WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Design of PID Controller using Reinforcement Learning

Ashis De [a], Barun Mazumdar [b], Aritra Dhabal [c], Saikat Bhattacharjee [d], Aridip Maity [e], Sourav Bandopadhyay [f]

[a-f]*Gargi Memorial Institute of Technolgy, Balarampur, Baruipur, Kolkata-700144, India*

DOI: https://doi.org/10.55248/gengpi.4.1123.113004

A B S T R A C T

This paper presents the design challenge of integrating a new adaptive updating rule based on Reinforcement Learning (RL) approach into a Proportional-Integral-Derivative (PID) controller for nonlinear systems. This study presents a novel design approach where Reinforcement Learning (RL) complements conventional PID control technology. The proposed scheme utilizes a single Radial Basis Function (RBF) network to concurrently calculate the control policy function of the Actor and the value function of the Critic. The inputs of the RBF network correspond to system error, output difference, and second-order output difference, defining them as system states within the PID controller structure. A newly defined Temporal Difference (TD) error, incorporating an error criterion based on the discrepancy between one-step ahead prediction and the reference value, is employed. By utilizing the gradient descent method with the TD error performance index, the updating rules are derived, enabling adaptive calculation of network weights and the kernel function. The efficiency and robustness of the proposed scheme are demonstrated through numerical simulations conducted on nonlinear systems.

Keywords: Reinforcement Learning, PID control, Adaptive control

1. Introduction

PID control has long been recognized as a highly effective and widely utilized control scheme in various industrial processes and mechanical systems, owing to its versatility, exceptional reliability, and straightforward operational characteristics [1]. When the mathematical model of a controlled plant is unknown, PID controllers offer the advantage of being manually tuned by operators and control engineers using empirical knowledge. However, to enhance performance, classical tuning methods like the Ziegler-Nichols [2] method and Chien-Hrones-Reswich method [3] are often employed in process control, surpassing the outcomes achieved through manual tuning. Although classical tuning methods are effective for simple controlled plants, their performance cannot be guaranteed when dealing with complex systems that exhibit non-linearity, uncertainty, and unknown dynamics. Moreover, since it is often challenging to construct an exact model from real systems, the concept of adaptive PID control has gained significant attention over the past two decades as a means to address these complexities.

Various adaptive PID control strategies have emerged, among which model-based adaptive PID control is prominent that are addressed in [5], [6], [7], adaptive PID control based on neural network [8], [9]. It has been established that model-based adaptive PID control relies on the assumption that the constructed model accurately represents the true dynamics of the plant [10]. Nonetheless, the process of modeling complex systems is often time-consuming and prone to inaccuracies, leading to the possibility of improper adjustment of PID parameters. In contrast, adaptive PID control utilizing neural networks employs supervised learning to optimize network parameters. However, this approach has certain limitations, including the challenge of obtaining a suitable teaching signal and the difficulty of predicting values for unlabeled data. Consequently, with the rapid advancement of computer science, the application of adaptive PID control based on more advanced machine learning technologies has been extensively explored as a potential solution to overcome these limitations.

The control engineering community has witnessed a growing application of machine learning technology across diverse fields that introduced in the [11]. A multitude of algorithms have been developed to address complex control problems, enabling desirable performance and intelligent decision-making. Furthermore, the significant advancements in computing power have facilitated the practical implementation of sophisticated learning algorithms. In the realm of machine learning, Bishop et al. [12] have classified algorithms into three categories: supervised learning, unsupervised learning, and reinforcement learning (RL). RL stands out as distinct from supervised and unsupervised learning [13]. According to a definition provided by [14], RL involves an agent that aims to learn the optimal approach for accomplishing a task by iteratively interacting with its environment. RL has already demonstrated its transformative potential across various applications [15], [16]. From a control perspective, RL refers to an agent (controller) that interacts with a controlled system (environment) and adjusts its control actions (control signal) accordingly [17]. The integration of RL technology with adaptive PID control holds significant promise for process control applications, and numerous studies have investigated this combination [4], [18], [19], [20], [21]. In [4], the reinforcement signal was defined as the error between the current output and the reference signal, potentially leading to prediction losses. [18],

[20], [21] adopted the same updating rule but did not provide the trajectories of PID parameters. Additionally, the updating rule for all three parameters was condensed into a single equation. A model-based design method was presented in [19].

Based on the aforementioned observations, this paper focuses on the development of a PID controller with a new adaptive updating rule based on RL technology for nonlinear systems. The Actor-Critic structure [22], a prominent class of RL technology, is considered as a benchmark in some design methods. This structure involves an actor component responsible for applying control signals to the system, and a critic component that simultaneously evaluates the output's value. Notably, the Actor-Critic structure is widely recognized as the most versatile and successful approach to date [13]. In this study, the concept of implementing an actor and a critic using a Radial Basis Function (RBF) network is explored, which can reduce storage requirements and avoid repetitive calculations. Within the Actor-Critic structure based on the RBF network, a novel adaptive updating rule can be devised.

