Leveraging Deep Reinforcement Learning for Effective PI Controller Tuning in Industrial Water Tank Systems

Vijaya Lakshmi Korupu
School of Electrical Engineering, Vellore Institute of Technology (VIT), Vellore, Tamil Nadu, India
vijaya.korupu@gmail.com

Muthukumarasamy Manimozhi
School of Electrical Engineering, Vellore Institute of Technology (VIT), Vellore, Tamil Nadu, India
mmanimozhi@vit.ac.in (corresponding author)

Received: 13 November 2024 | Revised: 9 December 2024 and 27 December 2024 | Accepted: 29 December 2024

 DOI: https://doi.org/10.48084/etasr.9602

ABSTRACT

This paper addresses the level control problem in water tank systems by proposing a Deep Deterministic Policy Gradient (DDPG) algorithm to automatically tune the parameters of a Proportional-Integral (PI) controller. The integration of the PI controller with the DDPG algorithm leverages the strengths of both methods, enabling the algorithm to learn optimal controller gains through the exploration of the state-action space and reward feedback from the system. The proposed approach eliminates manual tuning, automates gain adaptation to varying system states, and ensures a robust performance under uncertainties and disturbances. The validation results demonstrate that the DDPG-tuned PI controller outperforms the manually tuned controller using the PID Tuner app in Simulink, achieving no overshoot, faster settling times, and enhanced robustness. These findings highlight the potential of Reinforcement Learning (RL) for adaptive control in industrial applications, particularly for systems with dynamic and uncertain environments.

Keywords-DDPG algorithm; level control; PI controller; tuning; reinforcement learning; water tank system

I. INTRODUCTION

The control of the water level in tanks is a crucial task in many industrial applications [1], ensuring process efficiency, safety, environmental compliance, equipment protection and product quality across industries such as manufacturing, power generation, wastewater treatment, oil and gas, HVAC, and agriculture. In these applications, a precise level control is required to ensure efficient operation and prevent the overflow or underflow. For the level control, the PI controller is sufficient as it is not a slow process like temperature, which requires the Proportional-Integral-Derivative (PID) for fast response. There are various methods for tuning the PI controller parameters, but the stability and system performance when the PI controller is tuned by these methods are reduced when there are time varying uncertain states and disturbances [2]. The gains should be re-tuned when there are variations in the environment, as they are optimal under only one set of operating conditions. This becomes a time-consuming process, and often requires more knowledge of the system dynamics [3].

To overcome the limitations of the manual tuning process, several researchers have proposed the use of intelligent control methods, such as the fuzzy logic control [4-6], Neural Network (NN) control [7-9], and RL-based control methods [10-12]. Among these techniques, many people today are interested in control approaches based on RL because they can learn and adapt to the changing conditions of the control problem. RL provides a framework for learning optimal control policies through repeated interactions with the environment [13-15]. One of the popular RL algorithms that uses Deep Neural Networks (DNN) is DDPG, which is a model-free, actor-critic RL algorithm that has been effectively utilized in various control problems [16].

From the existing knowledge about the environment model, the RL algorithms can be classified into two distinct categories: model-based and model-free [17, 18]. Authors in [19] present a brief review of the model-based and model-free RL and some of the recent developments in RL algorithms. Model-based RL algorithms learn a model of the system dynamics and then use this model to make decisions. The model accuracy is very important [17, 18]. Model-free RL algorithms do not require a model of the system dynamics and learn the optimal policy directly from the data. These approaches do not need an estimation of the Markov Decision Process (MDP) model, and the value or policy function can be directly evaluated [17, 18].

Q-learning is a popular RL algorithm in which an agent learns a Q-function that estimates the expected long-term

reward of taking an action in each state. Deep Q-Networks (DQNs) use DNNs to approximate the Q-function. DQNs have been successfully applied to control various processes. DDPG is a deep RL algorithm that directly optimizes the policy of the agent and has been applied to various systems. DDPG uses a deterministic policy instead of a stochastic policy and is more stable. Proximal Policy Optimization (PPO) is a policy optimization method that has been applied to many systems. PPO uses a clipped surrogate objective function to ensure stable policy updates. Trust Region Policy Optimization (TRPO) is another policy optimization method that has been applied to various systems. TRPO utilizes a trust region constraint to ensure that the policy updates are not too large.

The limitations of traditional PI controllers lie in the management of time-varying uncertain states and disturbances, emphasizing the need for more adaptive control methods. Key studies are reviewed that explore the use of intelligent control techniques, such as fuzzy logic, NNs, and RL, to address these challenges. However, there is still a need for real-time adaptation and robust control solutions that can maintain efficiency and safety under varying operating conditions. This research aims to fill these gaps by employing the DDPG algorithm in RL to develop controllers to extract control actions or control parameters from the agent. DDPG is well suited for continuous action spaces [20]. Various modifications of the DDPG algorithm have been proposed in the literature [21]. This algorithm was chosen in this work due to its higher efficiency in utilizing sample data compared to PPO [22].

