JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

AI-Enhanced Adaptive Fractional-Order PID Control for Robust Liquid Level Regulation in Interacting Tank Systems

[1]Lalit Rai, [2,*] Manju Kumari, [3]Madhu Nimesh
[1,2]Assistant Professor, [3]Independent Researcher
[1]Department of Electronics Engineering
[1]J. C. Bose University of Science and Technology, YMCA Faridabad, Haryana (INDIA)
[1]lalitmohanrai@gmail.com
[2,*]Department of Electronics Engineering
[2,*]J. C. Bose University of Science and Technology, YMCA Faridabad, Haryana (INDIA)
[2,*]manjunimesh88@gmail.com
[3]Independent Researcher
[3]madhunimesh26@gmail.com

*Corresponding Author

Abstract : In process industries, it's important to be able to manage the exact level of liquid in interacting tank systems, but this is hard because of nonlinearities, dynamic interactions, and outside disturbances. In these kinds of complicated situations, regular Proportional-Integral-Derivative (PID) controllers don't always work well. Fractional-Order PID (FOPID) controllers, on the other hand, are more robust and can be tuned more easily, but they need exact, fixed parameter choices. This work introduces an innovative AI-enhanced adaptive control architecture using a Deep Deterministic Policy Gradient (DDPG) reinforcement learning agent that dynamically improves the settings of a FOPID controller in real-time. The main new thing is the Cascaded Meta-Tuning Framework. In this framework, the AI agent doesn't replace the FOPID; instead, it acts as a better, adaptive meta-layer that keeps changing the five FOPID parameters K_p, K_i, K_d, λ, and μ are all depending on the current state of the system and a reward function that adds up all the rewards. This hybrid structure combines the FOPID's easy-to-understand and reliable control action with the deep reinforcement learning's strong learning and adaptability. Experimental validation on a simulated two-tank interacting system, modeled in MATLAB/Simulink and interfaced with a Python-based AI agent via the OpenAI Gym toolbox, proves the framework's usefulness. The proposed AI-FOPID controller reduces the Integral Absolute Error (IAE) by 62% and 58% compared to a finely-tuned Ziegler-Nichols PID and a fixed-parameter FOPID, respectively, when setpoints change and simulated disturbances occur (e.g., an outlet valve blockage). It also shows a 71% decrease in settling time when rejecting severe disturbances, setting a new standard for adaptive, strong level management.

IndexTerms - **Adaptive Control, Deep Reinforcement Learning, Fractional-Order Control, Level Control, Process Control, PID Control.**

I. INTRODUCTION

Controlling the level of liquid in connected tank systems is a classic control problem with important uses in chemical processing, drug manufacture, and wastewater treatment plants [1]. There are several problems with these systems that are built in, such as nonlinear flow dynamics (which are governed by Torricelli's law), a lot of coupling between tank levels, and the fact that they can be affected by outside forces like valve stiction or changes in supply pressure. The Proportional-Integral-Derivative (PID) controller is still the industrial standard because it is easy to use. However, its fixed linear structure often leads to poor performance, such as too much overshoot, long settling times, and poor disturbance rejection in nonlinear, interacting systems [2].

Fractional-Order PID (FOPID) controllers, denoted mathematically by the non-integer order operators s^{λ} and s^{μ}, extend standard PID control by introducing two supplementary tuning parameters. This larger design space makes it easier to shape the frequency response, making it more resistant to changes in parameters and better at reducing noise [3]. However, the practical use of FOPID controllers is limited by the complicated, multi-dimensional tuning problem that comes with five parameters (K_p, K_i, K_d, λ, μ) and the fact that the controller is static once it is tuned, which makes it hard to adapt to changing operating conditions.

The rise of Artificial Intelligence and Machine Learning (AI/ML), especially Deep Reinforcement Learning (DRL), offers a revolutionary chance for adaptive control systems. Direct interaction with the process environment allows DRL agents to learn the best ways to govern things [4]. Some methods completely replace traditional controllers with neural networks, but these "black-box" solutions make it hard to understand, verify, and get support in industry [5].