The main contributions of this study can be summarized as follows. Firstly, the reinforcement signal is redefined to incorporate the one-step predictive output, thereby including the prediction error in the TD error. Secondly, the new adaptive updating rule is derived based on the one-step TD error. Lastly, the proposed scheme adopts a model-free design approach, making it highly suitable for complex real systems.

The structure of this paper is as follows. Section 2 presents the problem formulation, along with the introduction of two reasonable assumptions. In Section 3, the proposed adaptive PID controller based on the Actor-Critic algorithm is described. Section 4 showcases numerical simulations and a comparative study to demonstrate the effectiveness and feasibility of the approach. Lastly, Section 5 concludes the paper.

Nomenclature

Aradius of

Bposition of

Cfurther nomenclature continues down the page inside the text box

1.1 Problem Statement

Let us consider the following discrete-time systems, which are described by nonlinear dynamics in the form of affine state space difference equations

$$x(n+1) = f\big(x(n)\big) + g\big(x(n)\big)u(n)$$

$$y(n) = h(x(n),\ u(n-1)) \tag{1}$$

with state $x(.) \in R^m$, control input $u(.) \in R^n$ and output $y(.)$. Given the nature of RL technology, where detailed model information may be unknown, it is possible to generalize the above system into a more compact form as

$$x(n+1) = F(x(n), u(n))$$

$$y(n) = h(x(n), u(n-1)) \tag{2}$$

It is required to provide two assumptions on the above system in order to capture the ideas about RL technology.

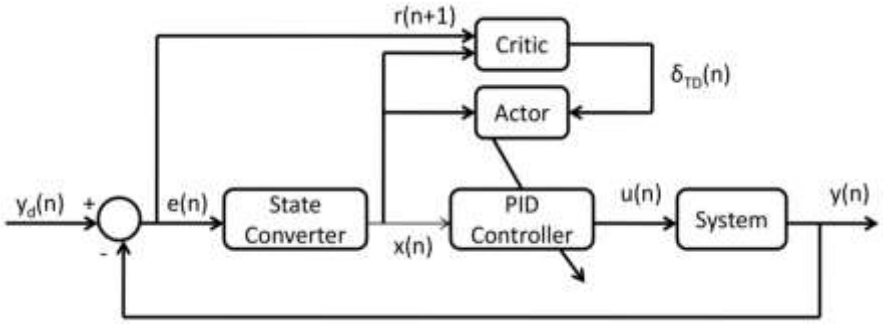

Fig. 1. The block diagram of the proposed scheme

Assumption 2.1: The aforementioned system adheres to the 1-step Markov property as the state at time n + 1 solely relies on the state and inputs at the preceding time n, irrespective of any historical data.

This assumption falls within the domain of Markov decision processes (MDP), which aims to attain a predetermined objective by employing a satisfactory control policy. MDP is formulated in a manner akin to RL technology, thereby playing a pivotal role in integrating control problems with RL technology. MDP serves as a mathematically idealized representation of the RL problem [14].

Assumption 2.2: The sign of partial derivatives of $h(.)$ with respect to all arguments is known, and it is also regarded as the sign of system Jacobian [25].

1.2 Controller Structure

The application of PID controllers to process systems is widely acknowledged, and it is commonly understood that the presence of derivative kick can affect the performance of the closed-loop system. Consequently, this research paper presents a velocity-type PID controller that effectively mitigates the issue of derivative kick:

$$u(n) = u(n-1) + K_I(n)e(n) + K_P(n)\Delta y(n) - K_D(n)\Delta^2(n) \quad (3)$$

that is

$$\Delta u(n) = K(n)\emptyset(n) \quad (4)$$

where, $\emptyset(n)$ is defined as

$$\emptyset(n) := [e(n), -\Delta y(n), -\Delta^2 y(n)]^T \quad (5)$$

and it is regarded as system state. Δ denotes the difference operator defined by $\Delta := 1 - z^{-1}$. The $\Delta^2(n)$ then becomes:

$$\Delta^2 y(n) = y(n) - 2y(n-1) + y(n-2) \quad (6)$$

$K(n) := [K_I(n), K_P(n), K_D(n)]$ is a vector of control parameters. $e(n)$ is the control error and is defined by the difference between reference signal y_d and system output y as follows,

$$y(n) = y_d(n) - y(n) \quad (7)$$

1.3 Objective

In this paper, our objective is to design a PID controller with a novel adaptive updating rule under the Actor-Critic structure. To achieve this, we present a schematic diagram in Fig. 1, illustrating the system state $\emptyset(n)$ construction process. Initially, the system state is constructed based on the input error $e(n)$ and the current system output. These constructed values are then fed into the Actor-Critic structure as inputs.

The Actor component of our method continuously adjusts the controller online using the observed system state throughout the system trajectory. On the other hand, the critic component evaluates the system's performance by receiving both the system state and the reinforcement signal $r(n+1)$.

This evaluation results in the production of the Temporal Difference (TD) error, which is considered a crucial basis for updating the parameters.

By utilizing the TD error, we update the parameters in order to optimize the performance of the PID controller. Overall, our paper focuses on developing a PID controller with an innovative adaptive updating rule within the framework of the Actor-Critic structure.