This work proposes a method for tuning the PI controller parameters for water level control in a tank using the DDPG algorithm. The DDPG approach generates adaptive optimal gains in response to the changing system states. As a model-free algorithm, DDPG learns both a deterministic policy and a Q-function to estimate long-term rewards [23]. The aim is to control the water levels by minimizing the overshoot and settling time, with the agent tuning the PI controller parameters based on the error, its integral, and its derivative. The performance is evaluated by comparing the DDPG-tuned controllers with those manually tuned using the PID Tuner app [24], showing superior results across all metrics. This approach can be applied to other control problems, offering an efficient solution for the PI controller tuning.

II. WATER TANK SYSTEM MODEL

Figure 1 depicts the water tank system. Water enters the tank at a rate proportional to the pump supply voltage (V) from the top as in (1). The water drains out at a rate equal to the square root of the water level (H) through a hole in the bottom of the tank as in (2).

The water tank system is a nonlinear system because the outflow rate equation contains the square root function [25].

$$Q_{in} = b*V \quad (1)$$

$$Q_{out} = a*\text{sqrt}(H) \quad (2)$$

where a and b are proportional constants. The water tank system model is described by the differential in (3).

$$dH/dt = (Q_{in} - Q_{out})/A \quad (3)$$

where A is the tank's cross-sectional area. Substituting (1) and (2) into (3), we obtain:

$$dH/dt = (b*V - a*\text{sqrt}(H))/A \quad (4)$$

The water tank model implemented from (4) is shown in Figure 2. The tank level is controlled by a PI controller that adjusts the flow rate of the pump with the parameters, as presented in Table I. The goal of the PI controller is to maintain the water level at the desired value while minimizing the overshoot and settling time.

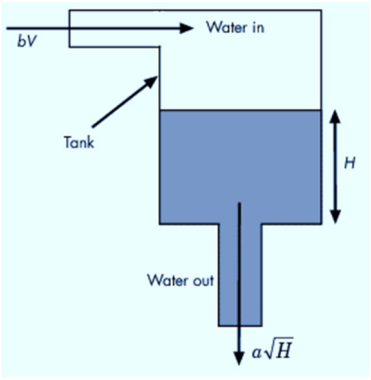


Fig. 1. Water tank system.

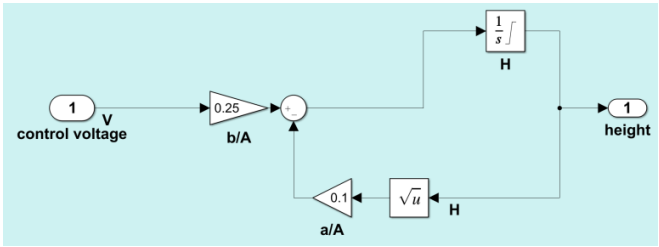


Fig. 2. Simulink model of the water tank system.

III. DEEP DETERMINISTIC POLICY GRADIENT ALGORITHM

The DDPG algorithm is widely used for continuous control tasks, utilizing DNNs to approximate optimal control policies. It has been applied to systems, such as robotics, quadcopters, and process control [26]. DDPG, a model-free, off-policy algorithm, can tune a PI controller for the level control of a water tank system, as displayed in Figure 3. It uses actor and critic networks, where the critic estimates the Q-value, representing the cumulative reward for following the current policy [27]. The critic is trained using the Bellman equation and Mean Squared Error (MSE) loss to refine the Q-value estimates [28]. The critic network in DDPG processes the state and action inputs and outputs a scalar Q-value that evaluates the current policy and guides the actor network toward improvement [29]. The critic estimates the Q-value to guide the actor network, which maps the states to actions by optimizing a policy for maximum long-term rewards using the Q-value gradient through a policy gradient loss function [30]. Both the

actor and critic networks use DNNs with Rectified Linear Unit (ReLU) or tanh activation functions, optimized via Adam. The exact architecture and hyperparameters of the networks can be tuned to achieve optimal performance on a specific task [31]. The DDPG algorithm also employs a replay buffer to store experience tuples that are randomly sampled, to update the networks, as shown in Figure 4, preventing overfitting by breaking the correlations between experiences [32].

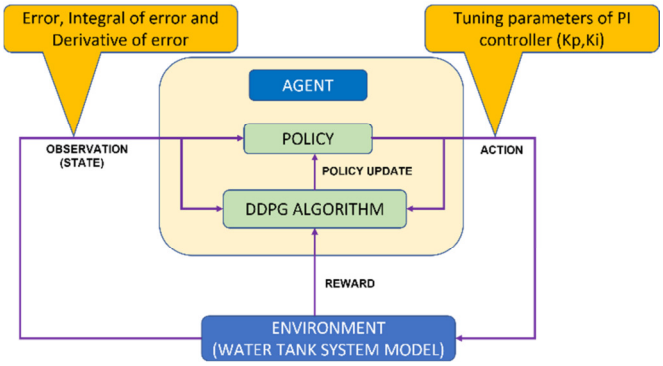


Fig. 3. Representation of the interaction between an agent and an environment.

