LLM-Enhanced Human-in-the-Loop MPC: TCLab Demonstration for Pharma 4.0

Jonathan Pershing*, Nathan Stone*, Joseph Allison*, Darren Whitaker†, John D. Hedengren*
*Department of Chemical Engineering, Brigham Young University, Provo, UT, USA
Emails: {persh, nstone42, joface02, john.hedengren}@byu.edu
†Small Molecule Process Development, Takeda Pharmaceuticals, Cambridge, MA, USA
Email: darren.whitaker@takeda.com

***Abstract*—Industry 4.0 in pharmaceutical manufacturing demands advanced automation and control to handle increasingly specialized drugs and personalized therapies. However, leveraging sophisticated controls like model predictive control (MPC) often requires expertise beyond the typical plant operator. This paper proposes a human-in-the-loop framework where a large language model (LLM) assists operators in configuring and tuning an MPC using natural language commands. The LLM translates high-level objectives (e.g. "speed up the response," "avoid overshoot") into MPC tuning adjustments and provides real-time analysis of controller performance. A human supervisor remains in the loop as a safety net, which is essential in regulated pharma environments to ensure trust and compliance. The concept is demonstrated on a Temperature Control Lab (TCLab) hardware setup using a GEKKO-based MPC. Results show that key control objectives can be modified on the fly via natural language, enabling intuitive operator interaction. Four representative scenarios illustrate how LLM-translated commands affect MPC behavior, and an insight-generation feature provides automated textual analysis of performance. This human-in-the-loop, LLM-assisted MPC approach offers a pathway to deploy advanced control in pharma manufacturing while preserving human oversight.**

***Index Terms*—Industry 4.0, human-in-the-loop control, large language models (LLMs), model predictive control (MPC), natural language interfaces, advanced process control (APC)**

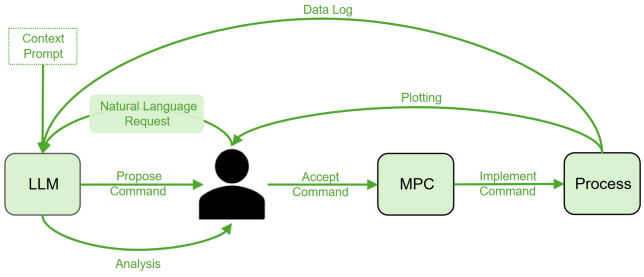

Fig. 1. Information flow in the LLM-enhanced Human-in-the-loop MPC framework. The cycle begins with an operator stating an objective in natural language. The LLM interprets the request, consults the context prompt and process data, and proposes corresponding tuning commands. The operator reviews and approves these commands before they are passed to the MPC. The MPC then implements the commands on the process, while sensor data, logs, and plots are fed back to both the operator and the LLM for analysis. This loop demonstrates how operator intent is translated into real-time MPC adjustments.

I. Introduction

Industry 4.0 envisions smart, autonomous manufacturing systems with tight integration of advanced control, data analytics, and human operators. In pharmaceutical manufacturing ("Pharma 4.0"), processes are becoming more complex and highly customized, such as small-batch personalized medicines. This increases the need for sophisticated control strategies (to maintain quality and efficiency) even as operators might not be experts in control theory. Model predictive control (MPC) is a prime candidate for advanced process control in pharma due to its ability to handle multivariate constraints and anticipate future behavior [1], [2]. Yet, tuning MPC or adjusting its objectives typically requires specialized knowledge. Bridging this gap motivates the use of large language models (LLMs) as intuitive interfaces between human operators and control systems.

LLMs have demonstrated exceptional capability to interpret high-level human intentions in natural language and translate them into actionable commands [3]. Recent research has explored integrating LLMs into control loops across domains such as robotics, autonomous driving, and embodied agents [4], [5]. An LLM can serve as an interpreter of operator instructions, effectively acting as a high-level planner or policy generator in complex control tasks [3]. For instance, instead of manually re-tuning controller gains, an operator could simply state a goal ("make the response faster without overshoot") and have the LLM reformulate the MPC configuration accordingly. Prior work shows the promise of this approach: Luo et al. (2024) use an LLM to generate MPC control code from natural-language task descriptions, pairing it with a numeric solver to enforce physical constraints [6]. In robotics, the SayCan framework combines an LLM's high-level reasoning with low-level skill affordances so that a robot can follow complex instructions safely [7]. Nwankwo and Rueckert demonstrated that conversational dialogue with an LLM and vision model can command an autonomous robot in natural language, translating human requests into precise robot actions [8]. These developments suggest that LLMs can serve as a natural language interface layer for control systems.