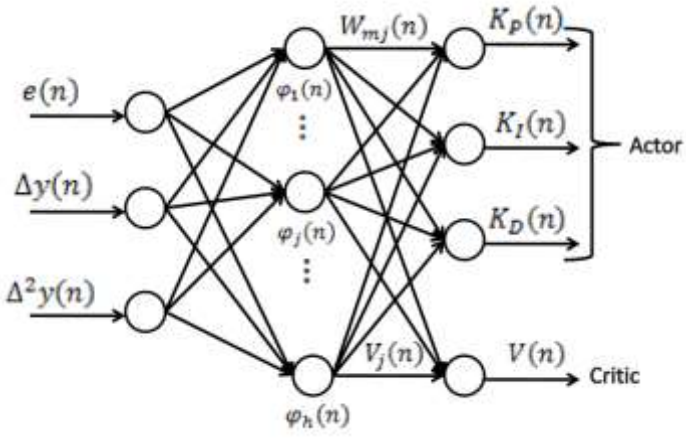

Fig. 2. RBF network topology with Actor-Critic structure

2. Adaptive Controller Design

This section will provide a comprehensive explanation of the proposed algorithm, delving into its intricacies and details.

2.1 Temporal Difference (TD) error

Let's begin by introducing a value function, which is defined as follows:

$$V(n) = \sum_{i=n}^{\infty} \gamma^{i-n} r(x(i),\ u(i)) \quad (8)$$

with $0 < \gamma \leq 1$ a discount factor and u(n) is the control signal. $r(x(i), u(i))$ is called reinforcement signal and can be selected based on a quadratic function. By rewriting eq. (8) as

$$V(n) = r\big(x(n), u(n)\big) + \gamma \sum_{i=n+1}^{\infty} \gamma^{i-(n+1)} r(x(i), u(i)) \quad (9)$$

Rather than computing the infinite sum of the aforementioned equation, one can employ the present control signal u(n) to solve the corresponding discrete difference equation:

$$V(n) = r\big(x(n), u(n)\big) + \gamma V(n+1),\ V(n) = 0 \quad (10)$$

This equation is commonly referred to as the Bellman equation, and leveraging it allows us to define a TD error as the disparity between the two sides:

$$\delta_{TD}(n) = r\big(x(n), u(n)\big) + \gamma V(n+1) - V(n) \quad (11)$$

2.2 Actor-Critic learning based on RBF network

The RBF network has been employed as a method for parameter identification through function mappings. Its simple structure, parameter convergence, and effective learning capabilities are acknowledged as advantages, as discussed in reference [23]. Consequently, in this study, the implementation of the Actor-Critic approach utilizes the RBF network, and its network topology is illustrated in Fig. 2, comprising three-layer neural networks.

The input layer is responsible for gathering the available process measurements and constructing the system states. Within the RBF network topology, this enables the passage of system states to the hidden layers, which are directly shared by both the Actor and the Critic. To generate the control signal u(n) and value function, a straightforward approach is adopted, involving a weighted sum of the function values associated with the units in the hidden layer [24]. The following description provides a more in-depth insight into each layer.

In the input layer, the system state variable x_i (where i represents the input variable index) is incorporated. The input vector $\emptyset(n) \in R^3$ is then forwarded to the hidden layer, where it is utilized to compute the output of the hidden unit.

Within the hidden layer, $\emptyset_j(n)$ represents a vector encompassing the elements $[\emptyset_1(n), \dots\dots, \emptyset_h(n)]$, where *h* corresponds to the number of hidden units. The chosen kernel function for the hidden unit in the RBF network is the Gaussian function. Consequently, the resulting output $\emptyset(n)$ is represented as follows:

$$\emptyset_{\boldsymbol{j}}(\boldsymbol{n}) = \boldsymbol{exp}\left(-\frac{\|\emptyset(\boldsymbol{n}) - \boldsymbol{\mu_j}(\boldsymbol{n})\|^2}{2\sigma_j^2(n)}\right),\ \boldsymbol{j} = \boldsymbol{1, 2, 3, \dots\dots\dots, h} \quad (12)$$

where, μ_j and σ_j are the center vector and width scalar of the unit, respectively. The center vector is defined as follows:

$$\mu_j(n) \coloneqq [\mu_{1j},\ \mu_{2j},\ \mu_{3j}]^T$$

The third layer is called output layer where the outputs of the Actor and the Critic are involved. It should be noted that as mentioned previously the outputs are calculated in a simple and direct way. Therefore, it can yield the PID parameters K(n) in the following:

$$K_{P,I,D}(n) = \sum_{j=1}^{h} w_j^{P,I,D}(n) \emptyset_j(n) \quad (13)$$

with the weights w_{nj} between the j[th] hidden unit and output layer of the Actor. The value function of critic part can be obtained as follows:

$$\boldsymbol{V(n)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{h}} \boldsymbol{V_i(n)} \emptyset_{\boldsymbol{i}}(\boldsymbol{n}) \quad (14)$$

In the context of the Critic, $\boldsymbol{V_i(n)}$ represents the weight connecting the *i*th hidden unit to the output layer.