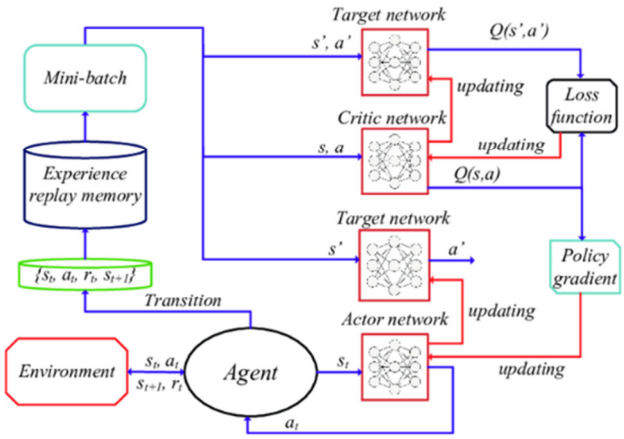


Fig. 4. Structure of the DDPG algorithm.

The actor and critic networks work together in DDPG to learn a policy that maximizes the long-term rewards in a continuous action space. The actor selects actions, while the critic estimates their Q-value. Through iterative updates, the DDPG algorithm converges to an optimal policy.

IV. IMPLEMENTATION

In this work, the DDPG algorithm in RL has been used to tune the PI controller gains for the level control of a water tank system. Here, the goal is to maintain the water level at the required value by manipulating the pump speed. To implement the DDPG algorithm for tuning the PI controller for the water tank system, the steps are as follows:

- Define the state space: The state space for the water tank system includes error, integral of error, and derivative of error, where error is the difference between the current water level and the desired water level.
- Define the action space: The action space includes the PI controller parameters such as the proportional gain (K_p) and integral gain (K_i).
- Define the reward function: The reward function should incentivize the agent to keep the water level close to the desired level. A possible reward function could be based on the previous control effort and the error between the desired level and the actual level.
- Specify the NN architecture: The actor NN should take the current state as input and provide the PI controller parameters. The critic NN should take the current state and the actor network output as inputs and output the corresponding state-action value.
- Initialize the networks and the replay buffer: The replay buffer is a data structure used to store experience tuples (state, action, reward, next state) from which the agent can sample during training.
- Train the networks using the DDPG algorithm: During each episode, the agent selects a small batch of experience tuples from the replay buffer and updates the DNN based on the gradients computed from the loss function.
- Evaluate the trained policy: After training, the efficacy of the learned policy can be evaluated by running simulations of the water tank system and measuring the error between the desired and actual levels.
- Fine-tune the policy if necessary: Depending on the results of the evaluation, the policy can be fine-tuned by adjusting the hyperparameters or the NN architecture.

Once the DDPG algorithm has converged, the learned policy can be used to control the level of the water tank system in real time. The PI controller can be implemented using the learned policy by utilizing the current water level and setpoint as inputs to the policy and applying the resulting pump speed. The DDPG agent is trained using the specified training options and the training progress is plotted. The trained agent can be then utilized to maintain the tank level at the desired value. In this environment, the RL agent is defined using the 'rlDDPGAgent' function, which creates a DDPG agent with specified options [33-35]. The model for the training of the RL agent is portrayed in Figure 5. During training, the agent interacts repeatedly with the environment by taking actions based on its current policy, receiving feedback in the form of the reward signal, and updating its policy and value functions based on the received feedback.

MATLAB's RL toolbox was deployed to create the RL agent. It requires an observation vector, a reward function, and the Boolean 'isdone'. With each new training phase, the target water level is chosen randomly. The quadratic reward function is designed with penalties based on the squared error, squared control effort, and exceeding limits. If the water level exceeds the limits, the 'isdone' block terminates the simulation. The agent receives nothing at the end of the episode if this occurs. The RL agent's output contains the optimal gains of the PI controller. These gains are then added together to input the net control effort into the water tank system model.