Despite their potential, LLM-driven control faces critical concerns in regulated and safety-critical environments like pharma manufacturing. LLMs are stochastic and not guar-

anteed to respect domain constraints, so fully autonomous LLM control could produce unsafe or invalid actions [3]. Researchers emphasize incorporating moderation layers or human oversight to ensure reliability [9]. Indeed, a common pattern is to keep a human "on-the-loop" or "in-the-loop" [10], [11]. For example, Luo et al. present a code-generation approach for robotic assembly where an LLM produces control programs and a closed user-in-the-loop validation stage (including simulation and human feedback) verifies and refines the programs [6]. Similarly, trustworthiness is improved by adding explainability and constraint-checking to LLM controllers [3]. In pharmaceutical production, regulatory bodies (e.g., FDA) require assurance of product quality and process understanding, making a human-in-the-loop indispensable as a safety gate. The human operator provides final judgment on any LLM-suggested action, mitigating risks of blindly following AI recommendations. This aligns with the broader paradigm of human-in-the-loop AI, which leverages human oversight to maintain safety and ethical standards [9], [10].

Another emerging role of LLMs relevant to this work is automated insight generation from data. LLMs have been applied to analyze datasets and articulate insights in natural language for end-users. For instance, InsightPilot autonomously generates summaries and explanations of data patterns, allowing users to query their data in plain language [12]. Sánchez Pérez et al. (2025) similarly use LLMs to produce concise textual insights from database queries, by formulating hypotheses and summarizing the findings [13]. Systems such as JarviX and TableGPT further fine-tune LLMs to work with tabular data, yielding user-friendly analyses and visualizations of results [14], [15]. These advances show that LLMs can serve not only as control policy generators but also as analysis assistants that explain system behavior. In a process control context, this means an LLM could ingest the controller's performance metrics or time-series data and provide a human-readable evaluation ("the temperature overshot by 5°C and took 3 minutes to settle, which is within acceptable limits").

Contributions. This work proposes an LLM-enhanced MPC framework tailored for Pharma 4.0 that addresses both usability and interpretability challenges. The key contributions are: (i) enabling intuitive, high-level control adjustments via natural language commands, (ii) introducing an automated insight-generation feature that produces human-readable analysis of controller performance, and (iii) demonstrating the approach on a TCLab hardware case study with GEKKO-based MPC, where operators can dynamically tune objectives and constraints while retaining human-in-the-loop oversight. Together, these contributions illustrate a pathway to deploy advanced control in pharmaceutical manufacturing while preserving trust and regulatory compliance.

II. Methods

A. System Architecture

The proposed architecture consists of an operator station, an LLM engine, and an MPC controller running on a process (here, the TCLab device). The interaction flow is shown in Figure 1. The human operator issues a request in natural language (e.g., "reduce the overshoot on temperature setpoint changes"). This request is sent to the LLM engine via a client application using WebSocket communication. The LLM, which has been provided with contextual prompts about the control system, translates the request into specific MPC parameter adjustments or constraints that best reflect operator intent. The parameter adjustments are then applied to the MPC running on the server, which controls the TCLab hardware in real time. Throughout this loop, the human remains the final authority: the operator can choose whether to apply the LLM's suggestion and can monitor the system's response via plots, dashboards, and live data analysis (when requested).

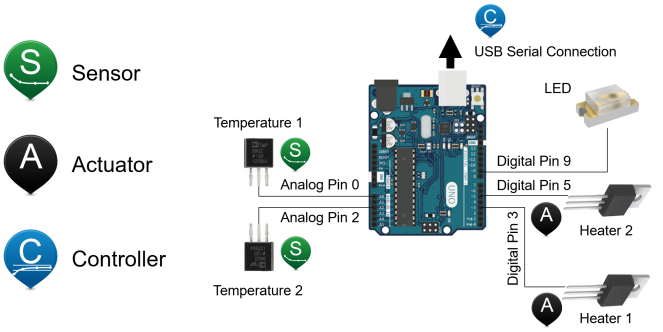

Fig. 2. Schematic of the Temperature Control Lab (TCLab) hardware illustrating the control loop architecture. Two temperature sensors (S) provide measurements to the Arduino controller (C) through analog inputs, while two heaters (A) receive control signals through digital outputs. The USB serial connection links the Arduino to the host computer, where the MPC and LLM interface run.

