $1.0015e^{-5.34s}/(1.585s + 1)$ | $e^{-5s}/(2s + 1)$ |

To quantitatively assess the quality of the model, the fit criterion (FIT) was calculated over the time interval of 0 to 50 s according to [23]

$$FIT = 100\left(1 - \frac{\|y - \hat{y}\|}{\|y - \overline{y}\|}\right)\%, \tag{17}$$

where $y$ denotes the output of the actual (benchmark) model; $\hat{y}$ denotes the output of the identified model; and $\overline{y}$ is the mean value of the benchmark output.

Table 4 summarises the *FIT* values.

**Table 4.** Quantitative comparison of process characteristics estimated from open-loop data.

| Process | ChatGPT *FIT* (%) | Copilot *FIT* (%) |
|---|---|---|
| P1 | 99.84 | 100.00 |
| P2 | 91.41 | 100.00 |
| P3 | 67.21 | 71.16 |
| P4 | 91.36 | 94.31 |
| P5 | 95.88 | 92.74 |

Very good agreement was achieved for the first-order process P1 and the delayed first-order process P2, where both AI systems reproduced the benchmark dynamics with *FIT* values exceeding 90%. Copilot achieved 100% *FIT* values for both processes. On the contrary, identification accuracy decreased considerably for the integrating process P3, indicating the difficulty of inferring integrating behaviour solely from open-loop step-response data.

Despite the structural mismatch between the original and identified models, both AI systems provided accurate approximations of the second-order processes P4 and P5. Although the original second-order dynamics were represented by simplified lower-order models, the resulting *FIT* values remained above 90%, suggesting that the dominant dynamic characteristics were successfully preserved.

In general, the average *FIT* values obtained by ChatGPT and Copilot were 89.14% and 91.64%, respectively. These results indicate that both Generative AI systems were capable of extracting process models that captured the dominant process dynamics and provided a sufficiently accurate basis for subsequent controller design and simulation studies.

#### 4.1.2. Closed-Loop Dataset Scenario

The second experiment evaluated the ability of GenAI systems to infer process dynamics from closed-loop operational data. In contrast to the open-loop scenario, AI systems received not only the process response, but also the reference signal, controller output, and controller parameters used during data acquisition. The transfer functions estimated by ChatGPT and Copilot are summarised in Table 5.

The results obtained show that both AI systems were able to identify the dominant dynamics of the self-regulating processes P1, P2, P4, and P5. Copilot consistently produced

models that were closer to the benchmark dynamics across all investigated processes, particularly for P1, P2, and P5. ChatGPT tended to simplify the process dynamics and, in some cases, introduced artificial delays or substantially different time constants. For the P3 integrating process, neither AI system was able to correctly reconstruct the true structure of the process, and both generated simplified first-order plus dead-time approximations.

**Table 5.** Benchmark process models and transfer functions inferred from closed-loop data.

| Process | Actual Transfer Function | ChatGPT Estimated Transfer Function | Copilot Estimated Transfer Function |
|---|---|---|---|
| P1 | $1/(5s + 1)$ | $1.02\, e^{-0.87s}/(1.18s + 1)$ | $1/(4s + 1)$ |
| P2 | $e^{-5s}/(5s + 1)$ | $1/(5s + 1)$ | $e^{-5s}/(6s + 1)$ |
| P3 | $1/(s(5s + 1))$ | $1.55\, e^{-0.6s}/(4.9s + 1)$ | $4\, e^{-0.2s}/(8.1s + 1)$ |
| P4 | $1/(s^2 + 2s + 1)$ | $1.01/(2.38s + 1)$ | $1.01\, e^{-0.3s}/(2.12s + 1)$ |
| P5 | $e^{-5s}/(s^2 + 2s + 1)$ | $1.04/(7.85s + 1)$ | $e^{-5.2s}/(3.1s + 1)$ |

To quantitatively evaluate the identified models, the fit criterion defined by (17) was calculated over the time interval from 0 to 50 s. The resulting values are summarised in Table 6.

**Table 6.** Quantitative comparison of process characteristics estimated from closed-loop data.

| Process | ChatGPT *FIT* (%) | Copilot *FIT* (%) |
|---|---|---|
| P1 | 33.37 | 83.35 |
| P2 | 42.19 | 90.91 |
| P3 | −70.25 | −56.56 |
| P4 | 82.27 | 86.16 |
| P5 | 57.56 | 81.49 |

The *FIT* values presented in Table 6 reveal noticeable differences between the two GenAI systems. Copilot consistently outperformed ChatGPT for the first-order and delayed first-order processes P1 and P2, achieving *FIT* values of 83.35% and 90.91%, respectively. In comparison, ChatGPT achieved significantly lower values because the estimated models did not accurately reproduce the dominant process time constants and delay characteristics.

For the second-order processes P4 and P5, both systems generated simplified lower-order approximations of the original dynamics. However, the resulting *FIT* values indicate that the dominant transient behaviour was captured with reasonable accuracy. This observation demonstrates that accurate reproduction of the dominant closed-loop dynamics does not necessarily require recovery of the exact process structure. In both cases, Copilot achieved higher *FIT* values than ChatGPT. The P3 integrating process remained a challenge for both AI systems, producing negative *FIT* values. This result indicates that the identified models reproduced the process response worse than the mean-value approximation and therefore failed to capture the essential integrating dynamics of the benchmark process. Process P3 exhibited the highest identification errors among the benchmark systems. This process contains an integrator, whose effect is partially suppressed by the feedback controller during closed-loop operation. As a consequence, the identification data contain less information about the true integrating behaviour of the process than would be available under open-loop conditions. The reduced observability of the integrating dynamics can lead to the estimation of a model that appears more self-regulating than the actual process. Therefore, the larger identification errors observed for P3 are consistent with the combined effects of integrating dynamics and the masking influence of feedback control.

A comparison with the open-loop identification results presented in Section 4.1.1 shows that the closed-loop datasets generally resulted in lower identification accuracy for the investigated Generative AI systems, particularly for processes P1, P2 and P3. This finding suggests that the feedback controller may obscure important process characteristics that are more readily observable in open-loop step-response data. Overall, closed-loop datasets proved more challenging for process identification, especially when integrating dynamics was present. However, Copilot generated reasonably accurate models for processes P1, P2, P4, and P5, whereas ChatGPT achieved satisfactory results primarily for the second-order processes P4 and P5.

### *4.2. PI Controller Synthesis Results*

The next stage of the investigation focused on the ability of generative AI systems to synthesise continuous-time PI controllers based on the inferred process dynamics. For each benchmark process, ChatGPT and Copilot were requested to determine the parameters of the ideal and parallel PI controller structures while satisfying the performance requirements defined in Section 2.2.

The generated controller parameters reflect the different process interpretations discussed in Section 4.1. Consequently, noticeable differences between the tuning recommendations provided by the two AI systems can be observed for several benchmark processes. In particular, the largest discrepancies are associated with processes that exhibit integrating behaviour or second-order dynamics, where the inferred process models differed substantially from the original benchmark systems.

#### 4.2.1. Open-Loop Dataset Scenario

The PI controller parameters generated from open-loop step-response data are presented in Table 7. IMC controller parameters were calculated separately for the process models identified by ChatGPT and Microsoft Copilot.

**Table 7.** Comparison of AI-generated and IMC-based continuous-time ideal (6) and parallel (7) PI controller parameters obtained from open-loop data.

| Process | Identification Source | Tuning Method | $K_c$ | $T_i$, s | $K_p$ | $K_i$ |
|---|---|---|---|---|---|---|
| P1 | ChatGPT | IMC | 1.0 | 5.01 | 1.0 | 0.1996 |
| P1 | ChatGPT | ChatGPT | 4.95 | 5.0 | 4.95 | 0.99 |
| P1 | Copilot | IMC | 1.0 | 5.0 | 1.0 | 0.2 |
| P1 | Copilot | Copilot | 4.9 | 5.0 | 4.9 | 0.98 |
| P2 | ChatGPT | IMC | 0.75 | 6.0 | 0.75 | 0.125 |
| P2 | ChatGPT | ChatGPT | 0.75 | 6.0 | 0.75 | 0.125 |
| P2 | Copilot | IMC | 0.5 | 5.0 | 0.5 | 0.1 |
| P2 | Copilot | Copilot | 0.5 | 5.0 | 0.5 | 0.1 |
| P3 | ChatGPT | IMC | 3.33 | 0.4 | 3.33 | 8.33 |
| P3 | ChatGPT | ChatGPT | 1.67 | 2.4 | 1.65 | 0.69 |
| P3 | Copilot | IMC | 2.834 | 1.44 | 2.834 | 1.968 |
| P3 | Copilot | Copilot | 0.91 | 4.48 | 0.91 | 0.203 |
| P4 | ChatGPT | IMC | 2.314 | 1.584 | 2.314 | 1.461 |
| P4 | ChatGPT | ChatGPT | 2.32 | 1.58 | 2.32 | 1.47 |
| P4 | Copilot | IMC | 0.781 | 1.584 | 0.781 | 0.493 |
| P4 | Copilot | Copilot | 1.8 | 1.6 | 1.8 | 1.125 |
| P5 | ChatGPT | IMC | 0.148 | 1.585 | 0.1482 | 0.0935 |
| P5 | ChatGPT | ChatGPT | 0.148 | 1.585 | 0.148 | 0.0935 |
| P5 | Copilot | IMC | 0.2 | 2.0 | 0.2 | 0.1 |
| P5 | Copilot | Copilot | 0.2 | 2.0 | 0.2 | 0.1 |

The results indicate that both AI systems generated physically meaningful controller parameters for all benchmark processes. For the first-order process P1 and the first-order plus dead-time process P2, the controller parameters proposed by ChatGPT and Copilot were generally consistent with the corresponding IMC tuning recommendations. In several cases, including P2 and P5, the AI-generated parameters were identical or nearly identical to the IMC-derived values, suggesting that the systems successfully inferred tuning relationships consistent with established control engineering practice.