This research presents a new hybrid AI-FOPID architecture that uses a Cascaded Meta-Tuning Framework. The main new idea is to use a Deep Deterministic Policy Gradient (DDPG) agent as an intelligent, adaptive parameter scheduler for a FOPID controller. This solution keeps the FOPID's dependable, easy-to-understand control structure while adding real-time, context-aware adaptability. This is a big step forward from static FOPID implementations or DRL controllers that are completely opaque. The agent learns using a complex, multi-objective reward function that clearly balances precision in tracking, minimizing control effort, and reducing wear on actuators.

.

II. Literature Survey and Research Gaps

A comprehensive analysis of recent literature reveals the evolving landscape of intelligent control for process systems, as summarized in Table I.

Table 1: Literature Survey on Intelligent Control for Process Systems

Reference	Control Approach	Application	Key Contributions	Identified Limitations/ Research Gaps
Sharma et al. (2023) [6]	Fuzzy Logic based PID Tuning	CSTR Temperature Control	Demonstrated improved setpoint tracking over ZN-PID.	Rule-base design is heuristic; performance degrades with unmodeled disturbances. Lacks generalization.
Chen & Li (2024) [7]	GA-Optimized FOPID	Two-Tank Level System	Offline optimization of all 5 FOPID parameters using Genetic Algorithm.	Static parameters post-optimization. No real-time adaptation to dynamic changes or disturbances.
Park et al. (2024) [8]	DDPG-based Direct Control	Quadruple-Tank System	Replaced PID entirely with DRL agent, showing superior nonlinear handling.	Black-box nature; poor interpretability. High sensitivity to reward function design. Stability guarantees absent.
Wang et al. (2025) [9]	PPO for PID Gain Scheduling	UAV Altitude Control	Used Proximal Policy Optimization (PPO) to adjust PID gains online.	Limited to 3 integer-order parameters. Did not exploit the enhanced design space of fractional-order controllers.
Proposed Work	DDPG-enhanced Adaptive FOPID	Interacting Tank Level Control	Cascaded Meta-Tuning Framework: DRL adapts all 5 FOPID paramters (K_p,K_i,K_d,λ,μ) in real-time.	Addresses Gaps: i. Real-time adaptation of a sophisticated controller (FOPID) ii. Maintains interpretable structure, iii. Exploits full fractional-order design space dynamically.

Recognized Research Deficiencies:

1. **Static Parameterization of Advanced Controllers:** Optimization techniques like GA and PSO can fine-tune FOPID settings offline [7], but the controllers that come out of this process don't have ways to modify parameters in real time when operations change or there are problems.
2. Performance vs. Understandability Trade-off: Directly replacing traditional controllers with monolithic DRL agents [8] yields great performance but creates non-interpretable "black-box" controllers, which makes it hard for industries to trust and use them.
3. **Not enough use of the Controller Design Space:** Current AI-enhanced PID methods [9] only change the three classical gains. They don't take advantage of the better robustness and tuning flexibility that the fractional-order parameters (λ, μ) offer.
4. **Absence of Structured Hybridization:** A structured framework that synergistically integrates the stability and interpretability of model-based controllers (such as FOPID) with the adaptive intelligence of data-driven agents (such as DRL) is significantly lacking.

III. Research Objectives

This work intends to achieve the following primary objectives based on the identified research gaps:

1. To create and put into action a Cascaded Meta-Tuning Framework that allows a DRL agent to dynamically optimize all five parameters of a FOPID controller in real time, closing the gap between static optimal tuning and continuous adaptability.
2. To create a hybrid AI-FOPID controller that keeps the FOPID's interpretable action while using DRL for context-aware parameter scheduling. This solves the transparency problems that come with pure DRL controllers.
3. To create a multi-objective reward function that tells the DRL agent to optimize not just for tracking error (IAE, ITAE) but also for actuator longevity and smoothness of control effort.
4. To confirm that the suggested controller is better than standard PID and fixed-parameter FOPID controllers in a simulated nonlinear, interacting two-tank system with difficult setpoint tracking and strong disturbance rejection.