The different output weights can be fine-tuned using a gradient-based learning algorithm, allowing us to derive an adaptive updating rule based on user-specified parameters. Referring back to Equation (5), the reinforcement signal in this study is defined as

$$\boldsymbol{r\big(x(n), u(n)\big) \coloneqq \frac{1}{2}(y_d(n+1) - y(n+1))^2} \quad (15)$$

This term signifies the distinction between predictive performance and the reference value. As a result, the TD error transforms into

$$\boldsymbol{\delta_{TD}(n) = \frac{1}{2}(y_d(n+1) - y(n+1))^2 + \gamma V(n+1) - V(n)} \quad (16)$$

Consequently, the cost function in this study is represented as follows:

$$\boldsymbol{J(n) = \frac{1}{2}\delta_{TD}^2(n)} \quad (17)$$

Hence, the partial differential equations concerning each output weight of the Actor are formulated as follows:

$$\boldsymbol{w_i^P(n+1) = w_i^P(n) - \alpha_w \frac{\partial J(n)}{\partial w_i^P(n)}} \quad (18)$$

Where $\boldsymbol{\alpha_w}$ is learning rate and

$$\boldsymbol{\frac{\partial J(n)}{\partial w_i^P(n)} = \frac{\partial J(n)}{\partial \delta_{TD}(n)} \cdot \frac{\partial \delta_{TD}(n)}{\partial y(n+1)} \cdot \frac{\partial y(n+1)}{\partial u(n)} \cdot \frac{\partial u(n)}{\partial K_P(n)} \cdot \frac{\partial K_P(n)}{\partial w_i^P(n)} = \delta_{TD}\big(y(n) - y(n-1)\big)\emptyset_i(n)\frac{\partial y(n+1)}{\partial u(n)}} \quad (19)$$

$$\frac{\partial J(n)}{\partial w_i^I(n)} = \frac{\partial J(n)}{\partial \delta_{TD}(n)} \cdot \frac{\partial \delta_{TD}(n)}{\partial y(n+1)} \cdot \frac{\partial y(n+1)}{\partial u(n)} \cdot \frac{\partial u(n)}{\partial K_I(n)} \cdot \frac{\partial K_I(n)}{\partial w_i^I(n)} = -\delta_{TD} e(n) \emptyset_i(n) \frac{\partial y(n+1)}{\partial u(n)} \quad (20)$$

$$\frac{\partial J(n)}{\partial w_i^D(n)} = \frac{\partial J(n)}{\partial \delta_{TD}(n)} \cdot \frac{\partial \delta_{TD}(n)}{\partial y(n+1)} \cdot \frac{\partial y(n+1)}{\partial u(n)} \cdot \frac{\partial u(n)}{\partial K_D(n)} \cdot \frac{\partial K_D(n)}{\partial w_i^D(n)} = \delta_{TD}(y(n) - 2y(n-1) + y(n-2)) \emptyset_i(n) \frac{\partial y(n+1)}{\partial u(n)} \quad (21)$$

It's important to emphasize that prior knowledge of the system Jacobian, denoted as $\partial y(n+1)/\partial u(n)$, is essential for computing the aforementioned equations. In this context, we introduce a relationship $\in = |\in| sign(\in)$, and consequently, the system Jacobian can be derived using the following equation.

$$\frac{\partial y(n+1)}{\partial u(n)} = \left|\frac{\partial y(n+1)}{\partial u(n)}\right| sign\left(\frac{\partial y(n+1)}{\partial u(n)}\right) \quad (22)$$

with $sign(\in) = 1(\in > 0), -1(\in < 0)$. Building upon the aforementioned assumption, we can deduce the sign of the system Jacobian [25]. Moving forward, the updating rule for the output weight of the Critic is as follows:

$$v_i(n+1) = v_i(n) - \alpha_v \frac{\partial J(n)}{\partial v_i(n)} = v_i(n) + \alpha_v \delta_{TD}(n) \emptyset_i(n) \quad (23)$$

where α_v is the learning rate.

The hidden layer's hidden units' centers and widths are updated as follows:

$$\mu_{ij}(n+1) = \mu_{ij}(n) - \alpha_\mu \frac{\partial J(n)}{\partial \mu_{ij}(n)} = \mu_{ij} + \alpha_\mu \delta_{TD}(n) v_j(n) \emptyset_j(n) \frac{\varphi_i(n) - \mu_{ij}(n)}{\sigma_j^2(n)} \quad (24)$$

while

$$\sigma_i(n+1) = \sigma_i(n) - \alpha_\sigma \frac{\partial J(n)}{\partial \sigma_i(n)} = \sigma_i + \alpha_\sigma \delta_{TD}(n) v_i(n) \emptyset_i(n) \frac{\|\varphi_i(n) - \sigma_i(n)\|^2}{\sigma_i^3(n)} \quad (25)$$

where α_μ is the learning rate of center and α_σ is the learning rate of width.