The largest discrepancies between the AI-generated and IMC-derived controller parameters were observed for the integrating process P3. Both ChatGPT and Copilot proposed substantially smaller controller gains and larger integral times than those obtained using IMC tuning. These differences are consistent with the lower identification accuracy reported for P3 in Section 4.1.1 and indicate that the integrating dynamics remained a challenge for the investigated Generative AI systems.

For the second-order processes P4 and P5, both AI systems generated PI controller settings that were broadly consistent with the corresponding IMC solutions despite being derived from simplified process models. This observation suggests that accurate controller tuning may still be obtained even when the identified process model does not exactly match the true process structure. Overall, the results indicate that the AI-generated tuning recommendations frequently followed trends similar to those produced by the IMC method, particularly when the process dynamics were identified with high accuracy.

#### 4.2.2. Closed-Loop Dataset Scenario

The PI controller parameters generated using closed-loop operational data are presented in Table 8.

**Table 8.** Comparison of AI-generated and IMC-based continuous-time ideal (6) and parallel (7) PI controller parameters from closed-loop data.

| Process | Identification Source | Tuning Method | $K_c$ | $T_i$, s | $K_p$ | $K_i$ |
|---|---|---|---|---|---|---|
| P1 | ChatGPT | IMC | 0.665 | 1.18 | 0.665 | 0.564 |
| P1 | ChatGPT | ChatGPT | 0.67 | 1.18 | 0.67 | 0.57 |
| P1 | Copilot | IMC | 4.0 | 4.0 | 4.0 | 1.0 |
| P1 | Copilot | Copilot | 3.48 | 1.36 | 3.5 | 2.56 |
| P2 | ChatGPT | IMC | 5.0 | 5.0 | 5.0 | 1.0 |
| P2 | ChatGPT | ChatGPT | 0.65 | 8.0 | 0.8 | 0.1 |
| P2 | Copilot | IMC | 0.6 | 6.0 | 0.6 | 0.1 |
| P2 | Copilot | Copilot | 0.6 | 6.0 | 0.6 | 0.1 |
| P3 | ChatGPT | IMC | 1.04 | 4.9 | 1.04 | 0.211 |
| P3 | ChatGPT | ChatGPT | 2.05 | 4.8 | 2.05 | 0.427 |
| P3 | Copilot | IMC | 0.244 | 8.1 | 0.244 | 0.0301 |
| P3 | Copilot | Copilot | 1.8 | 5.0 | 1.15 | 0.5 |
| P4 | ChatGPT | IMC | 4.713 | 2.38 | 4.713 | 1.98 |
| P4 | ChatGPT | ChatGPT | 0.82 | 2.4 | 0.82 | 0.34 |
| P4 | Copilot | IMC | 3.5 | 2.12 | 3.5 | 1.65 |
| P4 | Copilot | Copilot | 2.1 | 2.12 | 2.1 | 0.99 |
| P5 | ChatGPT | IMC | 0.9615 | 7.85 | 0.9615 | 0.1225 |
| P5 | ChatGPT | ChatGPT | 0.22 | 5.5 | 0.22 | 0.04 |
| P5 | Copilot | IMC | 0.298 | 3.1 | 0.298 | 0.096 |
| P5 | Copilot | Copilot | 0.6 | 5.7 | 0.6 | 0.105 |

Compared with the open-loop scenario, the controller parameters obtained from closed-loop datasets exhibited substantially larger variations between AI-generated and IMC-derived tuning results. This behaviour is consistent with the lower process-

identification accuracy reported in Section 4.1.2, since the controller synthesis procedure relied directly on the identified process models.

For some benchmark processes, particularly P1 (ChatGPT) and P2 (Copilot), the AI-generated controller parameters were identical or very close to the corresponding IMC tuning recommendations. This result suggests that when the identified process model adequately captured the dominant process dynamics, Generative AI systems were capable of reproducing tuning trends consistent with established control engineering practice.

The largest discrepancies between the AI-generated and IMC-derived controller parameters were observed for processes P3 and P4. In particular, substantial differences were obtained in controller gain and integral time for the integrating process P3, reflecting the poor identification results reported in Table 6. These findings indicate that identification errors propagated directly into the tuning stage and resulted in controller parameters that deviated significantly from the classical recommendations of IMC.

Overall, the closed-loop datasets produced less consistent tuning recommendations than the corresponding open-loop datasets. However, both ChatGPT and Copilot continued to generate physically meaningful controller parameters for all benchmark processes, enabling further evaluation through closed-loop simulation studies.

### *4.3. PID Controller Synthesis Results*

To further improve transient performance and disturbance rejection capability, both GenAI systems were subsequently requested to determine continuous-time ideal and parallel PID controller parameters. In contrast to the PI controller design discussed in Section 4.2, the introduction of derivative action provides an additional degree of freedom that may improve response speed and reduce overshoot, particularly for processes exhibiting dead time, oscillatory dynamics, or significant dynamic lag.

The controller parameters were analysed with respect to the identified process models and the imposed design requirements. Particular attention was paid to differences between the tuning strategies adopted by ChatGPT and Copilot, as these differences can significantly influence the resulting closed-loop performance.

#### 4.3.1. Open-Loop Dataset Scenario

The PID controller parameters generated from the open-loop step response data are summarised in Table 9.

The PID controller parameters obtained from the open-loop datasets reveal noticeable differences between the AI-generated tuning recommendations and the corresponding IMC-derived controller settings. For the self-regulating processes P1 and P2, both ChatGPT and Copilot generated physically meaningful PID parameters; however, substantial deviations from the IMC solutions were observed, particularly in the proportional gains and derivative time constants.

The largest discrepancies between the AI-generated and IMC-derived controllers were observed for the integrating process P3. Both AI systems proposed significantly smaller controller gains and substantially different integral and derivative settings than those obtained using IMC tuning. These differences are consistent with the reduced identification accuracy reported for P3 in Section 4.1.1 and suggest that the inferred process dynamics had a strong influence on the resulting controller parameters.

For the second-order processes P4 and P5, both AI systems generated PID settings that remained broadly consistent with the characteristics of the identified models. Although the controller parameters did not exactly match the IMC recommendations, the selected proportional, integral, and derivative actions reflected reasonable control design choices for systems containing higher-order dynamics and time delays.

Overall, the results indicate that both ChatGPT and Copilot considered the derivative action beneficial to improving transient-response characteristics. However, the generated PID parameters frequently differed from the corresponding IMC solutions, indicating that the AI systems relied on tuning heuristics that were not always consistent with classical model-based tuning rules.

**Table 9.** Comparison of AI-generated and IMC-based continuous-time ideal (8) and parallel (9) PID controller parameters from open-loop data.

| Process | Identification Source | Tuning Method | $K_c$ | $T_i$, s | $T_d$, s | $K_p$ | $K_i$ | $K_d$ |
|---|---|---|---|---|---|---|---|---|
| P1 | ChatGPT | IMC | 1.0 | 5.01 | 0.0 | 1.0 | 0.1996 | 0.0 |
| P1 | ChatGPT | ChatGPT | 4.7 | 5.0 | 0.8 | 4.7 | 0.94 | 3.76 |
| P1 | Copilot | IMC | 1.0 | 5.0 | 0.0 | 1.0 | 0.2 | 0.0 |
| P1 | Copilot | Copilot | 4.9 | 5.0 | 0.5 | 4.9 | 0.98 | 2.45 |
| P2 | ChatGPT | IMC | 1.333 | 8.0 | 1.5 | 1.333 | 0.1667 | 2.0 |
| P2 | ChatGPT | ChatGPT | 1.1 | 6.0 | 1.5 | 1.1 | 0.183 | 1.65 |
| P2 | Copilot | IMC | 1.0 | 7.5 | 1.67 | 1.0 | 0.133 | 1.667 |
| P2 | Copilot | Copilot | 0.6 | 6.0 | 0.0833 | 0.6 | 0.1 | 0.05 |
| P3 | ChatGPT | IMC | 5.27 | 0.54 | 0.084 | 4.9 | 8.17 | 0.42 |
| P3 | ChatGPT | ChatGPT | 2.2 | 2.2 | 0.12 | 2.2 | 1.0 | 0.26 |
| P3 | Copilot | IMC | 4.94 | 0.56 | 0.06 | 4.94 | 8.82 | 0.296 |
| P3 | Copilot | Copilot | 1.25 | 3.2 | 0.35 | 1.25 | 0.391 | 0.438 |
| P4 | ChatGPT | IMC | 3.418 | 1.755 | 0.154 | 3.418 | 1.948 | 0.526 |
| P4 | ChatGPT | ChatGPT | 2.8 | 1.5 | 0.28 | 2.85 | 1.9 | 0.75 |
| P4 | Copilot | IMC | 1.212 | 1.805 | 0.194 | 0.999 | 0.553 | 0.194 |
| P4 | Copilot | Copilot | 2.7 | 2.08 | 0.148 | 2.7 | 1.3 | 0.4 |
| P5 | ChatGPT | IMC | 0.531 | 4.255 | 0.995 | 0.5304 | 01247 | 0.5275 |
| P5 | ChatGPT | ChatGPT | 0.185 | 1.585 | 0.4 | 0.185 | 0.117 | 0.074 |
| P5 | Copilot | IMC | 0.6 | 4.5 | 1.11 | 0.6 | 0.133 | 0.667 |
| P5 | Copilot | Copilot | 0.24 | 2.0 | 0.5 | 0.24 | 0.12 | 0.12 |

#### 4.3.2. Closed-Loop Dataset Scenario

The PID controller parameters generated from closed-loop operational data are presented in Table 10.