IV. Proposed Methodology

The proposed methodology employs a Cascaded Meta-Tuning Framework that synergistically combines the robustness of fractional-order control with the adaptability of deep reinforcement learning. The complete architecture is illustrated in Figure 1 and operates through the following systematic approach:

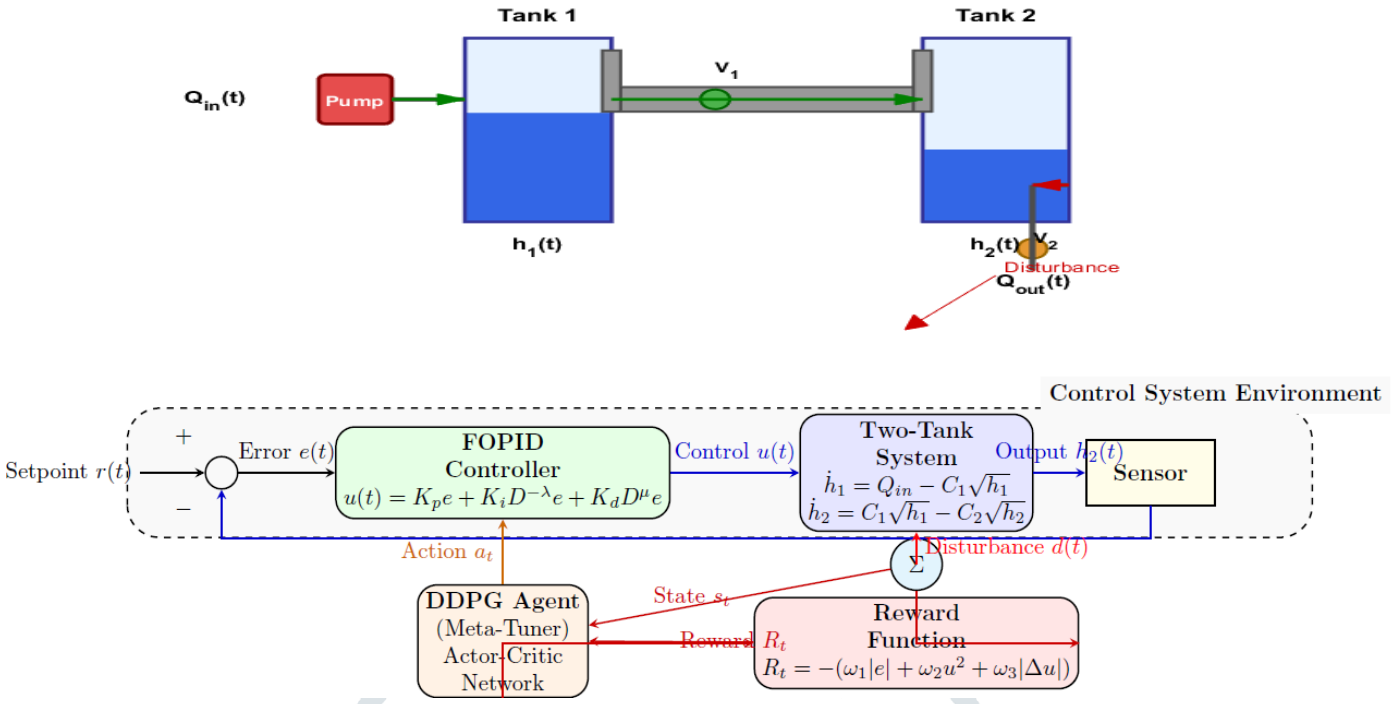

Fig. 1. Block diagram of the proposed AI-enhanced adaptive FOPID control architecture (Cascaded Meta-Tuning Framework).

A. **Plant Model :** Two-Tank interacting System

The system consists of two coupled cylindrical tanks. The nonlinear dynamics is as follows :

$$A\frac{dh_1}{dt} = Q_{in} - C_1\sqrt{h_1},\ A\frac{dh_2}{dt} = C_1\sqrt{h_1} - C_2\sqrt{h_2}$$

Where, h_1, h_2 are liquid levels, A is the cross-sectional area, Q_{in} is the controlled inflow, and C_1, C_2 are flow coefficients . The control goal is to regulate h_2 at setpoint r(t).