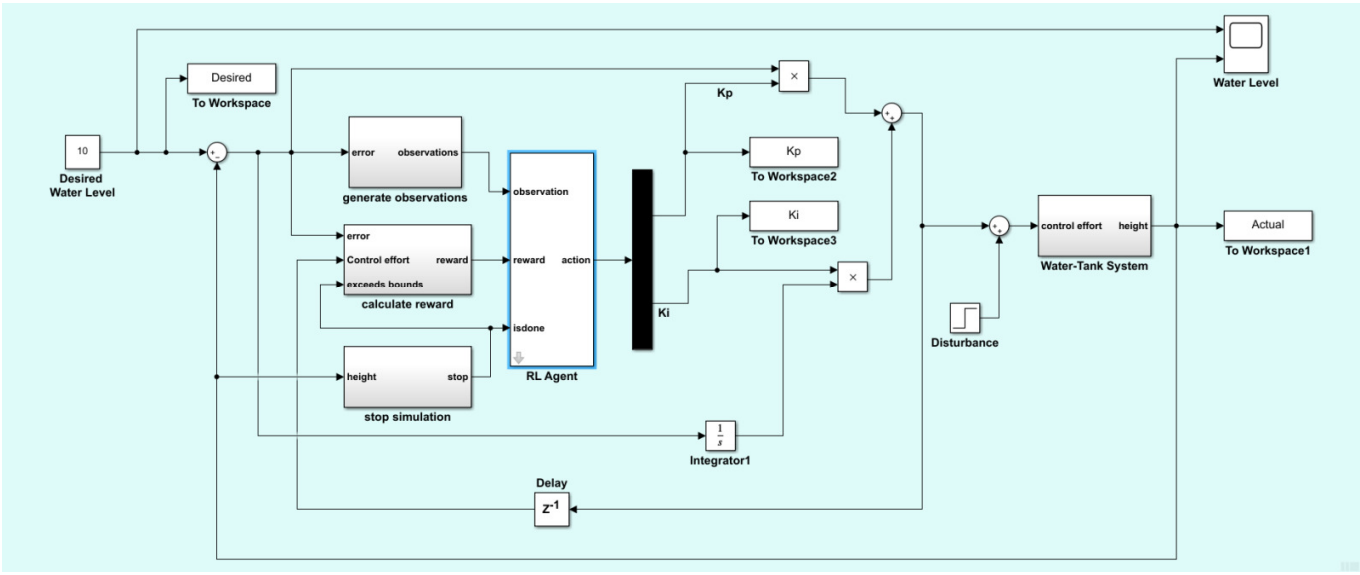


Fig. 5. Simulink model of the water tank system with RL agent.

The agent consists of two fully connected layers: an actor and a critic, each with 500 hidden units, and a ReLU activation layer. ReLU is preferred in DDPG for its constant gradient for positive inputs, aiding the gradient propagation during backpropagation. The sampling time T_s is set to 1, with a final time T_f of 200. The training progress over the episodes can be seen from the RL episode manager displayed in Figure 6.

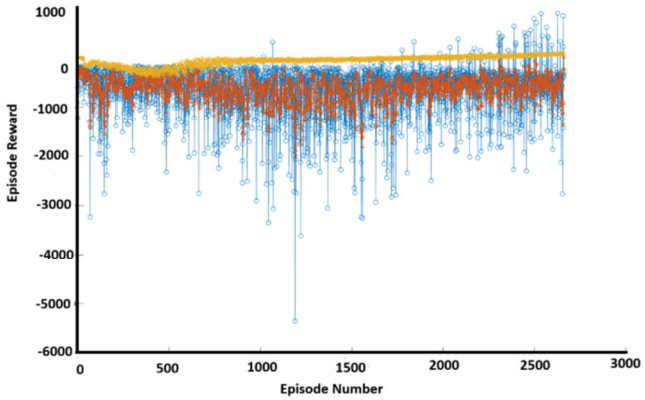


Fig. 6. Training progress of RL DDPG agent for water tank system.

The training is completed when the output water level matches the desired water level repeatedly over a predetermined number of training episodes, indicating that the agent has well-tuned the PID controller to provide the desired setpoint tracking response / disturbance rejection response. Then, validation is carried out on the agent.

V. RESULTS AND DISCUSSION

The desired and actual water levels in the case of setpoint tracking are presented in Figure 7. A water level of 10 cm is considered as the reference signal. Figure 8 shows a comparison of the disturbance rejection responses of both controllers. The dynamic response is measured in terms of rise time, overshoot, settling time, and steady state error.

The PI controller tuning parameters obtained with both methods are tabulated in Table I. The proportional constant (K_p) determines the controller's responsiveness, with the PID Tuner app selecting a higher K_p=1.555 for both scenarios, while the RL DDPG uses a lower K_p=1 for improved stability and robustness. The integral constant (K_i) in the RL DDPG approach is adaptive, with K_i=0.029 for stable setpoint tracking and K_i=0.4015 for faster disturbance rejection, demonstrating its flexibility compared to the fixed K_i=0.077 of the PID Tuner app. Typical ranges for PI controller constants are K_p between 0.1 and 10 and K_i between 0.01 and 1.