B. Temperature Control Lab Setup

The Temperature Control Lab (TCLab) was used as a hardware testbed to demonstrate the LLM-assisted MPC framework. The device consists of two heaters and two temperature sensors with nonlinear thermal dynamics, channel coupling, and operating constraints [16]. These characteristics make it a compact but representative proxy for pharmaceutical unit operations where overshoot, long time constants, and actuator limits must be managed. For our experiments, both heaters and sensors were used in a two-input, two-output configuration.

C. Model Predictive Control Formulation

We implemented a model predictive controller using the GEKKO optimization suite (APMonitor) in Python [17], [18]. GEKKO was chosen for its ability to handle dynamic models, constraints, and its Python API which allows real-time parameter changes. A first-principles model of the convection, radiation, and heat exchange between heaters 1 and 2 of the TCLab was identified. The discrete MPC formulation used a prediction horizon of H_p time steps (with $\Delta t = 5$ s) and a control horizon of H_c.

Within this formulation, the MPC incorporates an L1-norm objective function. The controller penalizes the absolute deviation between the predicted temperature trajectory and

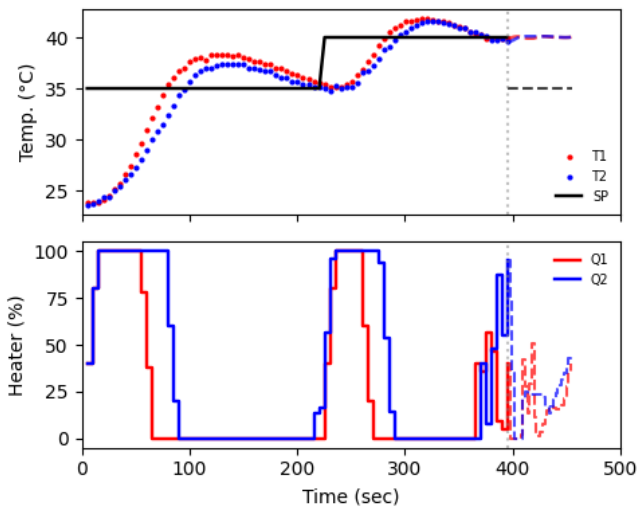

Fig. 3. Baseline MPC performance on the TCLab without tuning modifications. The default configuration produces 2–3°C overshoot, illustrating the need for operator-guided tuning.

the desired setpoint band across the entire horizon. This is implemented in GEKKO by introducing upper and lower setpoint bounds (SP_{lo}, SP_{hi}), with penalties applied linearly to deviations outside this band. At each step in the horizon, the optimization problem evaluates these deviations and aggregates their weighted contributions to the objective function:

$$\Phi = \sum_{k=1}^{H_p} \left(w_{hi} e_{hi}(k) + w_{lo} e_{lo}(k) \right) + \sum_{k=1}^{H_c} \left(u(k)^T w_u + \Delta u(k)^T w_{\Delta u} \right) \quad (1)$$

Here, e_{hi} and e_{lo} represent the nonnegative deviations above or below the allowable range, while u and Δu capture actuator effort and movement penalties, respectively. This structure ensures that the MPC evaluates both tracking error and actuator usage consistently over the prediction and control horizons.

GEKKO provides a wide range of tunable parameters that directly shape the MPC objective and constraint structure. Table I summarizes the key attributes exposed to the LLM, highlighting their mathematical role in the optimization problem and their qualitative impact on controller performance. These parameters can be adjusted in real time by issuing JSON commands to the server.