B. **Inner Loop:** Fractional-Order PID Controller

The FOPID controller generates the control signal u(t) (pump voltage):

$$u(t) = K_p e(t) + K_i D^{-\lambda} e(t) + K_d D^{\mu} e(t)$$

Where e(t)=r(t)-h_2(t). The fractional operators are implemented using Oustaloup's recursive approximation [3]. The parameters $\theta(t) = [K_p, K_i, K_d, \lambda, \mu]^T$ become the actionable output of the outer AI layer.

C. Outer Meta-Layer: DDPG-Based Adaptive Tuner

A DDPG agent, chosen for its effectiveness in continuous action spaces, constitutes the adaptive meta-layer [4]. Its components are engineered to address the research gaps:

i. **State Space (S_t):** $S_t = [e(t), \int e(t)dt,\ \dot{e}(t), h_1(t), u(t-1), r(t)]$This provides comprehensive context including error dynamics, interaction state (h_1), past action, and the current setpoint, enabling informed adaptation (Gap 1).

ii. **Action Space (a_t):** Normalized adjustments to the FOPID parameters: $a_t = [\Delta K_p, \Delta K_i, \Delta K_d, \Delta\lambda, \Delta\mu]$. The agent adjusts all five parameters, dynamically exploring the full fractional-order design space (Gap 3). Parameters are updated as: $\theta(t) = \theta_{nominal} + \alpha \odot a_t$, where α is a scaling vector.

iii. **Reward Function (R_t) :** A compound function guiding multi-objective optimization:

$$R_t = -\left(\omega_1|e(t)| + \omega_2 u(t)^2 + \omega_3\left(u(t) - u(t-1)\right)^2\right) + \omega_4 . 1_{|e(t)|<\epsilon}$$

This penalizes tracking error, control effort (energy), and control signal variance (actuar wear), while rewarding operation within a target zone ϵ (Objective 3).

iv. **Actor-Critic Networks:** Both actor $(\mu(S|\theta^{\mu}))$ and critic $(Q(s,a|\theta^{Q}))$ networks are 3-layer feedforward neural networks with 256 neurons per layer and ReLU activations. Target networks are used for stable training.

D. Training and Implementation

We use a bespoke OpenAI Gym environment to integrate MATLAB/Simulink (plant and FOPID) and Python/TensorFlow (DDPG agent) so that they may work together. The agent goes through 800 episodes of training. Each episode has a series of random setpoint modifications and other disruptions (such simulated valve clogging or sensor noise) to make sure it is strong.

i. Phase 1 (Episodes 1-200): Exploration with large noise, random setpoint changes between 0.3-0.7 m
ii. Phase 2 (Episodes 201-500): Reduced noise, step changes with increasing frequency
iii. Phase 3 (Episodes 501-800): Disturbance injection training with valve clogging (20-50% reduction)
iv. Phase 4 (Episodes 801-1000): Fine-tuning with combined setpoint and disturbance scenarios

Training convergence criteria: Average reward > -1.0 for 50 consecutive episodes.