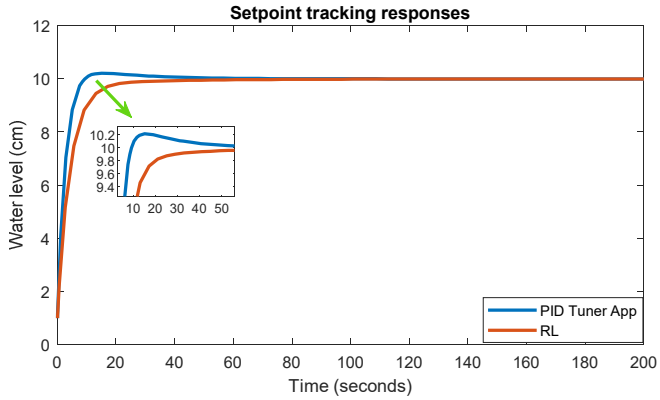
TABLE I. COMPARISON OF TUNING PARAMETERS

Tuning method	Parameter	Setpoint tracking	Disturbance rejection
PID Tuner app	K_p	1.555	1.555
	K_i	0.077	0.077
RL DDPG	K_p	1	1
	K_i	0.029	0.4015

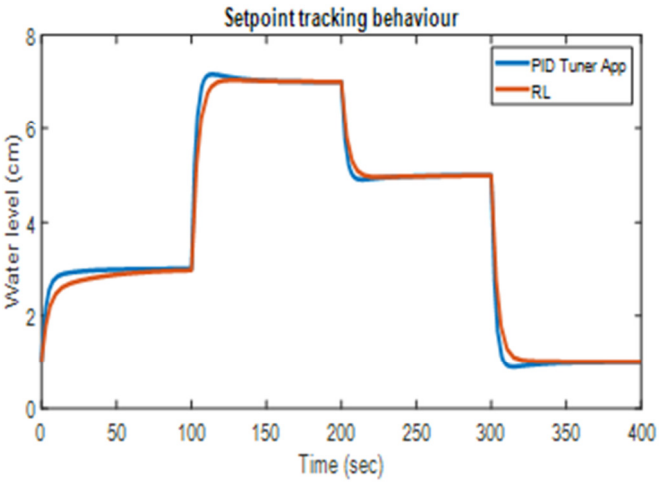
In case of 10% additive model uncertainty, where a/A is 0.11 and b/A is 0.275, the response with RL is better with no overshoot and less settling time, as evidenced in Figure 9(a). Also, in the case of 10% subtractive model uncertainty, where a/A is 0.09 and b/A is 0.225, the RL tuned controller shows better performance than the controller tuned with the PID Tuner app, as demonstrated in Figure 9(b).

Both the DDPG and PID Tuner app controllers demonstrate stable responses, but the RL tuning method achieves a higher phase margin and faster settling time without overshoot. DDPG achieves adaptive gains for time-varying states, with optimal performance at noise variance 0.6, while the excessive network complexity (>500 layers) causes reward oscillations. The

optimal hyperparameters are tabulated in Table II. Table II also shows the range of each DDPG parameter, guiding the setup of the algorithm for stable and efficient training.



(a)



(b)

Fig. 7. Comparison of setpoint tracking responses (a) single setpoint (b) setpoint variations.

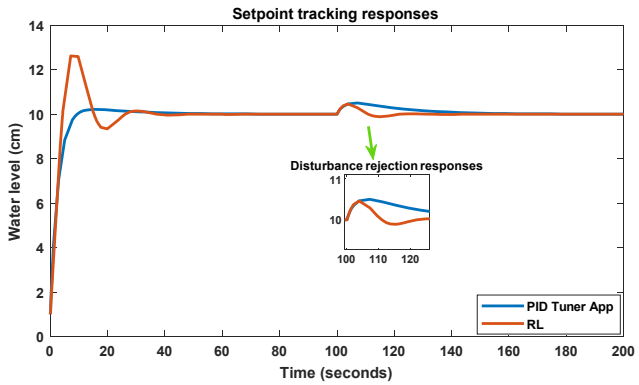
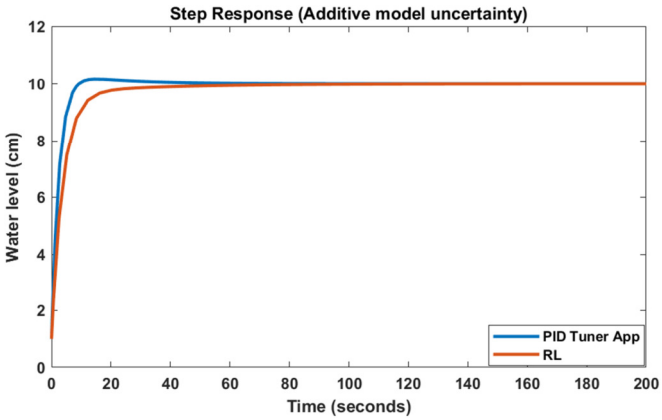
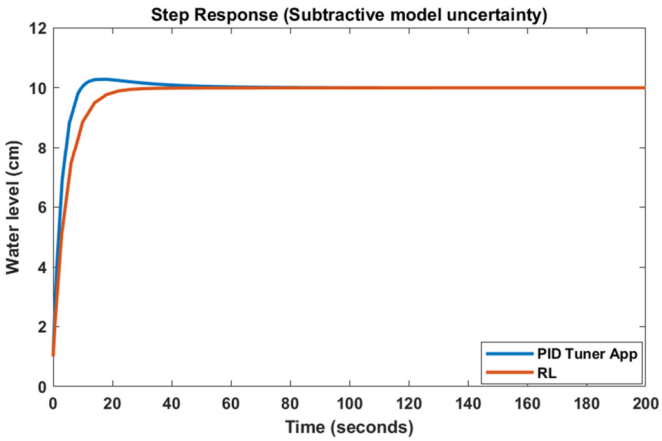


Fig. 8. Comparison of disturbance rejection responses.