Compared with the open-loop tuning results, the PID controllers obtained from the closed-loop datasets exhibited substantially greater variability. This observation is consistent with the lower process identification accuracy reported in Section 4.1.2, since the controller design procedure relied directly on the identified process models.

The comparison between the AI-generated and IMC-derived controller parameters reveals considerable differences for several benchmark processes. For some cases, such as P1 identified by ChatGPT and P5 identified by Copilot, the controller parameters remained broadly consistent with the corresponding IMC tuning recommendations. In contrast, substantial deviations from the IMC solutions were observed for processes P2 and P3, particularly in the proportional gains and derivative-time settings.

The integrating process P3 again produced the largest discrepancies between the AI-generated and IMC-derived controllers. This result is consistent with the poor identification accuracy reported in Table 6 and suggests that the model identification errors propagated directly into the controller design stage. Similar effects can be observed for P2, where inaccurate estimation of process dynamics resulted in controller settings that differed significantly from the corresponding IMC recommendations.

Overall, the results indicate that the tuning recommendations obtained from closed-loop datasets were less consistent than those derived from open-loop data. However, both ChatGPT and Copilot generated physically meaningful PID controller parameters for all

benchmark processes, enabling further evaluation through closed-loop simulation and performance analysis studies.

**Table 10.** Comparison of AI-generated and IMC-based continuous-time ideal (8) and parallel (9) PID controller parameters from closed-loop data.

| Process | Identification Source | Tuning Method | $K_c$ | $T_i$, s | $T_d$, s | $K_p$ | $K_i$ | $K_d$ |
|---|---|---|---|---|---|---|---|---|
| P1 | ChatGPT | IMC | 1.213 | 1.615 | 0.318 | 1.213 | 0.751 | 0.386 |
| P1 | ChatGPT | ChatGPT | 0.82 | 1.2 | 0.26 | 0.82 | 0.683 | 0.213 |
| P1 | Copilot | IMC | 4.0 | 4.0 | 0.25 | 4.0 | 1.0 | 0.0 |
| P1 | Copilot | Copilot | 4.0 | 1.6 | 0.55 | 4.2 | 2.63 | 2.6 |
| P2 | ChatGPT | IMC | 5.0 | 5.0 | 0.0 | 5.0 | 1.0 | 0.0 |
| P2 | ChatGPT | ChatGPT | 1.1 | 6.11 | 0.636 | 1.15 | 0.2 | 0.65 |
| P2 | Copilot | IMC | 1.133 | 8.5 | 1.765 | 1.133 | 0.133 | 2.0 |
| P2 | Copilot | Copilot | 0.75 | 6.0 | 1.5 | 0.75 | 0.125 | 1.125 |
| P3 | ChatGPT | IMC | 0.976 | 8.2 | 0.099 | 0.976 | 0.119 | 0.096 |
| P3 | ChatGPT | ChatGPT | 2.45 | 4.2 | 0.85 | 2.45 | 0.583 | 2.08 |
| P3 | Copilot | IMC | 3.73 | 5.2 | 0.283 | 3.73 | 0.717 | 1.06 |
| P3 | Copilot | Copilot | 1.8 | 3.0 | 0.667 | 1.8 | 0.6 | 1.2 |
| P4 | ChatGPT | IMC | 4.713 | 2.38 | 0.0 | 4.713 | 1.98 | 0.0 |
| P4 | ChatGPT | ChatGPT | 1.35 | 2.4 | 0.42 | 1.35 | 0.563 | 0.567 |
| P4 | Copilot | IMC | 4.99 | 2.27 | 0.14 | 4.99 | 2.2 | 0.7 |
| P4 | Copilot | Copilot | 3.0 | 2.12 | 0.2 | 3.0 | 1.42 | 0.6 |
| P5 | ChatGPT | IMC | 0.9615 | 7.85 | 0.0 | 0.9615 | 0.1225 | 0.0 |
| P5 | ChatGPT | ChatGPT | 0.28 | 4.8 | 0.55 | 0.28 | 0.058 | 0.154 |
| P5 | Copilot | IMC | 0.731 | 5.7 | 1.41 | 0.731 | 0.128 | 1.034 |
| P5 | Copilot | Copilot | 0.7 | 6.0 | 2.0 | 0.7 | 0.1167 | 1.4 |

### *4.4. Closed-Loop Performance Evaluation*

The closed-loop performance of the controllers proposed by ChatGPT and Microsoft Copilot was evaluated using MATLAB/Simulink simulations. The controller parameters generated by the AI systems were applied to the benchmark process models (12–16), and the resulting closed-loop responses were analysed in terms of overshoot $M_p$, settling time $T_s$, steady-state error $e_{ss}$, integral of time-weighted absolute error $ITAE$, disturbance rejection time $T_{DR}$, and the maximum control-signal magnitude $|u(t)|$.

The prediction ability of the GenAI systems was assessed by comparing the performance indices predicted by the AI tools with the corresponding values obtained from simulation. The accuracy of the prediction was evaluated using absolute and relative prediction error metrics commonly used in prediction and machine learning studies [24,25].

The relative prediction error was calculated as

$$ER = 100\frac{|P_{pred} - P_{sim}|}{|P_{sim}|}\ \%, \tag{18}$$

where $P_{pred}$ denotes the AI-predicted value, and $P_{sim}$ denotes the simulated value. Cases with $P_{sim} = 0$ were excluded from the relative error analysis because the error is mathematically undefined.

Additionally, the absolute prediction error was evaluated as follows:

$$AE = |P_{pred} - P_{sim}|. \tag{19}$$

Performance indices $M_p$, $T_s$, and $ITAE$ were evaluated from the reference follow-up response prior to disturbance application ($0 \leq t < 100$ s). Settling time was defined as the

time it takes for the process output to enter and remain within a ±2% band around the final value.

To evaluate the performance of disturbance rejection, a step disturbance with an amplitude of 0.5 was introduced at the input of the process at $t$ = 100 s. The disturbance remained active for the remainder of the simulation. The same disturbance scenario was applied to all benchmark processes and controller designs, including the IMC-based controllers, to ensure a consistent comparison framework. The disturbance rejection time $T_{DR}$ was defined as the time it takes for the process output to return to and remain within the ±2% band after the application of the disturbance.

#### 4.4.1. Open-Loop Dataset Scenario

Table 11 presents the performance indices predicted by ChatGPT and Microsoft Copilot when open-loop datasets were provided.

**Table 11.** The predicted performance by GenAI from open-loop data.

| Process | GenAI | Controller | $M_p pred$, % | $T_s pred$, s | $ITAE pred$ | $T_{DR} pred$, s |
|---|---|---|---|---|---|---|
| P1 | ChatGPT | Ideal PI | ≈0–2 | ≈5–6 | ≈5–6 | ≈4–5 |
| P1 | ChatGPT | Parallel PI | ≈1–2 | ≈5.2 | ≈5.4 | ≈4.5 |
| P1 | ChatGPT | Ideal PID | ≈0.5–1.5 | ≈4.2 | ≈4.6 | ≈3.8 |
| P1 | ChatGPT | Parallel PID | ≈1 | ≈4.2 | ≈4.6 | ≈3.8 |
| P1 | Copilot | Ideal PI | ≈0 | ≈4.0 | ≈1.04 | ≈4–5 |
| P1 | Copilot | Parallel PI | ≈0 | ≈4.0 | ≈1.04 | ≈4.5 |
| P1 | Copilot | Ideal PID | ≈0 | ≈3.3 | ≈0.8 | ≈3.0–3.5 |
| P1 | Copilot | Parallel PID | ≈0 | ≈3.3 | ≈0.8 | ≈3.0–3.5 |
| P2 | ChatGPT | Ideal PI | ≈2–4 | ≈18 | ≈20–24 | ≈14–16 |
| P2 | ChatGPT | Parallel PI | ≈3 | ≈18 | ≈22 | ≈14–16 |
| P2 | ChatGPT | Ideal PID | ≈2–3 | ≈12 | ≈15 | ≈9–10 |
| P2 | ChatGPT | Parallel PID | ≈2–3 | ≈12 | ≈15 | ≈9–10 |
| P2 | Copilot | Ideal PI | ≈4.0 | ≈30 | ≈72 | ≈25–30 |
| P2 | Copilot | Parallel PI | ≈4 | ≈30 | ≈72 | ≈25–30 |
| P2 | Copilot | Ideal PID | ≈0.4 | ≈24 | ≈60.5 | ≈20–24 |
| P2 | Copilot | Parallel PID | ≈0.44 | ≈24 | ≈60.5 | ≈20–24 |
| P3 | ChatGPT | Ideal PI | ≈2–3 | ≈3.5–4 | ≈1.2–1.4 | ≈3 |
| P3 | ChatGPT | Parallel PI | ≈3 | ≈3.8 | ≈1.3 | ≈3 |
| P3 | ChatGPT | Ideal PID | ≈2 | ≈2.8 | ≈0.95 | ≈2.2 |
| P3 | ChatGPT | Parallel PID | ≈2–3 | ≈2.7 | ≈0.95 | ≈2.2 |
| P3 | Copilot | Ideal PI | ≈1–3 | ≈5 | ≈5.5–6.5 | ≈4 |
| P3 | Copilot | Parallel PI | ≈2 | ≈5 | ≈6 | ≈3–4 |
| P3 | Copilot | Ideal PID | ≈3 | ≈3.2 | ≈3.5 | ≈2.5 |
| P3 | Copilot | Parallel PID | ≈2.5–3.0 | ≈3.2 | ≈3.5 | ≈2.5 |
| P4 | ChatGPT | Ideal PI | 4.7 | 2.1 | 0.34 | 2.6 |
| P4 | ChatGPT | Parallel PI | 4.7 | 2.1 | 0.34 | 2.6 |
| P4 | ChatGPT | Ideal PID | 2.3 | 1.55 | 0.19 | 1.8 |
| P4 | ChatGPT | Parallel PID | 2.1 | 1.5 | 0.18 | 1.7 |
| P4 | Copilot | Ideal PI | 4.9 | 2.7 | 0.58 | 3.1 |
| P4 | Copilot | Parallel PI | 4.9 | 2.7 | 0.58 | 3.1 |
| P4 | Copilot | Ideal PID | 0.63 | 1.21 | 0.319 | 3.05 |
| P4 | Copilot | Parallel PID | 0.63 | 1.21 | 0.319 | 3.05 |
| P5 | ChatGPT | Ideal PI | ≈2 | ≈27 | Low | ≈22 |
| P5 | ChatGPT | Parallel PI | ≈2 | ≈27 | Low | ≈22 |
| P5 | ChatGPT | Ideal PID | ≈3 | ≈18 | Lower PI | ≈15 |
| P5 | ChatGPT | Parallel PID | ≈3 | ≈18 | Low | ≈15 |
| P5 | Copilot | Ideal PI | ≈4.0 | ≈30.2 | ≈71.7 | ≈30–40 |
| P5 | Copilot | Parallel PI | ≈4.0 | ≈30.2 | ≈71.7 | ≈30–40 |
| P5 | Copilot | Ideal PID | ≈3–4 | ≈22–25 | ≈58 | ≈20–25 |
| P5 | Copilot | Parallel PID | ≈3.5 | ≈23 | ≈58 | ≈22 |