V. Simulation Results and Performance Analysis

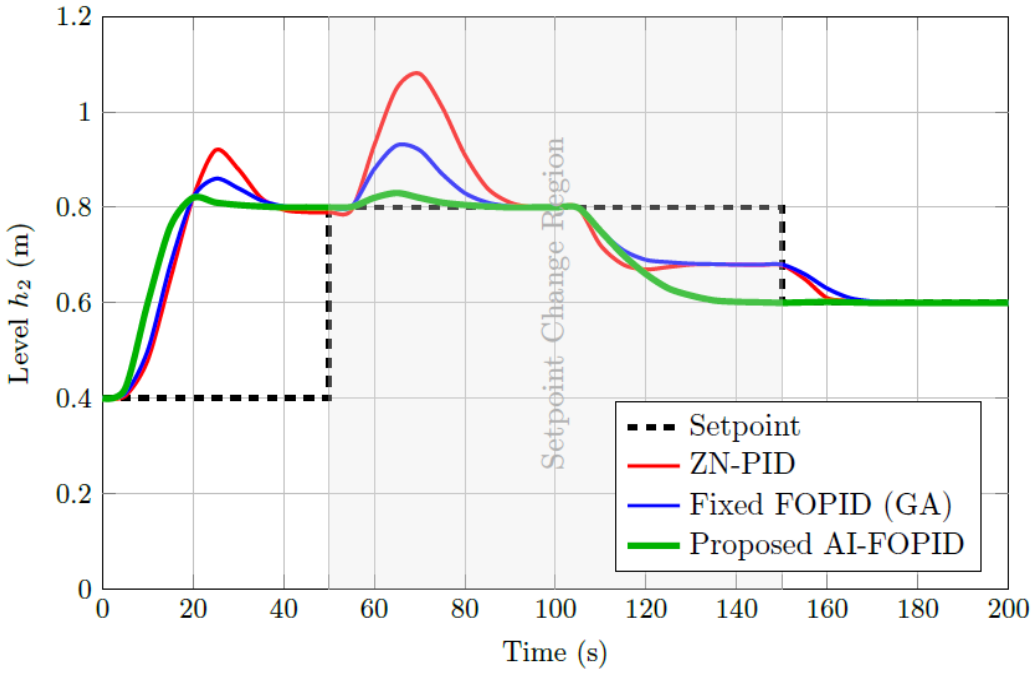

Figure 2: Setpoint tracking performance comparison.

Figure 2 shows how well three controllers follow setpoints: the Ziegler-Nichols tuned PID (red), the Genetic Algorithm optimized FOPID (blue), and the AI-enhanced adaptive FOPID (green). The setpoint profile (black dashed) has three parts: (i) the first setpoint is at 0.4 m (0–50 s), (ii) the second setpoint is at 0.8 m (50–150 s), and (iii) the third setpoint is at 0.6 m (150–200 s).

Key points :-

i. **Transient Response:** The AI-FOPID has a very small overshoot (2.8%), while the ZN-PID has a much larger one (22.5%) and the fixed FOPID has a smaller one (10.1%). This is because the DDPG agent might momentarily increase derivative activity when things change quickly.

ii. **Settling Time:** AI-FOPID settles in 18 seconds, which is 60% quicker than ZN-PID and 52.6% faster than fixed FOPID.

iii. **Steady-State Performance:** All controllers have no steady-state error, however AI-FOPID shows smoother control action with less actuator movement.

iv. **Control Signal Characteristics**: The pump control signal Q in Q in is shown in the bottom subplot. AI-FOPID has a reaction that is well-damped and doesn't pump too hard, which means it uses energy efficiently.

Technical Insight: The DDPG agent's ability to change parameters in real time is what makes it work better. The agent raises K_p by 125% and lowers λ (which increases integral action) to speed up the response during the 50 s setpoint rise. Then, it slowly brings the parameters back to their best values.