(a)



(b)

Fig. 9. Comparison of setpoint tracking responses in case of (a) additive model uncertainty (b) subtractive model uncertainty.

TABLE II. PARAMETERS USED FOR TUNING DDPG AGENT

Model training parameters		
Hyper parameter	**Value**	**Typical range**
Critic learn rate	1e-03	1e-05 to 1e-03
Actor learn rate	1e-04	1e-06 to 1e-04
Critic gradient threshold	1	0.5 to 5
Actor gradient threshold	1	0.5 to 5
Variance	0.6	0.1 to 1
Variance decay rate	1e-5	1e-5 to 1e-3
Experience buffer	1e6	1e4 to 1e6
Mini-batch size	64	32 to 128
Target smooth factor	1e-3	1e-4 to 1e-2
Discount factor	0.99	0.9 to 0.99
Training options		
Parameter	**Value**	**Typical range**
Max. episodes	5000	1000 to 100,000
Max. steps	T_f/T_s	Depends
Score averaging window length	5	5 to 50
Stopping criteria	Avg. reward	Depends

Adjusting the sampling time, final time, reward weights, learning rates, and variance, with a variance of 0.6, yielded stable performance with reduced oscillations, improved settling time, and minimized overshoot during agent training, as

detailed in Table III for setpoint tracking. The DDPG-tuned PI controller demonstrates a better overall performance with no overshoot, slightly faster settling time, and lower Integral Square Error (ISE) and Integral Absolute Error (IAE) values compared to the PI controller tuned with the PID Tuner app. Although the DDPG controller has a slightly slower rise time, it offers improved stability and robustness, with a higher phase margin and better disturbance rejection. The PID Tuner app prioritizes the faster response by selecting higher gains, potentially resulting in faster rise times but more tradeoffs, whereas the DDPG algorithm focuses on optimizing the long-term performance, improving settling time, overshoot, and robustness, potentially at the cost of a slower rise time. The DDPG-tuned controller dynamically adapts to varying conditions, balancing the system behavior more effectively than the fixed gains of manual tuning. The RL DDPG controller effectively tracks a desired water level of 10 cm with no overshoot and fast settling time, demonstrating closed-loop stability and robustness. In contrast, the PID Tuner app exhibits overshoot and requires user intervention for tuning, whereas the RL approach dynamically adapts gains based on random water levels without prior knowledge of desired characteristics.

TABLE III. COMPARISON OF PERFORMANCE INDICES

Parameter	PID Tuner app	DDPG
Rise Time (sec)	5.4842	9.9864
Settling Time (sec)	21.598	20.835
Overshoot (%)	2.1752	0
Steady state error	0	0
Gain Margin	Inf.	Inf.
Phase Margin	90.005	94.753
Stable	Yes	Yes
ISE	515.07	491.92
IAE	65.53	63.03

VI. CONCLUSION

This study presents a novel approach to tuning Proportional-Integral (PI) controller parameters for water level control in industrial tank systems using the Deep Deterministic Policy Gradient (DDPG) algorithm. The integration of DDPG with a PI controller demonstrates significant advantages, including adaptive gain tuning, no overshoot, faster settling time, and robustness against the model uncertainties and external disturbances. Compared to the manually tuned PI controller using the (PID) Tuner app, the proposed Reinforcement Learning (RL)-based method exhibits superior performance, validating its potential for real-time applications.

The novelty of this work lies in leveraging DDPG's model-free RL capabilities to address the limitations of conventional tuning methods, such as their dependence on accurate system dynamics and their inability to adapt to time-varying conditions. The proposed methodology automates the tuning process, reduces the need for manual intervention, and provides a scalable solution for industrial control systems. DDPG enables real-time, adaptive, and precise tuning of the PI controllers in dynamic systems, in contrast to the rule-based bounds of the Fuzzy Logic Controllers (FLCs) or the offline, time-consuming optimization of the Genetic Algorithms (GAs). Future research can extend this approach to nonlinear control systems, use alternative reward functions or algorithms, such as Proximal Policy Optimization (PPO), to minimize the settling time, and validate the controller performance in real-world scenarios.