2.3 Algorithm Summery

Algorithm 1 outlines each design step for the proposed adaptive PID controller under the Actor-Critic structure based on the RBF network. To enhance performance, it is essential to provide a clear explanation of the user-specified parameters, as some degree of trial and error may be necessary during the algorithm's implementation.

Algorithm 1: Adaptive PID controller under Actor-Critic based on RBF network

1. At time t = 0, initialize the control input signal $u(0)$ and the reference signal $y_d(n)$.
2. Commence by initializing the parameters $w_j^{P,I,D}(0)$, $v_i(0)$, $\mu_{ij}(0)$, $\sigma_i(0)$ and establish the user-specified learning rates for α_w, α_v, α_μ, and α_σ.
3. **for** t = 1 : EndTime
4. Obtain the system error e(t) by measuring the system output y(t).
5. Calculate the kernel function (12) within the hidden layer.
6. Determine the current PID parameters from equation (4) for the Actor's output, and compute the Critic value function V(t) using equation (14) at time t.
7. Obtain the current control signal by

   $\Delta u(n) = K_I(n) e(n) - K_P(n).\Delta y - K_d(n) \Delta^2 y(n)$
8. Execute the control signal on the controlled system and generate a predictive estimate of the system output y(n + 1).
9. Formulate the system state using the predictive value:

   $\emptyset(n+1) := [e(n+1), \Delta y(n+1), \Delta^2 y(n+1)]^T$.
10. Calculate the value function V (n + 1) from (14).
11. Obtain the TD error $\delta_{TD}(n)$ from (16).
12. Adjust the PID parameter weights using equations (19) through (21), and update the weights of the value function in accordance with equation (23).
13. Revise the centers and widths of the RBF kernel functions using equations (24) through (25).
14. **end for**

3. Numerical Simulations

In this section, we perform numerical simulations and a comparative analysis to assess the effectiveness and viability of the proposed approach. We will examine its performance on the non-linear system described in reference [26].

$$y(n+1) = \frac{y(n)y(n-1)[y(n)+2.5]}{1+y(n)^2+y(n-1)^2} + u(n) + \xi(n) \quad (26)$$

The Gaussian noise, denoted as ξ(n), has a zero mean and a variance of 0.01^2. It's important to mention that due to page limitations, the static characteristics of this nonlinear system are not presented. The reference signal values are defined as follows:

$$\boldsymbol{y_d(n)} = \begin{cases} \mathbf{2.5} & \boldsymbol{for\ 0 \le n} < 100 \\ \mathbf{3.5} & \boldsymbol{for\ 100 \le n} < 200 \\ \mathbf{1} & \boldsymbol{for\ 200 \le n} < 300 \\ \mathbf{3} & \boldsymbol{for\ 300 \le n} < 400 \\ \mathbf{0} & \boldsymbol{otherwise} \end{cases} \quad (27)$$

The user-specified learning rates included in the proposed are summarized as follows:

$$\boldsymbol{\alpha_w = 0.013, \alpha_v = 0.021, \alpha_\mu = 0.0025, \alpha_\sigma = 0.009}$$

and the coefficient γ equals to 0.98. The hidden units in topology RBF network are decided as 3. The initial PID parameters in the proposed scheme are set as

$$K(0) = [0\ 0\ 0]^T$$

In the proposed scheme, no initial value input is required. The simulation results are displayed in Fig. 3, demonstrating the scheme's ability to track the reference signal even under strong non-linearities. Furthermore, the scheme adapts well to changes in the reference signal. Fig. 4 shows the dynamic evolution of the PID parameters, which are updated based on the changing weights. Over time, these parameters tend to stabilize, indicating the effectiveness of the new updating rule within a specific range. The TD error, depicted in Fig. 5, hovers close to zero at steady state.