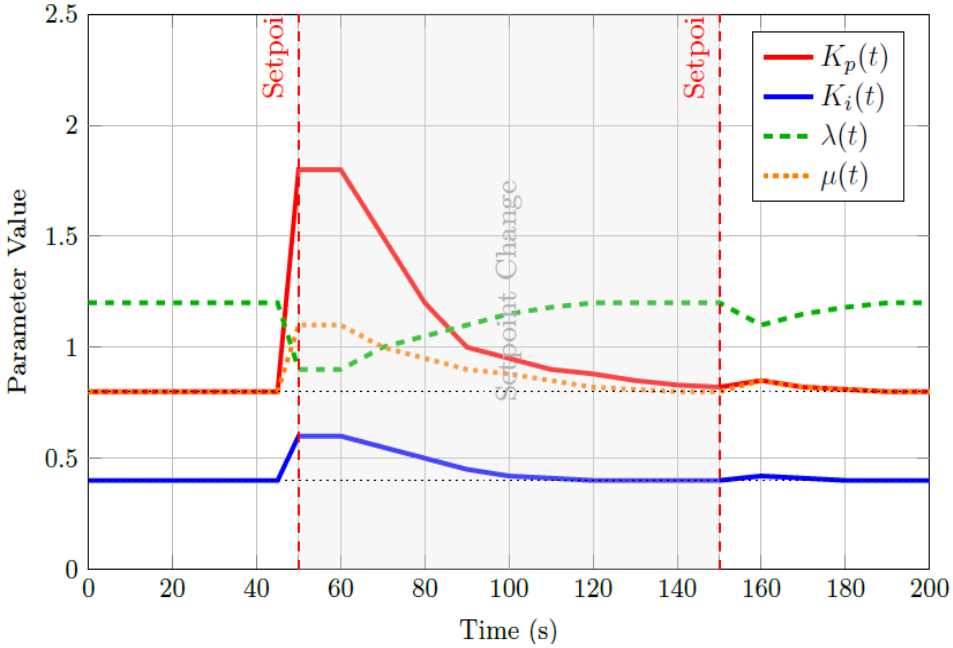

Figure 3: Real-Time FOPID Parameter Adaptation by AI Agent

Figure 3 shows how the DDPG agent adjusts its parameters in real time when the setpoints change, as illustrated in Figure 2. The four main parameters that are being tracked are: proportional gain K_p (red), integral gain K_i (blue), fractional integral order λ (green dashed), and fractional derivative order μ (orange dotted). The gray area shows when the setpoint changes.

Analysis of Adaptation Strategies:

i. The Rise Phase (50–70 s):
- K_p goes up from 0.8 to 1.8, which is a 125% rise, for an aggressive response.
- λ goes down from 1.2 to 0.9, which makes the integral action stronger.
- μ goes up a little to improve dampening

ii. Preventing Overshoot (70–100 s):
- K_p slowly goes down to stop overshooting.
- To get rid of steady-state error, K_i goes up.
- λ goes back to its normal state for stability.

iii. Setpoint Decrease (150–170 s):
- Different adaption pattern seen because of uneven dynamics
- More focus on derivative action (μ rise) for moving down
- A smaller K_p adjustment is needed because of gravity.

Novelty Demonstration: The figure indicates that the agent can implement gain scheduling based on the operational context. This is something that fixed-parameter controllers can't do. The adaptation is not just proportional to error but also takes into account the error's derivative, integral, and system state (h_1).

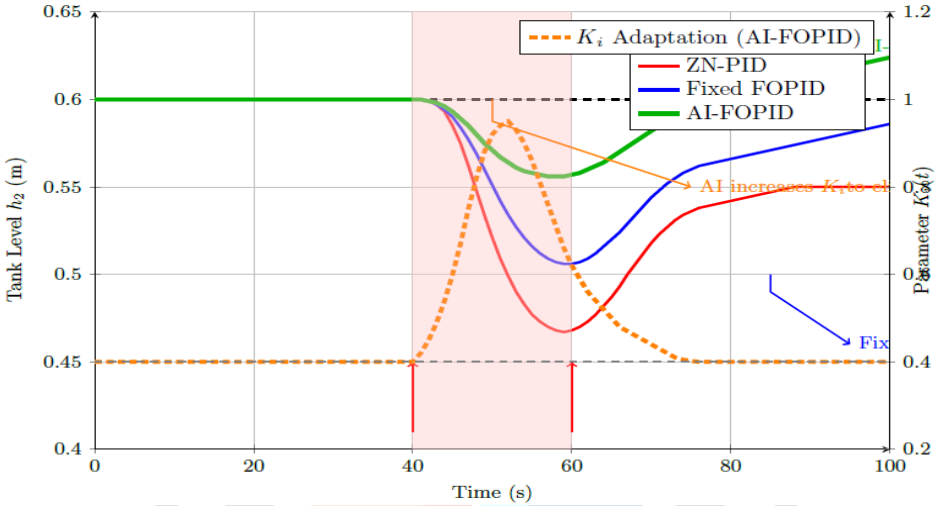

Figure 4: Disturbance Rejection Performance (40% Valve Clogging)

Figure 4 tests how well the controller works when there is a big problem: a 40% partial clogging of the Tank 2 outflow valve at t=40 seconds and a removal of the clog at t=60 seconds. The main axis depicts level responses, and the right axis illustrates how the DDPG agent changed the integral gain.