D. LLM Selection and Prompting Strategy

For the LLM component, we tested multiple models through API and local deployment, including `gpt-oss:20:9b`, `deepseek-ri:8:2b`, `llama3:2:3b`, and `gemma3:12b`. After evaluation, `gemma3:12b` was selected for its practical capabilities. It was effective enough to analyze system behavior and fast enough to run on our local GPU (32 GB VRAM, NVIDIA GeForce RTX), allowing the system to respond with commands and analysis in real time.

A structured system prompt was crafted to give the LLM context about the control system. The prompt describes the role of the LLM ("You are an expert control engineer assistant for a heating process with MPC..."), provides guidelines for formatting output responses, and it enumerates and describes the adjustable MPC parameters, their effect on the system, and what JSON commands to return to change them from default values.

When the LLM generates a response to an operator query, the response is scanned for JSON commands. A python function strips the JSON commands from the response, requests operator verification ("y"/"n"), and sends the approved JSON command via WebSocket communication to the server to implement the requested change.

The interaction history between operator, LLM, and controller is logged automatically by saving chat history. Recent process data, trends, and conditions (e.g. temperatures, setpoints, heater values, overshoot, etc.), while being displayed to the operator through plotting features, are assembled into a natural-language narrative using custom python functions. The log is updated at each control step and appended to the operator query, providing contextual grounding for LLM recommendations and analysis. A typical log entry looks like:

> *"Start of step 0. The current simulation has been running for 0.0 seconds. Temperature 1 sensor is moderate, measuring 20.9 °C. Heater 1 is currently off. Initial reading of setpoint 1 is 40.0 °C. Temperature 1 is 19.1 °C below its setpoint. End of step 0."*

E. MPC Objective Tuning via Natural Language

A key contribution of this work is enabling real-time tuning of MPC objectives through natural language commands. In this framework, the LLM serves as a translator from operator intent to JSON-formatted updates of GEKKO tuning attributes (Table I). This mapping allows high-level objectives to be expressed conversationally and enacted immediately in the controller.

For example, a request such as "make it respond faster" translates to decreasing `CV.COST` or reducing `CV.TAU`, while "avoid overshoot" can be achieved by increasing `CV.TAU`, tightening the setpoint band (`CV.SPLO`/`CV.SPHI`) or applying larger penalties on deviations above setpoint. Similarly, "smooth heater response" maps to increased penalties on actuator usage (`MV.COST`) or movement (`MV.DCOST`, `MV.DMAX`). Even hard constraints like safety boundaries can easily be implemented. For instance, "never exceed 70% heater" is enforced by setting `MV.UPPER`.

These mappings illustrate the novelty of the approach: operators can specify intuitive objectives in natural language, and the LLM automatically translates them into precise MPC adjustments.

F. Automated Analysis Feedback

The final component is the LLM-based analysis of controller performance. After or during an experiment, the operator can

TABLE I
GEKKO MPC TUNING ATTRIBUTES WITH EMPHASIS ON L1-NORM OBJECTIVE (CV.TYPE=1), THEIR EFFECT ON THE OBJECTIVE FUNCTION, ASSOCIATED MATHEMATICAL CONSTRAINTS, AND PERFORMANCE IMPLICATIONS

Attribute	Effect on Objective Function	Constraint Formulation	Impact on Performance
MV.UPPER /MV.LOWER	Adds bounds $u_{\min} \le u(t) \le u_{\max}$	$u_{\min} \le u_i \le u_{\max}$	Enforces actuator limits (e.g., heater capped at 50%), ensures safety and feasibility.
MV.DCOST	Weight on Δu term, $w_{\Delta u}$, penalizing control moves	$\Delta u_U \ge u_i - u_{i-1},\ \Delta u_L \ge u_{i-1} - u_i,\ \Delta u_U, \Delta u_L \ge 0$	Higher values reduce actuator activity and chatter; too high may slow responsiveness.
MV.DMAX	Maximum allowable step $\Delta u_{\max}$ per move	$\lvert u_i - u_{i-1} \rvert \le \Delta u_{\max}$	Prevents abrupt actuator changes, yielding smoother control but potentially limiting agility.
MV.COST	Weight on input magnitude u	Cost term $u^T w_u$	Discourages excessive heater usage, improving efficiency but risking slower tracking.
CV.COST	Weight on slack variables $e_{hi}, e_{lo} \Rightarrow$ L1 penalty outside $[SP_{lo}, SP_{hi}]$	$e_{hi} \ge y - y_{t,hi},\ e_{lo} \ge y_{t,lo} - y,\ e_{hi}, e_{lo} \ge 0$	Encourages staying within dead-band. Provides robustness and energy savings; may allow steady-state offset inside tolerance.
CV.SPLO /CV.SPHI	Define soft bounds around setpoint (dead-band)	$\tau_c \frac{dy_{t,hi}}{dt} + y_{t,hi} = sp_{hi}$ $\tau_c \frac{dy_{t,lo}}{dt} + y_{t,lo} = sp_{lo}$	Inside band: no penalty. Outside band: deviation penalized linearly. Reduces unnecessary actuation, prevents oscillations.
CV.TAU	Time constant for CV reference trajectory	Shapes reference dynamics τ_c	Small τ: aggressive, faster convergence but risk of overshoot. Large τ: gradual approach, safer dynamics.
CV.TR_INIT /CV.TR_OPEN	Shape initial or open-loop reference trajectory	Defines startup / open-loop trajectory behavior	Helps smooth startup behavior or allow freer CV evolution when relaxed.