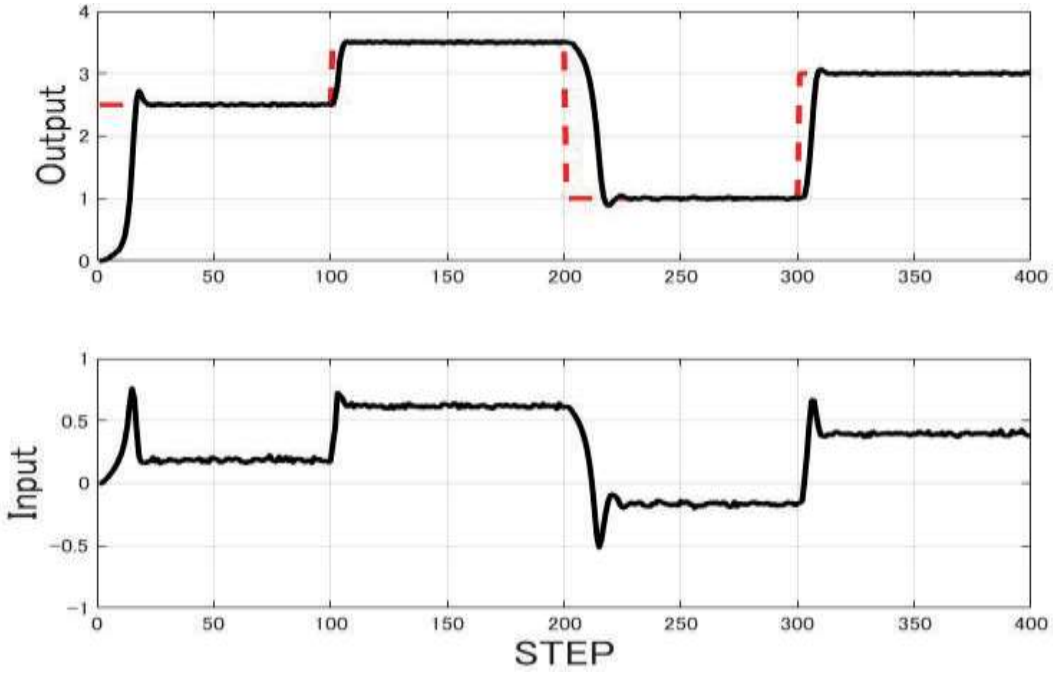

Fig. 3. Control result obtained by the proposed scheme

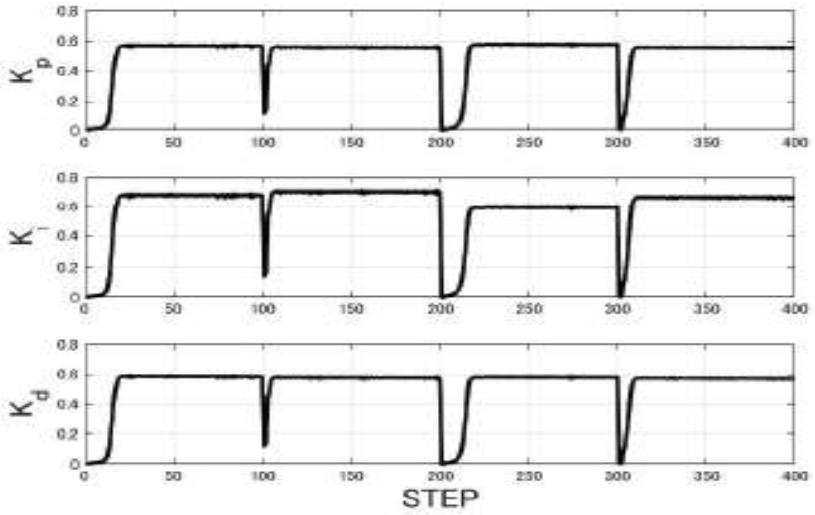

Fig. 4. Trajectories of adaptive PID parameters

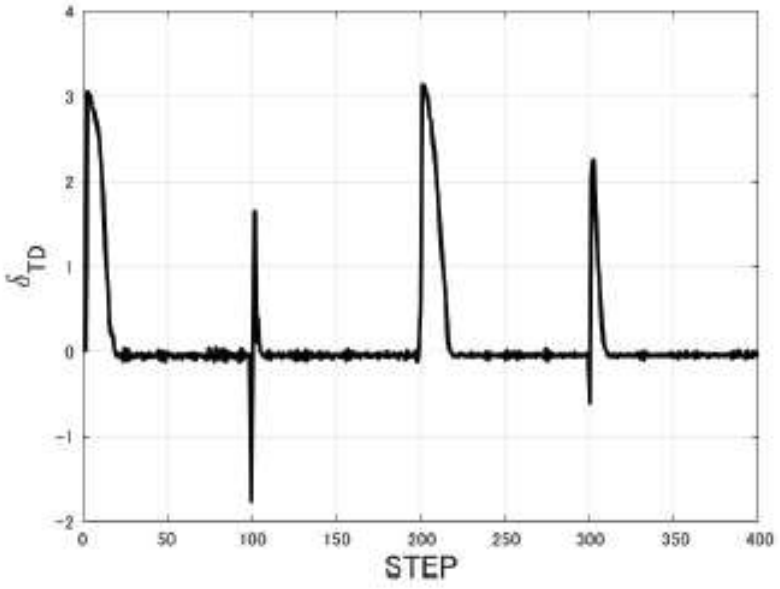

Fig. 5. Trajectories of TD error

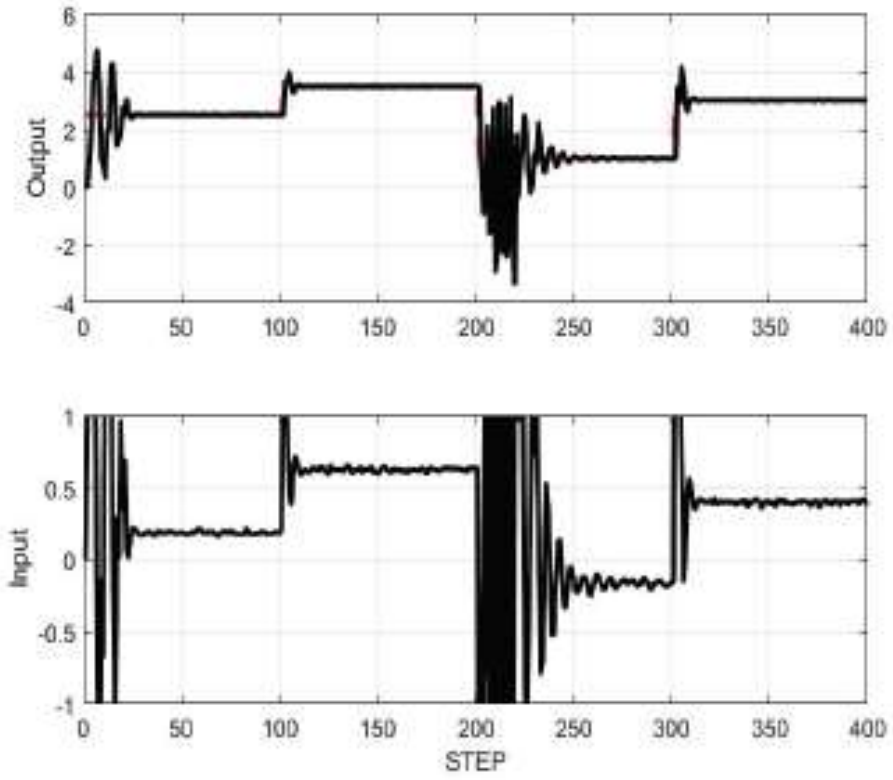

Fig. 6. Control result obtained by the conventional scheme

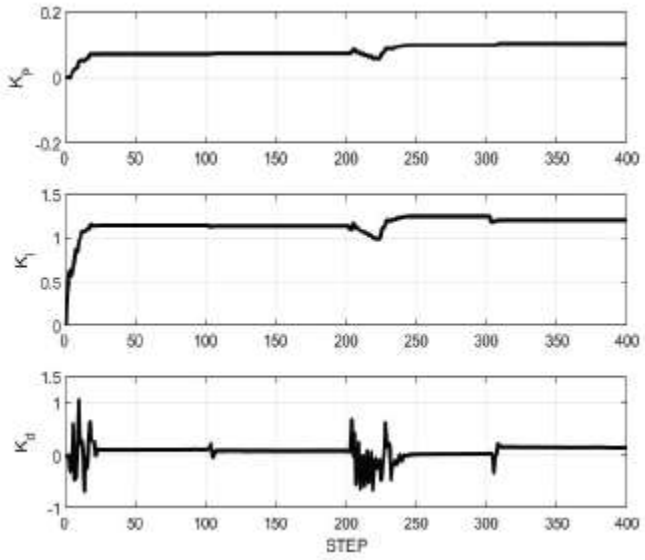

Fig. 7. Control result obtained by the conventional scheme

The comparative analysis of the proposed approach involves the utilization of a traditional adaptive PID tuning technique. The standard gradient method is applied to modify the PID parameters. Control outcomes are illustrated in Figures 6 and 7. Figure 6 demonstrates that the practical tracking issue is successfully addressed, albeit with a more pronounced overshoot compared to the proposed method. This can be attributed to the system's substantial non-linearity.

4. Conclusions

In this paper, a novel adaptive PID controller is examined within the Actor-Critic framework, employing an RBF network for nonlinear systems. An innovative adaptive update rule is introduced for weight adjustments in the network. Initially, a conventional PID controller was integrated with reinforcement learning, utilizing an RBF network, with online PID tuning. The reinforcement signal's determination incorporated predictive output, ensuring precise updates. Furthermore, the RBF network's hidden layer was shared by both the Actor and the Critic, leading to reduced storage