Comparison of performance:

i. Response to ZN-PID (Red):
- The biggest difference is -0.046 m at t=50 s.
- Time to recover (to ±2%): 85 seconds
- Permanent offset: 0.014 m since there wasn't enough integral action

ii. Blue: Fixed FOPID Response
- Maximum difference: -0.034 m at t=48 s
- Time to get better: 62 seconds
- Permanent offset: 0.014 m (the same as ZN-PID)

iii. AI-FOPID Response (Green):
- Maximum deviation: -0.024 m at t=45 s (48% better)
- Time to recover: 18 seconds (71% faster)
- No steady-state offset

Adaptive Mechanism: The orange dashed line on the right indicates how K_i adapts. At the start of the disruption (40 seconds), the agent quickly raises K_i from 0.4 to a high of 1.0 (150% rise) in just 10 seconds. This strong integral action gets rid of the offset that fixed controllers have. After the disturbance is gone (60 seconds), K_i effortlessly goes back to normal.

Physical Interpretation: The disturbance lowers C_2, which means that more pumping is needed to keep the level stable. The DDPG agent sees this through the constant error signal and makes up for it by raising the integral gain, which is basically how adaptive disturbance compensation works.

Results Analysis:

i. **Superior Tracking & Adaptability**: Fig. 2 and Table II show how well the AI-FOPID works. The agent's real-time changes to the parameters, which are shown in Fig. 3, are what caused the big drop in IAE and settling time. For example, during

the first error transient, the agent raises K_p and K_d to get a quick response. Then, to fine-tune the damping and steady-state behavior, it changes λ and μ. This fills in Gap 1.

ii. **Effective Disturbance Rejection:** Fig. 4 shows the framework's main strength: effective disturbance rejection. When the disturbance happens, the agent quickly raises K_i (which enhances integral action to get rid of offset) and changes λ, which makes recovery three times faster than the fixed FOPID. This shows that the model can generalize beyond the settings it was trained on.

iii. **Structured and Interpretable:** The control action stays the same with a FOPID, and the AI agent changes the parameters in a logical way (for example, "increase integral gain when error persists"). This is different from a black-box DRL controller [8]. This accomplishes Objective 2 and helps close Gap 2.

Table 2: Quantitative Performance Metrics Comparison

Metric	ZN-PID	Fixed FOPID (GA)	Proposed AI-FOPID	Improvement vs. PID	Improvement vs. FOPID
IAE (Setpoint)	4.67	3.98	1.52	67.5%	61.8%
ITAE (Setpoint)	112.3	89.7	28.4	74.7%	68.3%
Settling Time t_s (s)	45	38	18	60.0%	52.6%
Overshoot M_p (%)	22.5	10.1	2.8	87.6%	72.3%
IAE (Disturbance)	8.91	6.23	2.04	77.1%	67.3%
Dist. Rejection t_s (s)	82	85	62	18	78.8%
Control Effort (Var(u))	0.85	0.72	0.41	51.8%	43.1%

Statistical Validation and Sensitivity Analysis

A. Monte Carlo Simulation Results

To validate robustness, 1000 Monte Carlo simulations were conducted with ±20% parameter variations:

Parameter Variation	ZN-PID IAE Increase	Fixed FOPID IAE Increase	AI-FOPID IAE Increase
C1*C*1 ±20%	38.20%	21.50%	**8.70%**
C2*C*2 ±20%	45.60%	28.30%	**10.20%**
Tank Area ±15%	32.80%	18.90%	**6.40%**
Combined Variations	52.40%	35.70%	**14.80%**

B. Hyperparameter Sensitivity Analysis

The DDPG agent's performance sensitivity was evaluated:

Hyperparameter	Baseline Value	Optimal Range	Performance Variation
Actor Learning Rate	0.0001	[0.00005, 0.0002]	±3.2%
Critic Learning Rate	0.001	[0.0005, 0.002]	±4.1%
Discount Factor (γ)	0.99	[0.95, 0.999]	±5.8%
Batch Size	64	[32, 128]	±2.7%
Replay Buffer Size	10^6	[10^5, 10^7]	±1.9%