request an analysis in natural language, such as "How did the controller perform?" or "Explain the temperature response."

The LLM is informed by a short sequence of log entries that provide a natural language narrative of the process input and output responses to the control strategy. Leveraging its trained knowledge and the context of recent actions, the LLM generates an explanation or insight.

This kind of analysis is similar to what InsightPilot and JarviX aim to provide in data analytics contexts [12], [14], here applied to control performance data. We found `gemma3:12b` to be capable of generating accurate and useful explanations, especially when guided by thorough log entries and an effective context prompt. The analysis output is not only useful to the human operator for process understanding, but also serves as a trace for validation: In pharmaceutical settings, having a natural language summary of each batch's control behavior could aid in batch release decisions and regulatory audits.

III. RESULTS

We conducted a series of experiments on the TCLab to evaluate how well the LLM could translate natural language commands into effective MPC modifications and to observe the resulting control performance. The baseline scenario was an MPC tuned for moderate performance (no special optimization for speed or smoothness and allowing slight overshoot). We then tested four representative operator commands: (1) "Increase/decrease controller aggressiveness," (2) "No overshoot," (3) "Smooth heater response," and (4) "Apply temperature and heater safety constraints." Each command was applied at the start of a new run, initialized by the same first setpoint change from ambient to 35°C to isolate their effects. Figures 4–7 show the temperature response of the MPC under each scenario, compared to the baseline.

A. Changing Controller Aggressiveness

In this experiment, the operator directed the LLM to adjust controller aggressiveness with the natural language request: `"make temperature 1 reach the setpoint faster and setpoint 2 reach the setpoint slower."` The LLM translated this into JSON commands that modified the reference trajectory constants (`CV.TAU`). Specifically, the time constant for temperature 1 was reduced from the default of 30 s to 5 s, making it more aggressive, while the time constant for temperature 2 was increased to 60 s, slowing its response.

Figure 4 shows the resulting closed-loop behavior. Temperature 1 (red) rose quickly to its new setpoint, but at the cost of overshooting by about 3°C. In contrast, temperature 2 (blue) followed a slower, smoother trajectory with less overshoot. The actuator responses reflected this trade-off: Heater 1 output (Q1) was highly active, while Heater 2 (Q2) operated more conservatively.

B. Overshoot Avoidance

In this experiment, the operator requested: `"Can you prevent both temperatures from overshooting?"` The LLM responded with several tuning commands, including unnecessary adjustments that kept the soft bounds (`SPHI`, `SPLO`) at their default values. However, the key effective change was raising the reference