REFERENCES

[1] C. Urrea and F. Páez, "Design and Comparison of Strategies for Level Control in a Nonlinear Tank," *Processes*, vol. 9, no. 5, May 2021, Art. no. 735, https://doi.org/10.3390/pr9050735.

[2] M. J. Blondin, J. Sanchis Sáez, and P. M. Pardalos, "Control Engineering from Classical to Intelligent Control Theory—An Overview," in *Computational Intelligence and Optimization Methods for Control Engineering*, M. J. Blondin, P. M. Pardalos, and J. Sanchis Sáez, Eds. Cham, Switzerland: Springer International Publishing, 2019, pp. 1–30.

[3] M. Abdelkader, M. Mabrok, and A. Koubaa, "OCTUNE: Optimal Control Tuning Using Real-Time Data with Algorithm and Experimental Results," *Sensors*, vol. 22, no. 23, Jan. 2022, Art. no. 9240, https://doi.org/10.3390/s22239240.

[4] "Water Level Control in a Tank," Mathworks. [Online]. Available: https://www.mathworks.com/help/fuzzy/water-level-control-in-a-tank.html.

[5] M. Khairudin, A. Hastutiningsih, T. Maryadi, and H. Pramono, "Water level control based fuzzy logic controller: simulation and experimental works," *IOP Conference Series: Materials Science and Engineering*, vol. 535, no. 1, May 2019, Art. no. 012021, https://doi.org/10.1088/1757-899X/535/1/012021.

[6] K. T. Sundari, R. Giri, M. G. Umamaheswari, S. Durgadevi, and C. Komathi, "Fuzzy Logic Control of Liquid Level in a Single Tank with IoT-Based Monitoring System," in *ICDSMLA 2021*, A. Kumar, S. Senatore, and V. K. Gunjan, Eds. Singapore: Springer Nature, 2023, pp. 401–414.

[7] Muhlasin, Budiman, M. Ali, A. Parwanti, A. A. Firdaus, and Iswinarti, "Optimization of Water Level Control Systems Using ANFIS and Fuzzy-PID Model," in *2020 Third International Conference on Vocational Education and Electrical Engineering*, Surabaya, Indonesia, 2020, pp. 1–5, https://doi.org/10.1109/ICVEE50212.2020.9243229.

[8] S. Bhadra *et al.*, "Implementation of Neural Network Based Control Scheme on the Benchmark Conical Tank Level System," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference*, Las Vegas, NV, USA, 2019, pp. 0556–0560, https://doi.org/10.1109/CCWC.2019.8666578.

[9] B. S. Sousa, F. V. Silva, and A. M. F. Fileti, "Level Control of Coupled Tank System Based on Neural Network Techniques," *Chemical Product and Process Modeling*, vol. 15, no. 3, Sep. 2020, https://doi.org/10.1515/cppm-2019-0086.

[10] M. M. Noel and B. J. Pandian, "Control of a nonlinear liquid level system using a new artificial neural network based reinforcement learning approach," *Applied Soft Computing*, vol. 23, pp. 444–451, Oct. 2014, https://doi.org/10.1016/j.asoc.2014.06.037.

[11] D. Dutta and S. R. Upreti, "A multiple neural network and reinforcement learning-based strategy for process control," *Journal of Process Control*, vol. 121, pp. 103–118, Jan. 2023, https://doi.org/10.1016/j.jprocont.2022.12.004.

[12] O. Dogru *et al.*, "Reinforcement learning approach to autonomous PID tuning," *Computers & Chemical Engineering*, vol. 161, May 2022, Art. no. 107760, https://doi.org/10.1016/j.compchemeng.2022.107760.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[14] P. Ramanathan, K. K. Mangla, and S. Satpathy, "Smart controller for conical tank system using reinforcement learning algorithm," *Measurement*, vol. 116, pp. 422–428, Feb. 2018, https://doi.org/10.1016/j.measurement.2017.11.007.

[15] J. Shin, T. A. Badgwell, K.-H. Liu, and J. H. Lee, "Reinforcement Learning – Overview of recent progress and implications for process control," *Computers & Chemical Engineering*, vol. 127, pp. 282–294, Aug. 2019, https://doi.org/10.1016/j.compchemeng.2019.05.029.

[16] A. Alharin, T.-N. Doan, and M. Sartipi, "Reinforcement Learning Interpretation Methods: A Survey," *IEEE Access*, vol. 8, pp. 171058–171077, 2020, https://doi.org/10.1109/ACCESS.2020.3023394.

[17] Q. J. M. Huys and P. Seriès, "Reward-Based Learning, Model-Based and Model-Free," in *Encyclopedia of Computational Neuroscience*, D. Jaeger and R. Jung, Eds. New York, NY, USA: Springer, 2022, pp. 3042–3050.

[18] Q. Huang, "Model-Based or Model-Free, a Review of Approaches in Reinforcement Learning," in *2020 International Conference on Computing and Data Science*, Stanford, CA, USA, 2020, pp. 219–221, https://doi.org/10.1109/CDS49703.2020.00051.