The results indicate that both AI systems were able to identify general performance trends. Lower settling times and *ITAE* values were generally predicted for PID controllers than for PI controllers. Processes P2 and P5 were consistently associated with slower responses and larger performance indices than P1 and P4, while the integrating process P3 generated the largest differences between the predictions of ChatGPT and Copilot.

To quantify the precision of the prediction, the simulated responses were compared with the AI-generated predictions. To enable quantitative error analysis, the GenAI responses were converted into numerical values according to the following rules. Exact numerical values were used directly. The approximate numerical values (e.g., "≈2") were converted to the corresponding numerical value ("2"). Predictions expressed as intervals (e.g., "≈5–6") were represented by the midpoint of the interval. Qualitative descriptions such as "Low" or comparative statements (e.g., "Lower than the original controller") were not converted into numerical values and were excluded from numerical error calculations. These rules were applied consistently throughout the study.

The resulting prediction errors are summarised in Table 12.

**Table 12.** Prediction errors from open-loop data.

| Process | GenAI | Controller | $M_pAE$ | $T_sAE$ | *ITAE AE* | $T_{DR}AE$ | $M_pER$ | $T_sER$ | *ITAE ER* | $T_{DR}ER$ |
|---|---|---|---|---|---|---|---|---|---|---|
| P1 | ChatGPT | PI | 1.00 | 1.36 | 2.04 | 6.22 | N/A | 32.80 | 58.83 | 58.02 |
| | ChatGPT | PID | 4.39 | 10.33 | 3.75 | 7.94 | 81.45 | 71.10 | 44.91 | 67.62 |
| | Copilot | PI | 0.06 | 0.15 | 2.36 | 6.26 | 100.00 | 3.54 | 69.38 | 58.17 |
| | Copilot | PID | 3.75 | 8.91 | 5.35 | 8.56 | 100.00 | 72.97 | 87.00 | 72.48 |
| P2 | ChatGPT | PI | 8.86 | 16.91 | 44.16 | 13.00 | 74.71 | 48.44 | 66.75 | 46.43 |
| | ChatGPT | PID | 27.62 | 18.78 | 46.91 | 14.34 | 91.70 | 61.01 | 75.77 | 60.15 |
| | Copilot | PI | 0.39 | 1.92 | 3.22 | 4.50 | 8.93 | 6.02 | 4.28 | 14.06 |
| | Copilot | PID | 0.60 | 5.67 | 0.78 | 10.00 | 60.09 | 30.96 | 1.27 | 31.25 |
| P3 | ChatGPT | PI | 185.77 | 96.04 | 98552.99 | N/A | 98.67 | 96.24 | 100.00 | N/A |
| | ChatGPT | PID | 2.00 | 96.87 | 86078.40 | N/A | N/A | 97.19 | 100.00 | N/A |
| | Copilot | PI | 59.03 | 94.92 | 5744.21 | N/A | 96.72 | 95.00 | 99.90 | N/A |
| | Copilot | PID | 745.92 | 96.74 | 4850.06 | N/A | 99.60 | 96.80 | 99.93 | N/A |
| P4 | ChatGPT | PI | 10.15 | 3.27 | 1.63 | 1.88 | 68.34 | 60.88 | 82.73 | 41.95 |
| | ChatGPT | PID | 12.32 | 3.24 | 1.99 | 2.16 | 84.27 | 67.64 | 91.27 | 54.59 |
| | Copilot | PI | 4.42 | 3.13 | 1.44 | 1.49 | 47.43 | 53.69 | 71.29 | 32.46 |
| | Copilot | PID | 6.56 | 2.13 | 1.07 | 2.67 | 91.23 | 63.79 | 77.05 | 46.69 |
| P5 | ChatGPT | PI | 2.56 | 5.77 | 83.70 | 13.51 | 56.10 | 17.61 | 99.41 | 38.05 |
| | ChatGPT | PID | 11.51 | 12.44 | 93.03 | 21.07 | 79.33 | 40.87 | 99.57 | 58.41 |
| | Copilot | PI | 0.30 | 1.16 | 1.39 | 3.08 | 7.00 | 3.98 | 1.98 | 9.65 |
| | Copilot | PID | 9.05 | 5.09 | 19.01 | 12.74 | 72.10 | 17.81 | 24.68 | 36.16 |

Table 12 shows that prediction accuracy strongly depended on the type of process. The largest errors were obtained for the integrating process P3, where both AI systems produced settling time and *ITAE* prediction errors close to 100%. On the contrary, comparatively accurate predictions were achieved for the P2 process using the Copilot-generated PI controller, with relative errors below 15% for all investigated performance indices. Similarly, Copilot produced highly accurate predictions for process P5 with the PI controller, where the relative errors in overshoot, settling time, ITAE, and disturbance rejection time remained below 10%.

Although both AI systems were generally able to capture qualitative closed-loop performance trends, substantial discrepancies between predicted and simulated values were observed in many cases. These results indicate that quantitative performance prediction remains a challenging task for current Generative AI systems, particularly when integrating dynamics is present.

#### 4.4.2. Closed-Loop Dataset Scenario

Table 13 presents the controller performance predicted by ChatGPT and Copilot using closed-loop datasets.