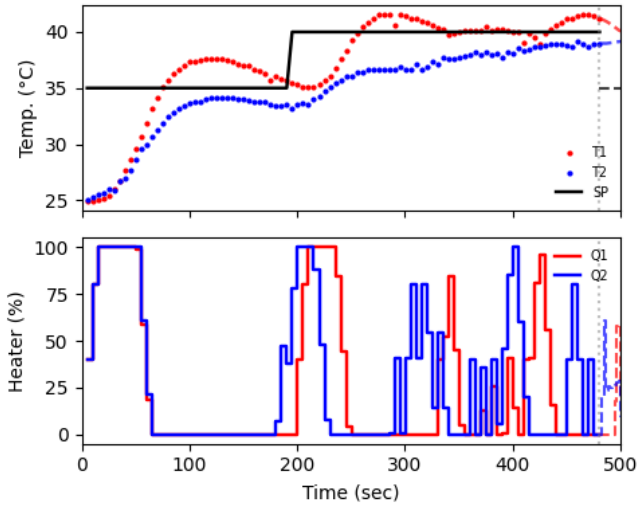

Fig. 4. Closed-loop response after natural language tuning of controller aggressiveness. Temperature 1 was made faster (lower τ_c) and overshot the setpoint, while Temperature 2 was slowed (higher τ_c) and tracked more smoothly.

trajectory constants (`CV.TAU`) for both temperatures from 30 s to 60 s, which slowed the closed-loop response and reduced the tendency to overshoot.

Figure 5 shows the resulting behavior. Both temperatures approached the setpoint smoothly from below, with no overshoot, at the cost of a slower rise time. The heater outputs (Q1, Q2) were correspondingly less aggressive, cycling less frequently than in the baseline case. This highlights both the promise and the challenge of natural language control: the LLM successfully captured the operator's intent by modifying trajectory aggressiveness, but also issued redundant parameter updates that did not contribute to overshoot prevention.

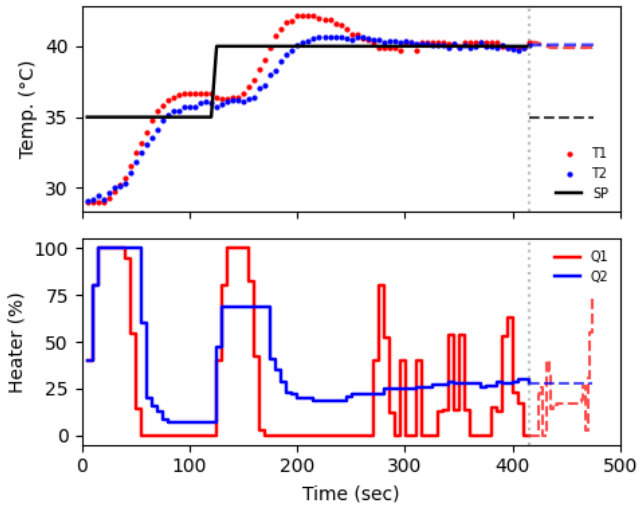

Fig. 5. Closed-loop response with overshoot avoidance tuning. Increasing the trajectory constants (τ_c) for both temperatures from 30 s to 60 s eliminated overshoot, though at the cost of slower convergence.

C. Smoothing Actuator Response

When the operator requested: "`heater 2 is jumping around too much, can you smooth it out?`", the LLM responded by increasing the movement penalty (`MV.DCOST`) on heater 2 from its default value, 0, to 10, and later to 20 after a follow-up request for even smoother behavior. These changes were intended to reduce oscillatory actuator behavior by penalizing rapid control moves.

Figure 6 shows the resulting closed-loop behavior. Heater 2 (Q2, blue) exhibited noticeably smoother actuation, with fewer abrupt changes and more gradual adjustments compared to the baseline. Heater 1 (Q1, red) was intentionally left at its default tuning to provide a comparison, and it continued to show chattering behavior. The smoothed heater response came at the cost of slower convergence, with both temperatures requiring nearly 400 s to approach the 40°C setpoint.