[19] M. Lapan, *Deep Reinforcement Learning Hands-On: Apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO, AlphaGo Zero and more*. Birmingham, UK: Packt Publishing, 2018.

[20] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning." arXiv, Jul. 05, 2019, https://doi.org/10.48550/arXiv.1509.02971.

[21] M. Hossny, J. Iskander, M. Attia, K. Saleh, and A. Abobakr, "Refined Continuous Control of DDPG Actors via Parametrised Activation," *AI*, vol. 2, no. 4, pp. 464–476, Dec. 2021, https://doi.org/10.3390/ai2040029.

[22] D. Salwan and S. Kant, "DDPG vs PPO in Prosthetics," *Journal of Critical Reviews*, vol. 7, no. 9, Apr. 2020, https://doi.org/10.31838/jcr.07.09.482.

[23] "Deep Deterministic Policy Gradient (DDPG) Agent," Mathworks. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/ddpg-agents.html.

[24] "Tune PID Controller," Mathworks. [Online]. Available: https://www.mathworks.com/help/control/ref/tunepidcontroller.html.

[25] "Watertank Simulink Model," Mathworks. [Online]. Available: https://www.mathworks.com/help/slcontrol/ug/watertank-simulink-model.html.

[26] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement Learning for UAV Attitude Control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, Apr. 2019, Art. no. 22, https://doi.org/10.1145/3301273.

[27] T. M. Luu and C. D. Yoo, "Hindsight Goal Ranking on Replay Buffer for Sparse Reward Environment," *IEEE Access*, vol. 9, pp. 51996–52007, Mar. 2021, https://doi.org/10.1109/ACCESS.2021.3069975.

[28] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Systems with Applications*, vol. 141, Mar. 2020, Art. no. 112963, https://doi.org/10.1016/j.eswa.2019.112963.

[29] R. Wu, F. Gu, H. Liu, and H. Shi, "UAV Path Planning Based on Multicritic-Delayed Deep Deterministic Policy Gradient," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, Mar. 2022, Art. no. 9017079, https://doi.org/10.1155/2022/9017079.

[30] A. Ajagekar and F. You, "Deep Reinforcement Learning Based Automatic Control in Semi-Closed Greenhouse Systems," *IFAC-PapersOnLine*, vol. 55, no. 7, pp. 406–411, 2022, https://doi.org/10.1016/j.ifacol.2022.07.477.

[31] Ó. Pérez-Gil *et al.*, "Deep reinforcement learning based control for Autonomous Vehicles in CARLA," *Multimedia Tools and Applications*, vol. 81, no. 3, pp. 3553–3576, Jan. 2022, https://doi.org/10.1007/s11042-021-11437-3.

[32] L. Wang, K. Wang, C. Pan, and N. Aslam, "Joint Trajectory and Passive Beamforming Design for Intelligent Reflecting Surface-Aided UAV Communications: A Deep Reinforcement Learning Approach," *IEEE Transactions on Mobile Computing*, vol. 22, no. 11, pp. 6543–6553, Nov. 2023, https://doi.org/10.1109/TMC.2022.3200998.

[33] "Train Reinforcement Learning Agents," Mathworks. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/train-reinforcement-learning-agents.html.

[34] "Water Tank Reinforcement Learning Environment Model," Mathworks. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/water-tank-simulink-reinforcement-learning-environment.html.

[35] "Tune PI Controller Using Reinforcement Learning," Mathworks. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/tune-pi-controller-using-td3.html.

AUTHORS PROFILE

Vijaya Lakshmi Korupu was born in Visakhapatnam, Andhra Pradesh, India in 1982. She received a B.Tech degree in Instrumentation Engineering from VRSEC in 2003, and an M.Tech degree in Industrial Process Instrumentation from the Andhra University, in 2006. Currently, she is pursuing a Ph.D. degree in School of Electrical Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu from 2017. From 2006 to 2008, she was a Lecturer at VRSEC, Vijayawada. Since 2008, she has been an Assistant Professor with the Electronics and Instrumentation Engineering Department, VRSEC, Vijayawada, India. Her research interests include process control, control systems, process modeling and simulation, reinforcement learning, and optimization techniques. She has completed a research project from ANURAG, DRDO, Hyderabad worth 9.8 Lakhs in 2018. She has completed 5 NPTEL certifications on recent technologies and acted as a reviewer for a few articles. K. Vijaya Lakshmi was a life member of ISOI and BMESI.

Muthukumarasamy Manimozhi received the B.E. degree in Electronics and Instrumentation engineering and the M.E. degree in Process Control and Instrumentation from Annamalai University, and the Ph.D. degree from VIT University, Vellore. She is currently a Professor in the Department of Control and Automation, VIT University. Her research work focused on the areas of Control and Instrumentation, State Estimation, Sensors and Signal Conditioning, reinforcement learning.