requirements and lower computational costs for hidden unit outputs. Notably, initial PID parameters were initialized to zero, eliminating the need for prior knowledge of the controlled system. Subsequently, numerical simulations were conducted to demonstrate the efficiency and feasibility of the proposed approach for complex nonlinear systems, resulting in the stabilization of PID parameters through the novel adaptive update rule. However, a limitation of this method lies in the requirement for user-specified parameters, necessitating empirical tuning within a certain range. An intriguing question pertains to the proper initialization of initial parameters. Additionally, the practical implementation of the proposed approach in a real-world system is necessary to validate its effectiveness.

Acknowledgements

The first author wants to acknowledge his seniors Dr. Parijat Bhowmick and Dr. Abhijit Banerjee for their constant support and inspirations. All authors want to thank Dr. Somnath Maity, Principal of Gargi Memorial Institute of Technology for his constant support and help.

References

[1] K. J. Astrom and T. Hagglund. PID Controllers: Theory, Design and Tuning - 2nd edition. Instrument Society of America, 1995.

[2] J. G. Ziegler and N. B. Nichols. Optimum Settings for Automatic Controllers. Trans. of the ASME, Vol. 64, pp. 759-768, 1942.

[3] K. L. Chien, J. A. Hrones, and J. B. Reswick. On the automatic control of generalized passive systems. Trans. of the ASME, Vol. 74, No. 2, pp. 175-185, 1952.