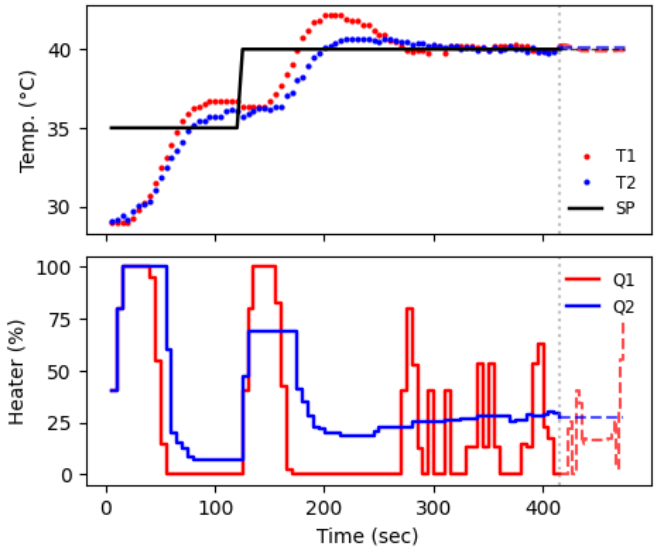

Fig. 6. Closed-loop response after natural language tuning to smooth actuator behavior. Increasing the movement penalty (`MV.DCOST`) for Heater 2 (Q2) reduced oscillations and produced smoother actuation, while Heater 1 (Q1) was left at default settings for comparison. The smoother actuation came at the cost of slower convergence to the setpoint.

D. Input and Output Constraints (Safety Limits)

The operator requested: "`keep temperature1 between 38 and 42 degrees, keep heater1 value between 20 and 70%`". The LLM translated this into four commands: upper and lower soft bounds on the controlled variable (`CV.SPHI`/`CV.SPLO`) and upper and lower limits on the manipulated variable (`MV.UPPER`/`MV.LOWER`). All four commands were successfully transmitted and applied.

Figure 7 shows the resulting closed-loop response. Heater 1 (bottom) was maintained strictly within the requested 20–70% range, confirming that input constraints were enforced as hard bounds. Temperature 1 (top) generally tracked within the desired band of 38–42°C but briefly exceeded the upper limit,

illustrating that setpoint bounds defined through SPHI/SPLO are treated as soft constraints. This behavior highlights a limitation: while input limits are enforced rigidly, output bounds require additional strategies if strict "hard" safety constraints on process variables are desired.

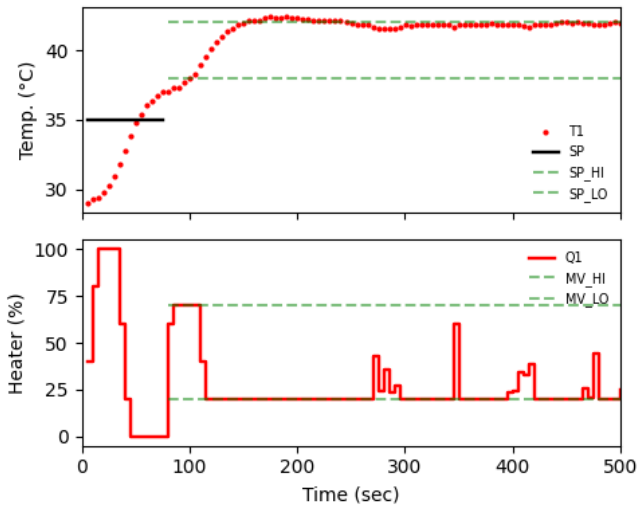

Fig. 7. Closed-loop response with input and output constraints applied through natural language commands. Heater 1 (bottom) stayed within 20–70%, but Temperature 1 (top) exceeded the 42°C soft upper bound. This demonstrates that input constraints act as hard limits, while output constraints require further methods to enforce strict safety guarantees.

IV. Conclusion

This work demonstrates a human-in-the-loop control framework that integrates large language models with model predictive control in the context of Industry 4.0 and Pharma 4.0. The LLM successfully translated high-level natural language commands into meaningful MPC adjustments, enabling operators to influence objectives such as response speed, overshoot minimization, energy efficiency, and actuator limits. Results on the TCLab confirmed that these objectives were achieved in line with classical control trade-offs: faster responses led to overshoot, constraints eliminated overshoot at the cost of slower convergence, energy prioritization reduced consumption with longer settling, and input limits ensured safety but introduced steady-state errors.

The human-in-the-loop paradigm ensured that all changes were validated by the operator, addressing safety and regulatory requirements central to pharmaceutical manufacturing. By combining intuitive operator interaction with the rigor of MPC, this approach lowers barriers to deploying advanced process control in regulated environments.

In the broader Pharma 4.0 vision, such systems can extend beyond heating experiments to batch crystallization, bioreactors, and chemical synthesis, where natural language guidance could simplify the tuning of complex multivariable systems. This human-centered integration of LLMs and MPC highlights a practical pathway toward smarter, more adaptive, and operator-friendly pharmaceutical manufacturing systems.