**Table 13.** The predicted performance by GenAI from closed-loop data.

| Process | GenAI | Controller | $M_p pred$, % | $T_s pred$, s | *ITAEpred* | $T_{DR} pred$, s |
|---|---|---|---|---|---|---|
| P1 | ChatGPT | Ideal PI | ≈2–4 | ≈5–6 | Lower than original controller | ≈4–5 |
| P1 | ChatGPT | Parallel PI | ≈3.2 | ≈5.4 | ≈2.9 | ≈4.3 |
| P1 | ChatGPT | Ideal PID | ≈2.1 | ≈3.4 | ≈1.85 | ≈2.8 |
| P1 | ChatGPT | Parallel PID | ≈2.0 | ≈3.3 | ≈1.8 | ≈2.7 |
| P1 | Copilot | Ideal PI | 4.6 | 7.1 | ≈2.3 | ≈7 |
| P1 | Copilot | Parallel PI | 4.6 | 7.1 | ≈2.3 | ≈7 |
| P1 | Copilot | Ideal PID | 2.8 | 5.3 | ≈1.5 | ≈5 |
| P1 | Copilot | Parallel PID | 1.5 | 5.0 | ≈1.2 | ≈5 |
| P2 | ChatGPT | Ideal PI | 3.8 | 22 | ≈20.5 | 18 |
| P2 | ChatGPT | Parallel PI | 4.1 | 20 | ≈18.7 | 16 |
| P2 | ChatGPT | Ideal PID | 2.3 | 12 | ≈10.6 | 9 |
| P2 | ChatGPT | Parallel PID | 2.0 | 11 | ≈9.8 | 8 |
| P2 | Copilot | Ideal PI | ≈4.1 | ≈30 | ≈72 | ≈20 |
| P2 | Copilot | Parallel PI | ≈4.1 | ≈30 | ≈71.7 | ≈20 |
| P2 | Copilot | Ideal PID | ≈2.5 | ≈18 | ≈48 | ≈12 |
| P2 | Copilot | Parallel PID | 2.5 | 18 | ≈48 | 12 |
| P3 | ChatGPT | Ideal PI | 3.7 | 11.4 | 9.1 | 7.6 |
| P3 | ChatGPT | Parallel PI | 3.7 | 11.4 | 9.1 | 7.6 |
| P3 | ChatGPT | Ideal PID | 2.8 | 8.6 | 6.5 | 5.2 |
| P3 | ChatGPT | Parallel PID | 2.8 | 8.6 | 6.5 | 5.2 |
| P3 | Copilot | Ideal PI | ≈3–4 | ≈8–10 | ≈10–12 | ≈6–8 |
| P3 | Copilot | Parallel PI | 4.6 | 11–12 | ≈9–11 | 8–10 |
| P3 | Copilot | Ideal PID | ≈3 | ≈8–9 | ≈6–8 | ≈5–7 |
| P3 | Copilot | Parallel PID | ≈3.6 | ≈8.5 | ≈7 | ≈6 |
| P4 | ChatGPT | Ideal PI | 2.8 | 14 | ≈4.0 | 18 |
| P4 | ChatGPT | Parallel PI | 2.8 | 14 | ≈4.0 | 18 |
| P4 | ChatGPT | Ideal PID | 1.5 | 8.5 | ≈2.3 | 10 |
| P4 | ChatGPT | Parallel PID | 1.5 | 8.5 | ≈2.3 | 10 |
| P4 | Copilot | Ideal PI | ≈0–1 | ≈3.2 | ≈0.78 | ≈3–4 |
| P4 | Copilot | Parallel PI | ≈0–1 | ≈3.2 | ≈0.78 | ≈3–4 |
| P4 | Copilot | Ideal PID | ≈3 | ≈2.0 | ≈0.45 | ≈2.0–2.5 |
| P4 | Copilot | Parallel PID | ≈3 | ≈2.0 | ≈0.45 | ≈2.0–2.5 |
| P5 | ChatGPT | Ideal PI | 3–5 | 22–26 | ≈38–42 | 14–18 |
| P5 | ChatGPT | Parallel PI | ≈3.8 | ≈23 | ≈40 | ≈15 |
| P5 | ChatGPT | Ideal PID | ≈2.8 | ≈17 | ≈28 | ≈10 |
| P5 | ChatGPT | Parallel PID | ≈2.8 | ≈17 | ≈28 | ≈10 |
| P5 | Copilot | Ideal PI | ≈2.6 | ≈32 | ≈66 | ≈12–15 |
| P5 | Copilot | Parallel PI | ≈2,6 | ≈32 | ≈66 | ≈12–15 |
| P5 | Copilot | Ideal PID | ≈0.9 | ≈20.5 | ≈44 | ≈8–10 |
| P5 | Copilot | Parallel PID | ≈0.94 | ≈20.5 | ≈44.3 | ≈8–10 |

Table 13 shows that both AI systems continued to predict lower settling times, *ITAE* values, and disturbance rejection times for the PID controllers than for the corresponding PI controllers. Compared to the open-loop scenario, the predicted performance indices changed noticeably for several processes, particularly for the integrating process P3 and the delayed process P5. In general, the predicted performance indices differed considerably between the two AI systems, particularly for the processes P3, P4, and P5, indicating different interpretations of the available closed-loop information. However, the actual accuracy

of these predictions can only be assessed through comparison with the MATLAB/Simulink simulation results.

The corresponding prediction errors are presented in Table 14.

**Table 14.** Prediction errors from closed-loop data.

| Process | GenAI | Controller | $M_pAE$ | $T_sAE$ | *ITAE AE* | $T_{DR}AE$ | $M_pER$ | $T_sER$ | *ITAE ER* | $T_{DR}ER$ |
|---|---|---|---|---|---|---|---|---|---|---|
| P1 | ChatGPT | PI | 13.75 | 17.45 | 17.39 | 12.37 | 82.09 | 76.03 | 77.67 | 73.32 |
| | ChatGPT | PID | 17.31 | 18.33 | 18.46 | 13.28 | 89.18 | 84.35 | 90.89 | 82.58 |
| | Copilot | PI | 10.30 | 0.02 | 0.76 | 2.45 | 69.13 | 0.21 | 24.80 | 53.79 |
| | Copilot | PID | 15.76 | 3.46 | 4.48 | 0.04 | 84.92 | 39.47 | 74.90 | 0.74 |
| P2 | ChatGPT | PI | 3.80 | 23.18 | 104.41 | 34.00 | N/A | 51.30 | 83.59 | 65.38 |
| | ChatGPT | PID | 29.66 | 27.08 | 83.57 | 19.00 | 92.80 | 69.29 | 88.74 | 67.86 |
| | Copilot | PI | 2.87 | 11.95 | 9.85 | 12.00 | 234.09 | 66.17 | 15.86 | 37.50 |
| | Copilot | PID | 2.66 | 16.29 | 16.60 | 20.00 | 51.52 | 47.50 | 25.70 | 62.50 |
| P3 | ChatGPT | PI | 257.56 | 88.55 | 4033.56 | N/A | 98.58 | 88.59 | 99.77 | N/A |
| | ChatGPT | PID | 53.65 | 7.96 | 16.96 | 5.79 | 95.04 | 48.06 | 72.30 | 52.67 |
| | Copilot | PI | 3.50 | 90.24 | 3166.67 | N/A | N/A | 90.93 | 99.65 | N/A |
| | Copilot | PID | 82.00 | 59.76 | 228.06 | 40.75 | 96.47 | 87.55 | 97.02 | 87.17 |
| P4 | ChatGPT | PI | 2.79 | 0.34 | 6.52 | 3.39 | 34472.87 | 2.48 | 61.97 | 23.20 |
| | ChatGPT | PID | 1.44 | 2.21 | 0.99 | 0.39 | 2615.78 | 35.10 | 30.04 | 3.76 |
| | Copilot | PI | 4.39 | 2.75 | 1.42 | 3.79 | 89.78 | 46.22 | 64.55 | 51.99 |
| | Copilot | PID | 3.89 | 1.46 | 1.11 | 2.94 | 56.47 | 42.18 | 71.09 | 56.65 |
| P5 | ChatGPT | PI | 4.00 | 56.61 | 499.18 | 70.62 | N/A | 70.23 | 92.58 | 81.53 |
| | ChatGPT | PID | 2.80 | 42.94 | 222.44 | 49.90 | N/A | 71.64 | 88.82 | 83.31 |
| | Copilot | PI | 3.67 | 8.91 | 16.60 | 20.66 | 58.55 | 21.79 | 20.10 | 60.49 |
| | Copilot | PID | 10.14 | 30.95 | 66.67 | 36.71 | 91.85 | 60.16 | 60.24 | 80.31 |

Table 14 shows that prediction accuracy strongly depended on both the type of process and the controller structure. The integrating process P3 again produced the largest prediction errors, with settling-time and ITAE relative errors frequently exceeding 90% for both ChatGPT and Copilot. These results indicate that integrating dynamics remains particularly difficult for both AI systems to represent accurately from closed-loop operating data.

The most accurate predictions were generally obtained by Copilot for several PI-controller cases. In particular, for process P1, Copilot achieved a settling time prediction error of only 0.21%, while for process P2 the relative *ITAE* prediction error remained below 16%. Similarly, comparatively small prediction errors were obtained for the P5 process using the Copilot-generated PI controller. These results suggest that accurate quantitative predictions were achievable in selected cases, despite the generally large prediction errors observed in the benchmark set.

Overall, the results indicate that closed-loop datasets did not lead to systematic improvement in prediction accuracy. Although several individual cases exhibited relatively small prediction errors, the average errors remained substantial for both AI systems. Consequently, the generated predictions should be regarded as qualitative estimates of expected controller performance rather than precise quantitative indicators suitable for direct engineering use without simulation-based validation.

#### 4.4.3. Comparative Analysis of Prediction Accuracy

Table 15 summarises the average absolute and relative prediction errors obtained for each AI system and dataset scenario.

**Table 15.** Summary statistics of absolute and relative prediction errors obtained for each AI system and dataset scenario.

| Scenario | GenAI | Statistic | $M_pAE$ | $T_sAE$ | *ITAE AE* | $T_{DR}AE$ | $M_pER$ | $T_sER$ | *ITAE ER* | $T_{DR}ER$ |
|---|---|---|---|---|---|---|---|---|---|---|
| Open-Loop | ChatGPT | Mean | 26.62 | 26.50 | 18490.86 | 10.02 | 79.32 | 59.38 | 81.92 | 53.15 |
| | ChatGPT | Median | 9.51 | 11.39 | 45.54 | 10.47 | 67.44 | 65.76 | 84.50 | 59.29 |
| | Copilot | Mean | 83.01 | 21.98 | 1062.89 | 6.16 | 68.31 | 44.45 | 53.68 | 37.62 |
| | Copilot | Median | 4.09 | 4.11 | 2.79 | 5.38 | 69.57 | 43.03 | 45.35 | 33.54 |
| Closed-Loop | ChatGPT | Mean | 38.68 | 28.46 | 500.35 | 23.19 | 5363.76 | 59.71 | 78.64 | 59.29 |
| | ChatGPT | Median | 8.88 | 20.76 | 94.00 | 13.28 | 95.04 | 69.29 | 88.74 | 67.86 |
| | Copilot | Mean | 13.92 | 22.58 | 351.22 | 15.48 | 92.53 | 50.22 | 55.39 | 54.57 |
| | Copilot | Median | 3.95 | 10.43 | 16.60 | 12.00 | 80.68 | 46.86 | 44.38 | 56.65 |

Table 15 summarises the overall prediction accuracy achieved by ChatGPT and Copilot for both dataset scenarios. For open-loop datasets, Copilot generally produced lower prediction errors than ChatGPT. In particular, the mean relative errors for overshoot, settling time, ITAE, and disturbance rejection time were 68.31%, 44.45%, 53.68%, and 37.62%, respectively, compared with 79.32%, 59.38%, 81.92%, and 53.15% obtained by ChatGPT. The corresponding mean absolute errors were also generally lower for Copilot, especially for *ITAE* prediction, where the mean absolute error was 1062.89 compared to 18490.86 for ChatGPT.