[4] K. S. Hwang, S. W. Tan, and M. C. Tsai. Reinforcement Learning to Adaptive Control of Nonlinear Systems. IEEE Trans. on Systems, Man, and Cybernetics Part: B Cybernetics, Vol. 33, No. 3, pp. 514-521, 2003.

[5] W. D. Chang, R. C. Hwang and J. G. Hsieh. A multi-variable online adaptive PID controller using auto-tuning neurons. Engineering Application of Artificial Intelligence 16, pp. 57-63, 2003.

[6] T. Yamamoto and S. L. Shah. Design and Experimental Evaluation of a Multivariable Self-Tuning PID Controller. IEEE Proc. of Control Theory and Applications, Vol. 151, No. 5, pp. 645-652, 2004.

[7] D. L. Yu, T. K. Chang and D. W. Yu. A stable self-learning PID control for multi-variable time varying systems. Control Engineering Practice, Vol. 15, No. 12, pp. 1577-1587, 2007.

[8] J. H. Chen and T. C. Huang. Applying neural networks to on-line updated PID controllers for nonlinear process control. Journal of Process Control, Vol. 14, No. 2, pp. 211-230, 2004.

[9] Y. T. Liao, K. Koiwai and T. Yamamoto. Design and implementation of a hierarchical-clustering CMAC PID controller. Asian Journal of Control, Vol. 21, No. 3, pp. 1077-1087, 2019.

[10] Z. S. Hou, R. H. Chi and H. J. Gao. An overview of dynamic linearization based data-driven control and applications. IEEE Trans. on Industrial Electronics, Vol. 64, No. 5, pp. 4076-4090, 2016.

[11] S. Y. Wang, W. Chaovalitwongse, and R. Babuska. Machine Learning Algorithms in Bipedal Robot Control. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, Vol. 42, No. 5, pp. 728-743, 2012.

[12] C. M. Bishop. Pattern Recognition and Machine Learning (Information Science and statistics). Springer-Verlag New York. Inc. Secaucus, NJ, USA, 2006.

[13] J. Shin, T. A. Badgwell, K. H. Liu and J. H. Lee. Reinforcement Learning - Overview of recent progress and implications for process control. Computers and Chemical Engineering, Vol. 127, pp. 282-294, 2019.

[14] R. S. Sutton and A. G. Barto. Reinforcement Learning: An Introduction. MIT Press, 2018.

[15] R. Pinsler, R. Akrour, T. Osa, J. Peters and G. Neumann, Sample and feedback efficient hierarchical reinforcement learning from human preferences. IEEE Int. Conf. Robotics and Automation, 2018.

[16] A. Ferdowsi, U. Challita, W. Saad and N. B. Mandayam, Robust deep reinforcement learning for security and safety in autonomous vehicle systems. Int. Conf. Intelligent Transp. Syst.,, 2018.

[17] F. L. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. IEEE Circuits and Systems Magazine, Vol. 9, No. 3, pp. 32-50, 2009.

[18] X. S. Wang, Y. H. Cheng and W. Sun. A proposal of adaptive PID controller based on reinforcement learning. Journal of China Univ. Mining and Technology, Vol. 17, No. 1, pp. 40-44, 2007.

[19] M. N. Howell and M. C. Best. On-line PID tuning for engine idle-speed control using continuous action reinforcement learning automata. Control Engineering Practice, Vol. 8, pp. 147-154, 2000.

[20] Z. S. Jin, H. C. Li and H. M. Gao. An intelligent weld control strategy based on reinforcement learning approach. The International Journal of Advanced Manufacturing Technology, Vol. 100, pp. 2163-2175, 2019.

[21] M. Sedighizadeh and A. Rezazadeh. Adaptive PID Controller based on Reinforcement Learning for Wind Turbine Control. World Academy of Science, Engineering and Technology 13, pp. 267-262, 2008.

[22] A. G. Barto, R. S. Sutton and C. Anderson. Neuron-like adaptive elements that can solve difficult learning control problems. IEEE Trans. Syst., Man, Cybern., Vol. SMC-13, pp. 834-846, 1983.

[23] Suni V. T. Elanayar and Y. C. Shin. Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems. IEEE Transaction on Neural Network, Vol. 5, No. 4, pp. 584-603, 1994.

[24] J. S. Roger Jang and C. T. Sun. Functional Equivalence Between Radial Basis Function Networks and Fuzzy Inference Systems. IEEE Transaction on Neural Network, Vol. 4, No. 1, pp. 156-159, 1993.

[25] T. Yamamoto, K. Takao and T. Yamada. Design of a Data-Driven PID controller. IEEE Transaction on Control Systems Technology, Vol. 17, No. 1, pp. 29-39, 2009.

[26] K. S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. IEEE Trans. Neural Netw., Vol. 1, No. 1, pp. 4/27-4/27, 1990.

[27] L. Zi-Qiang. On identification of the controlled plants described by the Hammerstein system. IEEE Trans. Automat. Contr., Vol. Ac-39, No. 2, pp. 569-573, 1994.

[28] Zhe Guan and Toru Yamamoto. Design of a Reinforcement Learning PID controller, IEEE International Joint Conference on Neural Networks (IJCNN), 2020.