Acknowledgment

The authors thank management at Takeda Pharmaceuticals for financial and technical support.

References

[1] A. Mesbah, J. A. Paulson, R. Lakerveld, and R. D. Braatz, "Model predictive control of an integrated continuous pharmaceutical manufacturing pilot plant," *Organic Process Research & Development*, vol. 21, no. 6, pp. 844–854, 2017.

[2] M. Jelsch, Y. Roggo, P. Kleinebudde, and M. Krumme, "Model predictive control in pharmaceutical continuous manufacturing: A review from a user's perspective," *European Journal of Pharmaceutics and Biopharmaceutics*, vol. 159, pp. 137–142, 2021.

[3] S. Sarkar, S. Ghorbanpour, R. L. Gutierrez, A. Guillen-Perez, V. Gundecha, A. R. Babu, and A. Naug, "Review of llm based control."

[4] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," *arXiv preprint arXiv:2209.07753*, 2022.

[5] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, "Voyager: An open-ended embodied agent with large language models," *arXiv preprint arXiv:2305.16291*, 2023.

[6] H. Luo, J. Wu, J. Liu, and M. F. Antwi-Afari, "Large language model-based code generation for the control of construction assembly robots: A hierarchical generation approach," *Developments in the Built Environment*, vol. 19, p. 100488, 2024.

[7] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[8] L. Nwankwo and E. Rueckert, "The conversation is the command: Interacting with real-world autonomous robots through natural language," in *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, 2024, pp. 808–812.

[9] E. Mosqueira-Rey, E. Hernández-Pereira, D. Alonso-Ríos, J. Bobes-Bascarán, and Á. Fernández-Leal, "Human-in-the-loop machine learning: a state of the art," *Artificial Intelligence Review*, vol. 56, no. 4, pp. 3005–3054, 2023.

[10] S. Kumar, S. Datta, V. Singh, D. Datta, S. K. Singh, and R. Sharma, "Applications, challenges, and future directions of human-in-the-loop learning," *IEEE Access*, vol. 12, pp. 75 735–75 760, 2024.

[11] W. Huang, H. Liu, Z. Huang, and C. Lv, "Safety-aware human-in-the-loop reinforcement learning with shared control for autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[12] P. Ma, R. Ding, S. Wang, S. Han, and D. Zhang, "Demonstration of insightpilot: An llm-empowered automated data exploration system," *arXiv preprint arXiv:2304.00477*, 2023.

[13] A. S. Pérez, A. Boukhary, P. Papotti, L. C. Lozano, and A. Elwood, "An llm-based approach for insight generation in data analysis," *arXiv preprint arXiv:2503.11664*, 2025.

[14] S.-C. Liu, S. Wang, W. Lin, C.-W. Hsiung, Y.-C. Hsieh, Y.-P. Cheng, S.-H. Luo, T. Chang, and J. Zhang, "Jarvix: A llm no code platform for tabular data analysis and optimization," *arXiv preprint arXiv:2312.02213*, 2023.

[15] P. Li, Y. He, D. Yashar, W. Cui, S. Ge, H. Zhang, D. Rifinski Fainman, D. Zhang, and S. Chaudhuri, "Table-gpt: Table fine-tuned gpt for diverse table tasks," *Proceedings of the ACM on Management of Data*, vol. 2, no. 3, pp. 1–28, 2024.

[16] J. Park, R. A. Martin, J. D. Kelly, and J. D. Hedengren, "Benchmark temperature microcontroller for process dynamics and control," *Computers & Chemical Engineering*, vol. 135, p. 106736, 2020.

[17] J. D. Hedengren, R. A. Shishavan, K. M. Powell, and T. F. Edgar, "Nonlinear modeling, estimation and predictive control in APMonitor," *Computers & Chemical Engineering*, vol. 70, pp. 133 – 148, 2014.

[18] L. D. R. Beal, D. C. Hill, R. A. Martin, and J. D. Hedengren, "Gekko optimization suite," *Processes*, vol. 6, no. 8, 2018. [Online]. Available: http://www.mdpi.com/2227-9717/6/8/106