Median statistics were included because the relative error values become highly sensitive when the corresponding reference values approach zero. Consequently, median values provide a more robust representation of typical prediction performance and reduce the influence of extreme observations.

For the closed-loop dataset scenario, the results were less conclusive. Copilot achieved lower mean relative errors for settling time (50.22% vs. 59.71%) and *ITAE* (55.39% vs. 78.64%), while also producing substantially lower mean overshoot relative errors (92.53% vs. 5363.76%). The exceptionally large mean overshoot relative error obtained by ChatGPT was primarily caused by several cases in which the simulated overshoot values were close to zero, resulting in very large percentage errors despite relatively small absolute differences.

In general, the aggregated results indicate that the use of closed-loop datasets did not provide a systematic improvement in prediction accuracy. Although several individual predictions were reasonably accurate, the average errors remained substantial for both AI systems. However, Copilot demonstrated more consistent performance and generally lower prediction errors, particularly for settling time and ITAE, which are among the most important indicators of control quality. These findings suggest that current Generative AI systems are more effective in identifying qualitative performance trends than in providing accurate quantitative predictions of closed-loop control performance. This observation is consistent with recent studies that highlight limitations of large language models in rigorous quantitative reasoning and performance evaluation tasks [26,27].

## 5. Discussion

### *5.1. Process Identification Capability*

The results demonstrate that both ChatGPT and Copilot were able to extract process information from simulated datasets and propose corresponding dynamic models. For the first- and second-order benchmark processes, the AI-generated models provided a sufficient basis for controller design and subsequent simulation-based validation. The process descriptions were generally consistent with the expected dynamic characteristics of the investigated systems.

However, the results also revealed substantial difficulties when integrating dynamics were present. Process P3 consistently yielded the lowest identification accuracy, regardless of the AI tool or dataset type. In both open-loop and closed-loop scenarios, Process P3 exhibited the lowest identification accuracy, indicating substantial difficulties in representing integrating behaviour. As discussed later in Section 5.3, P3 also produced the largest closed-loop performance prediction errors. This observation is consistent with previous studies showing that integrating processes represent a particularly challenging class of systems for process characterisation, identification, and controller tuning due to their non-self-regulating nature [23,28]. This observation suggests that current GenAI systems are more effective in interpreting conventional first- and second-order dynamics than integrating processes whose behaviour is often more difficult to infer from limited datasets.

The results indicate a clear relationship between the quality and the subsequent results of the controller design. Process P3 exhibited the largest identification errors and also produced the largest controller performance prediction errors. This behaviour suggests that the errors introduced during process identification propagated to subsequent tuning and prediction tasks. Since the controller parameters were generated from the identified models rather than the original benchmark processes, any model mismatch directly influenced the resulting controller performance.

However, the relationship was not strictly linear. Several cases demonstrated that reasonable controller performance could still be achieved despite imperfect model identification. On the contrary, accurate model identification did not always guarantee accurate performance predictions. Therefore, the results suggest that prediction represents an additional challenge beyond process identification alone.

Another important observation is that the use of closed-loop data did not systematically improve the quality of the generated models or subsequent performance predictions. Although closed-loop datasets contain additional information about system behaviour under feedback control, the results indicate that tested GenAI systems are not yet able to fully exploit this information to improve modelling accuracy.

These findings support the initial hypothesis that GenAI can be used as an additional tool for process identification and preliminary process analysis. However, generated models should be considered approximate representations of process dynamics and should be verified using established identification methods and simulation-based validation before being employed for engineering decision-making.

### *5.2. Controller Tuning Capability*

Both Generative AI systems successfully generated PI and PID controller parameters that produced stable closed-loop responses for most of the investigated processes. The proposed controller settings enabled subsequent simulation-based validation and provided a practical basis for evaluating the control performance predicted by the AI systems.

The generated controller settings suggest that the evaluated AI systems were able to recognise qualitative relationships between process dynamics and controller performance in several benchmark cases. In many cases, AI-generated PID controllers were predicted to provide lower settling times, lower *ITAE* values, and faster disturbance rejection than the corresponding PI controllers, particularly for processes with delays or more complex dynamic behaviour. The frequent recommendation of PID controllers by both AI systems suggests that derivative action was recognised as beneficial for improving transient response characteristics such as overshoot and settling time. However, the derivative term should not be interpreted as a direct compensation mechanism for process dead time.

The simulation results further demonstrated that the controllers proposed by ChatGPT and Copilot were generally capable of stabilising the investigated benchmark processes.

This finding suggests that Generative AI can provide useful preliminary controller settings without requiring detailed manual tuning procedures.

However, the variability of the performance indices and the large prediction errors observed in several cases indicate that AI-generated controller parameters should be treated as preliminary recommendations rather than as final tuning solutions. Consequently, simulation-based validation remains essential prior to practical implementation. Such validation is particularly important for processes containing integrating dynamics and significant dynamic complexity, where both controller performance and prediction accuracy were found to deteriorate.

### *5.3. Prediction Accuracy of Closed-Loop Performance*

Although both AI systems successfully generated controller parameters and provided performance predictions, the results obtained indicate fundamental limitations in the ability of current generative AI models to estimate quantitative closed-loop performance metrics. Predicting overshoot, settling time, ITAE, and disturbance rejection time requires an accurate understanding of the dynamic interactions between the process and the controller, which is considerably more complex than identifying qualitative process characteristics or proposing controller structures.

The transition from open-loop to closed-loop identification data affected the benchmark processes differently. For Process P1, the use of closed-loop data produced only minor changes in the identified dynamics and consequently had a limited effect on the tuning results. For Process P2, closed-loop data occasionally improved the controller-design outcome, although the improvement was not consistent across all tuning scenarios. The most pronounced degradation was observed for the integrating process P3. During closed-loop operation, the feedback controller partially compensates for the accumulation effect produced by the integrator, reducing the visibility of the underlying integrating dynamics in the identification data. Consequently, identified models contain less information about the true process behaviour, increasing the likelihood of inaccurate controller parameters and performance predictions.

The observed prediction inaccuracies suggest that the AI models relied primarily on generalised relationships learnt from typical control system behaviour, rather than on rigorous dynamic analysis of the specific processes considered in this paper. As a result, the generated predictions were often able to indicate the expected direction of performance changes, such as the potential benefits of derivative action, while exhibiting limited accuracy when estimating the actual numerical values of performance indices.

Another important observation is that providing closed-loop operating data did not lead to a substantial improvement in prediction quality. This finding suggests that limitations are not solely related to the amount of information available to AI systems. Instead, the results indicate that contemporary large-language models still lack the capability to fully reconstruct and quantitatively analyse dynamic system behaviour from simulated process data alone.

Substantial differences between mean and median errors, particularly for overshoot and ITAE predictions, indicate that the arithmetic means were strongly influenced by a small number of extreme observations, mainly associated with the integrating process P3 and cases with near-zero reference overshoot values. This observation highlights the importance of reporting both mean and median statistics when evaluating the quantitative prediction capabilities of Generative AI systems.

From a control engineering perspective, these findings imply that GenAI is currently more suitable as a decision-supporting and knowledge-assistance tool than as a replacement for established analytical, identification, and simulation methods. Such technology

can accelerate early-stage engineering activities by generating preliminary estimates and alternative design options; however, quantitative performance assessment continues to require a dedicated modelling and simulation tool.

### 5.4. Comparison with IMC-Based Controller Tuning

To assess whether the controllers generated by ChatGPT and Microsoft Copilot are competitive with established tuning approaches, an additional comparison with the IMC benchmark was performed. Since two process models were obtained for each benchmark process (identified by ChatGPT and Copilot), the reported IMC values represent the median performance of the corresponding IMC-designed controllers. Similarly, GenAI values represent the median performance of the controllers generated by ChatGPT and Microsoft Copilot.

Most controllers achieved negligible steady-state error. Noticeable steady-state deviations were observed primarily for PI controllers associated with the integrating process P3 and for the closed-loop P2 IMC benchmark.

Table 16 compares the median closed-loop performance achieved by the GenAI-generated and IMC-based controllers. Overall, the results indicate that the tuning recommendations generated by ChatGPT and Microsoft Copilot frequently produced performance comparable to that obtained using the classical IMC tuning approach.