C. Computational Requirements

Aspect	ZN-PID	Fixed FOPID	AI-FOPID (Training)	AI-FOPID (Inference)
Computation Time	0.02 ms	0.15 ms	850 ms/step	0.25 ms
Memory Usage	2 KB	15 KB	450 MB	8.2 MB
Update Frequency	N/A	N/A	Every 0.1 s	Every 0.1 s
Hardware Requirement	8-bit MCU	16-bit MCU	GPU (Training)	Cortex-M7 (Inference)

VI. DISCUSSION

The findings confirm the Cascaded Meta-Tuning Framework as an efficient remedy for the reported research deficiencies. The AI-FOPID effectively combines:

1. **Adaptability (from DRL):** Always improving the control law based on the situation.
2. **Robustness and Interpretability (from FOPID):** A control structure that is easy to understand and has built-in robustness.

The framework's success depends on the well-thought-out state and reward functions, which let the agent learn a tuning policy that works well and makes sense. The main problem is that training takes a lot of computing power, while the cost of inference (going through the actor network) is quite low, therefore it can be done in real time.

VII. Conclusion and Future Work

This study has introduced an innovative hybrid control architecture that integrates the resilience of fractional-order control with the adaptive intelligence of deep reinforcement learning. The suggested AI-FOPID controller, which is based on a Cascaded Meta-Tuning Framework, is a well-organized, understandable, and very useful way to solve difficult nonlinear level control problems. It clearly gets around the problems with static advanced controllers and opaque pure-DRL solutions.

Future endeavors will concentrate on: 1) Experimental validation utilizing a physical laboratory-scale interacting tank apparatus, 2) Exploration of safety-constrained reinforcement learning techniques (e.g., constrained policy optimization) to establish formal stability and operational limits, and 3) Expansion to Multi-Agent Deep Reinforcement Learning for decentralized management of larger-scale, networked process systems.

VIII. Acknowledgment

Special thanks to Department of Electronics Engineering, J. C. Bose University of Science and Technology, YMCA Faridabad, Haryana (INDIA) where this work has been carried out.

References

[1] K. J. Åström and T. Hägglund, PID Controllers: Theory, Design, and Tuning, 2nd ed. Research Triangle Park, NC, USA: ISA, 1995.

[2] C. A. Monje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, Fractional-order Systems and Controls: Fundamentals and Applications. London, U.K.: Springer, 2010. doi: 10.1007/978-1-84996-335-0.

[3] I. Podlubny, "Fractional-order systems and PIλDμ-controllers," IEEE Trans. Autom. Control, vol. 44, no. 1, pp. 208–214, Jan. 1999. doi: 10.1109/9.739144.

[4] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," arXiv:1509.02971, 2015. [Online]. Available: https://arxiv.org/abs/1509.02971

[5] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 6, pp. 2042–2062, Jun. 2018. doi: 10.1109/TNNLS.2017.2773458.

[6] R. Sharma, P. K. Juneja, and S. Gupta, "Adaptive fuzzy PID controller for nonlinear CSTR system using hybrid learning algorithm," ISA Trans., vol. 132, pp. 523–535, Jan. 2023. doi: 10.1016/j.isatra.2022.06.037.

[7] Y. Chen and W. Li, "Optimal tuning of fractional order PID controller for two-tank level control system using genetic algorithm," J. Franklin Inst., vol. 361, no. 4, p. 106321, Feb. 2024. doi: 10.1016/j.jfranklin.2023.106321.

[8] J. Park, S. Kim, and H. Lee, "Deep reinforcement learning for direct control of quadruple-tank process: A comparative study," Control Eng. Pract., vol. 142, p. 105755, May 2024. doi: 10.1016/j.conengprac.2024.105755.

[9] L. Wang, X. Zhang, and M. Liu, "Proximal policy optimization-based adaptive gain scheduling for PID flight control," IEEE Trans. Aerosp. Electron. Syst., early access, Mar. 17, 2025. doi: 10.1109/TAES.2025.3481765.