**Table 16.** Median performance comparison of IMC and GenAI-assisted tuning methods.

| Process | Controller | Tuning Method | Median $M_p$, % | Median $T_s$, s | Median Max $\lvert u(t) \rvert$ | Requirements Met |
|---|---|---|---|---|---|---|
| P1 | PI | IMC | 0.0 | 19.8 | 1.0 | Yes |
| P1 | PI | GenAI | 0.1 | 4.1 | 4.95 | Yes |
| P1 | PID | IMC | 0.0 | 19.8 | 1.0 | Yes |
| P1 | PID | GenAI | 4.6 | 13.4 | 4.8 | Yes |
| P2 | PI | IMC | 8.2 | 33.4 | 1.25 | No |
| P2 | PI | GenAI | 8.2 | 33.4 | 1.25 | No |
| P2 | PID | IMC | 19.6 | 33.9 | 1.65 | No |
| P2 | PID | GenAI | 15.6 | 24.6 | 1.45 | No |
| P3 | PI | IMC | 77.0 | 99.6 | >5.0 | No |
| P3 | PI | GenAI | 124.7 | 99.9 | >5.0 | No |
| P3 | PID | IMC | 0.0 | 99.4 | >5.0 | No |
| P3 | PID | GenAI | 374.5 | 99.8 | >5.0 | No |
| P4 | PI | IMC | 7.4 | 5.9 | 1.8 | No |
| P4 | PI | GenAI | 12.1 | 5.6 | 2.25 | No |
| P4 | PID | IMC | 6.9 | 3.45 | 2.35 | No |
| P4 | PID | GenAI | 10.9 | 4.05 | 2.75 | No |
| P5 | PI | IMC | 4.5 | 30.9 | 1.0 | Yes |
| P5 | PI | GenAI | 4.5 | 30.9 | 1.0 | Yes |
| P5 | PID | IMC | 8.6 | 22.9 | 1.25 | No |
| P5 | PID | GenAI | 13.5 | 29.5 | 1.15 | No |

The closest agreement between GenAI and IMC tuning was observed for the PI controllers of the processes P2 and P5, where the median performance indices were identical. These results suggest that when the identified process models adequately captured the dominant process dynamics, AI systems were able to generate controller parameters consistent with established model-based tuning methods.

Larger differences were observed for processes P1, P3, and P4. For process P1, GenAI-generated controllers produced substantially faster responses than the corresponding IMC controllers, although at the expense of significantly increased control effort. The

largest discrepancies occurred for the integrating process P3, where neither IMC-based nor GenAI-based tuning consistently satisfied the imposed design requirements. This finding is consistent with the identification and prediction results presented earlier and further confirms that integrating dynamics remain particularly challenging for both AI-assisted and conventional tuning approaches.

Overall, the comparison demonstrates that GenAI-generated tuning recommendations can often achieve control quality comparable to that obtained using a conventional model-based tuning method. However, the results also show that the quality of the generated controllers remains strongly dependent on the accuracy of the identified process model and may deteriorate substantially for processes with integrating dynamics.

### *5.5. Practical Implications and Limitations*

From a practical perspective, the results demonstrate that Generative AI can support several control-engineering tasks, including preliminary process characterisation, controller structure selection, initial controller tuning, and qualitative performance assessment. Both ChatGPT and Microsoft Copilot were able to generate process models and controller parameters in a short time and without requiring detailed manual analysis. Consequently, such tools may assist engineers during the early stages of control system design and provide useful starting points for further development.

However, the results also indicate that GenAI should currently be regarded as a decision support tool rather than an autonomous engineering design solution, an observation consistent with recent reviews highlighting both the opportunities and limitations of LLM-based engineering design workflows [10]. Although the generated controller parameters frequently produced stable closed-loop responses and, in several cases, achieved performance comparable to that obtained using IMC-based tuning, reliable controller evaluation still required simulation-based validation. Consequently, AI-generated recommendations should be treated as initial design proposals that require further verification using established control engineering methods.

An important limitation of the present paper is the limited number of benchmark processes considered. Although the selected process set included representative first-, delayed-, integrating and second-order dynamics, it does not fully reflect the diversity and complexity of industrial processes. Although this approach ensured consistent and repeatable comparisons between the evaluated GenAI systems, it may not fully represent the challenges of real-world process identification and controller tuning. The performance of GenAI may differ when applied to non-linear, time-varying, multivariable, or highly constrained systems.

Another limitation is related to the use of synthetic benchmark datasets. All datasets were generated under controlled simulation conditions and did not include measurement noise, parameter uncertainty, sensor quantisation effects, actuator non-linearities, or other disturbances commonly encountered in industrial environments. This approach ensured repeatability and fair comparison between the AI systems; however, it may not fully represent the complexity of real industrial data.

Furthermore, only two widely available GenAI systems, ChatGPT and Microsoft Copilot, were investigated. The conclusions obtained in this paper may therefore not be directly transferable to other large-language models or future model generations whose capabilities may differ substantially.

The prompt-driven nature of Generative AI systems represents an additional source of uncertainty. The generated process models, controller parameters, and performance predictions may depend on the wording, structure, and level of detail of the instructions provided to the AI system. Consequently, different prompting strategies may lead to

different engineering recommendations, raising concerns regarding reproducibility and standardisation in AI-assisted workflows.

Another limitation of the present study is that each prompt was evaluated only once. Response variability between repeated interactions was not investigated. Since large language models may produce different outputs for identical prompts, the reported results should be interpreted as observations obtained under a single-run evaluation protocol rather than as a complete characterisation of model behaviour. Future studies should examine the repeatability and statistical variability of the engineering solutions.

Finally, the internal reasoning mechanisms of the evaluated AI systems were not analysed. The generated controller parameters were assessed solely on the basis of identification accuracy, predicted performance, and simulated closed-loop behaviour. Consequently, it remains unclear whether the results obtained were primarily driven by implicit optimisation, pattern recognition, learned tuning heuristics, or other model-specific mechanisms.

In general, the findings suggest that Generative AI can support engineering workflows and preliminary control system design activities. However, conventional modelling, system identification, analytical design methods, and simulation tools remain essential to achieve the level of accuracy, robustness, and reliability required in practical control engineering applications.

### *5.6. Future Research Directions*

Several opportunities for future research can be identified.

First, larger and more diverse benchmark sets should be considered to evaluate the robustness of Generative AI across a wider range of industrial processes. Future studies should include non-linear, time-varying, multivariable, and constrained processes, as well as systems with significant integrating, delayed, and higher-order dynamics, which proved to be the most challenging cases in the present study.

Second, the use of industrial and experimentally obtained datasets should be investigated. The benchmark datasets used in this work were generated under idealised simulation conditions and did not include measurement noise, parameter uncertainty, sensor quantification, or actuator non-linearities. Therefore, future studies should evaluate whether the observed results remain valid when realistic industrial data are used.

Third, future work should investigate the influence of prompt engineering and data representation on process identification, controller tuning, and performance prediction. Since Generative AI systems are sensitive to input formulation, the development of standardised prompting strategies may improve the reproducibility and reliability of AI-assisted control-engineering workflows.

Fourth, future studies should analyse the variability of the response through repeated prompt evaluations. Because the present work adopted a single-run evaluation protocol, additional research is required to quantify the repeatability, robustness, and statistical variability of the generated models, controller parameters, and performance predictions.

Fifth, the capabilities of additional Generative AI systems and specialised engineering copilots should be evaluated. The rapid development of large-language models suggests that substantial improvements in modelling, tuning, and prediction accuracy may be expected in future generations of AI tools. Comparative studies involving a broader range of model architectures would help to determine to what extent the findings of this paper can be generalised.

Sixth, future research should investigate the reasoning and decision-making mechanisms employed by Generative AI systems in engineering tasks. Improved understanding of how large language models generate process models, controller parameters, and per-

formance estimates may help identify the causes of prediction errors and improve the transparency of AI-assisted engineering workflows.

Seventh, more advanced controller structures, including model predictive control (MPC), adaptive control systems, and intelligent control strategies, should be investigated to determine whether Generative AI can support more sophisticated control system design tasks than conventional PI and PID tuning.

Finally, future work should focus on tighter integration between Generative AI, system-identification software, simulation environments, and digital twins. Such hybrid approaches may combine the reasoning capabilities of large language models with the numerical accuracy of model-based engineering tools, which could lead to more reliable and practically applicable engineering decision support systems [29].

## 6. Conclusions

This paper evaluated the capabilities of ChatGPT and Microsoft Copilot for process identification, PI/PID controller tuning, and closed-loop performance prediction using simulated benchmark processes and a common prompting framework. Five benchmark processes representing first-, delayed-, integrating, and second-order dynamics were considered in open-loop and closed-loop dataset scenarios.

The results demonstrated that both AI systems were capable of extracting process information from the provided datasets and generating physically meaningful controller parameters. In most cases, the generated PI and PID controllers produced stable closed-loop responses and provided a suitable basis for simulation-based evaluation. The identified process models generally captured the dominant dynamics of first-order and second-order systems, although significant difficulties were observed for the integrating process.

Comparative analysis revealed that process-identification quality strongly influenced subsequent controller tuning and performance prediction. The largest identification and prediction errors were consistently associated with the integrating process P3, indicating that integrating dynamics remains a challenge for tested Generative AI systems. The results further showed that the use of closed-loop datasets did not provide a systematic improvement in identification quality or prediction accuracy compared with the corresponding open-loop datasets.

Both ChatGPT and Microsoft Copilot were able to generate controller parameters that, in many cases, achieved performance comparable to that obtained using IMC-based tuning. However, substantial discrepancies were observed between the predicted and simulated closed-loop performance indices. Although both AI systems were generally able to identify qualitative performance trends, accurate quantitative prediction of overshoot, settling time, ITAE, and disturbance rejection time remains a significant challenge.

Overall, the results indicate that Generative AI can serve as a useful support tool for preliminary process modelling, controller tuning, and engineering decision support. However, the generated models, controller parameters and performance predictions should be considered preliminary recommendations and verified using established identification methods and simulation-based validation before practical application. While Microsoft Copilot generally achieved lower prediction errors than ChatGPT in the investigated benchmark cases, neither system demonstrated sufficient accuracy to replace conventional modelling and simulation tools for quantitative control-system analysis.

**Supplementary Materials:** The following supporting information can be downloaded at https://www.mdpi.com/article/10.3390/app162010438/s1. File S1: Open-loop and closed-loop datasets used for process identification; File S2: GenAI-generated responses; File S3: MATLAB source code for results evaluation.

**Author Contributions:** Conceptualization, L.B. and V.B.; methodology, L.B.; software, L.B.; validation, L.B., D.A. and O.S.; formal analysis, L.B., D.A. and O.S.; investigation, L.B.; resources, V.B.; data curation, L.B.; writing—original draft preparation, L.B.; writing—review and editing, V.B., D.A. and O.S.; visualisation, L.B.; supervision, V.B.; project administration, V.B.; funding acquisition, V.B. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Research Council of Lithuania (LMTLT) and the Ministry of Education, Science, and Sport of the Republic of Lithuania, agreement No. S-A-UEI-23-1.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The raw data supporting the conclusions of this article is available in Supplementary Materials.

**Acknowledgments:** The authors thank Vidas Raudonis, Head of the Edge Artificial Intelligence and Robotics research group, for his support during the experimental testing of the proposed concept. During the preparation of this manuscript, the author(s) used Writefull for language editing and proofreading of the manuscript. Microsoft 365 Copilot (version number: bizchat.20260701.51.3) and ChatGPT (OpenAI, GPT-5.5) were used as supportive tools for literature exploration, text drafting, result simulation, and interpretation improvement. All AI-generated outputs were critically assessed and validated by the authors. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| AE | Absolute Prediction Error |
| AI | Artificial Intelligence |
| CSV | Comma-Separated Values |
| ER | Relative Prediction Error |
| FIT | Fit Criterion |
| GenAI | Generative Artificial Intelligence |
| IMC | Internal Model Control |
| ITAE | Integral Time-Weighted Absolute Error |
| LLM | Large Language Model |
| PI | Proportional–Integral |
| PID | Proportional–Integral–Derivative |

## References

1. Shamsuzzoha, M.; Raja, G.L. Introductory Chapter: PID-Based Industrial Process Control. In *PID Control for Linear and Nonlinear Industrial Processes*; Shamsuzzoha, M., Raja, G.L., Eds.; IntechOpen: London, UK, 2023. [CrossRef]
2. Åström, K.J.; Hägglund, T. The Future of PID Control. *Control Eng. Pract.* **2001**, *9*, 1163–1175. [CrossRef]
3. Hägglund, T.; Guzmán, J.L. Give Us PID Controllers and We Can Control the World. *IFAC-PapersOnLine* **2024**, *58*, 103–108. [CrossRef]
4. Ziegler, J.G.; Nichols, N.B. Optimum Settings for Automatic Controllers. *J. Dyn. Syst. Meas. Control* **1993**, *115*, 220–222. [CrossRef]
5. Rivera, D.E.; Morari, M.; Skogestad, S. Internal Model Control: PID Controller Design. *Ind. Eng. Chem. Process Des. Dev.* **1986**, *25*, 252–265. [CrossRef]
6. Mazhar, A.; Tanveer, A.; Izhan, M.; Khan, M.Z.T. Robust Control Approaches and Trajectory Planning Strategies for Industrial Robotic Manipulators in the Era of Industry 4.0: A Comprehensive Review. *Eng. Proc.* **2023**, *56*, 75. [CrossRef]
7. Mohindru, P. Review on PID, Fuzzy and Hybrid Fuzzy PID Controllers for Controlling Non-linear Dynamic Behaviour of Chemical Plants. *Artif. Intell. Rev.* **2024**, *57*, 97. [CrossRef]
8. Patil, R.S.; Jadhav, S.P.; Patil, M.D. Review of Intelligent and Nature-Inspired Algorithms-Based Methods for Tuning PID Controllers in Industrial Applications. *J. Robot. Control* **2024**, *5*, 336–360. [CrossRef]

9. Tanveer, M.; Azad, M.M.; Kim, D.; Khalid, S.; Kim, H.S. Generative Design for Engineering Applications: A State-of-the-Art Review. *Arch. Comput. Methods Eng.* **2026**, *33*, 53–79. [CrossRef]
10. Chiarello, F.; Barandoni, S.; Škec, M.M.; Fantoni, G. Generative Large Language Models in Engineering Design: Opportunities and Challenges. In *Proceedings of the Design Society, DESIGN 2024, Dubrovnik, Croatia, 20–23 May 2024*; Cambridge University Press: Cambridge, UK, 2024; pp. 1959–1968. [CrossRef]
11. Kevian, D.; Syed, U.; Guo, X.; Havens, A.; Dullerud, G.; Seiler, P.; Qin, L.; Hu, B. Capabilities of Large Language Models in Control Engineering: A Benchmark Study on GPT-4, Claude 3 Opus, and Gemini 1.0 Ultra. *arXiv* **2024**, arXiv:2404.03647. [CrossRef]
12. Nosrati, K.; Tepljakov, A.; Belikov, J.; Petlenkov, E. When Control Meets Large Language Models: From Words to Dynamics. *Eng. Appl. Artif. Intell.* **2026**, *178*, 115119. [CrossRef]
13. Aghaee, F.; Shaker, H.R. Large Language Models in Sensor-Driven Control Systems: Architectures, Challenges, and Opportunities. *Sensors* **2026**, *26*, 4350. [CrossRef]
14. Yu, Y.; Kim, S.G.; Lee, Y.P.; Cha, J. Toward Intelligent Mechanical Systems: Control and Decision-Making with Large Language Models. *JMST Adv.* **2025**, *7*, 91–97. [CrossRef]
15. Kamenko, I.; Ilic, S.; Congradac, V. LLM-Based PID Controller Optimization. In *Proceedings of the 2025 10th International Conference on Smart and Sustainable Technologies (SpliTech), Bol and Split, Croatia, 16–20 June 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–6. [CrossRef]
16. Peckham, O.; Raines, J.; Bulsink, E.; Goudswaard, M.; Gopsill, J.; Barton, D.; Nassehi, A.; Hicks, B. Artificial Intelligence in Generative Design: A Structured Review of Trends and Opportunities in Techniques and Applications. *Designs* **2025**, *9*, 79. [CrossRef]
17. Martins, F.G. Tuning PID Controllers Using the ITAE Criterion. *Int. J. Eng. Educ.* **2005**, *21*, 867–873.
18. Åström, K.J.; Hägglund, T. *Advanced PID Control*; ISA: Research Triangle Park, NC, USA, 2006.
19. Skogestad, S. Simple Analytic Rules for Model Reduction and PID Controller Tuning. *J. Process Control* **2003**, *13*, 291–309. [CrossRef]
20. Yu, C.C.; Wang, S.; Huang, B. *Control Performance Assessment: Theoretical Analyses and Industrial Practice*; Springer: Cham, Switzerland, 2019. [CrossRef]
21. OpenAI. ChatGPT. Available online: https://chatgpt.com/ (accessed on 27 July 2026).
22. Microsoft. Microsoft 365 Copilot Documentation. Microsoft Learn. 2025. Available online: https://learn.microsoft.com/en-us/microsoft-365/copilot/ (accessed on 3 August 2026).
23. Ljung, L. *System Identification: Theory for the User*, 2nd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 1999.
24. Hewamalage, H.; Ackermann, K.; Bergmeir, C. Forecast Evaluation for Data Scientists: Common Pitfalls and Best Practices. *Data Min. Knowl. Discov.* **2023**, *37*, 788–832. [CrossRef]
25. Rainio, O.; Teuho, J.; Klén, R. Evaluation Metrics and Statistical Tests for Machine Learning. *Sci. Rep.* **2024**, *14*, 6086. [CrossRef]
26. Laskar, M.T.R.; Alqahtani, S.; Bari, M.S.; Rahman, M.; Khan, M.A.M.; Khan, H.; Jahan, I.; Bhuiyan, A.; Tan, C.W.; Parvez, M.R.; et al. A Systematic Survey and Critical Review on Evaluating Large Language Models: Challenges, Limitations, and Recommendations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024), Miami, FL, USA, 12–16 November 2024*; Association for Computational Linguistics: Kerrville, TX, USA, 2024; pp. 13785–13816.
27. Wan, Y.; Wang, W.; Yang, Y.; Yuan, Y.; Huang, J.-T.; He, P.; Jiao, W.; Lyu, M. LogicAsker: Evaluating and Improving the Logical Reasoning Ability of Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024), Miami, FL, USA, 12–16 November 2024*; Association for Computational Linguistics: Kerrville, TX, USA, 2024; pp. 2124–2155.
28. Jorge, C.C.; Jorge, H.N.; Wendell, M.R.; Edalatpanah, S.A.; Shariq, A.B.; Naz, S.; Javier, J.C.; Gabriel, P.E. Novel characterization and tuning methods for integrating processes. *Int. J. Inf. Tecnol.* **2024**, *16*, 1387–1395. [CrossRef]
29. Gebreab, S.; Musamih, A.; Salah, K.; Jayaraman, R.; Boscovic, D. Accelerating Digital Twin Development with Generative AI: A Framework for 3D Modeling and Data Integration. *IEEE Access* **2024**, *12*, 185918–185936. [CrossRef]


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.