12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

A Deep Reinforcement Learning-Based PID Tuning Strategy for Nonlinear MIMO Systems with Time-varying Uncertainty

Hao Wang*, Luis A. Ricardez-Sandoval*

**University of Waterloo, Ontario, N2L 3G1 Canada (e-mail: laricard@uwaterloo.ca)*

Abstract: The application of proportional-integral-derivative (PID) control schemes to nonlinear multiple-input, multiple-output (MIMO) systems with time-varying uncertainty is challenging and underexplored. In this study, we formulated a deep Reinforcement Learning (RL) based PID tuning strategy with key novelty in designing an RL agent to achieve real-time adaptive MIMO PID tuning to track setpoints while considering time-varying uncertainty. We evaluated our tuning strategy on a continuous stirred-tank reactor subject to time-varying uncertainty. While conventional PID failed to track the effluent concentration setpoint and caused large errors and offsets, the proposed RL agents achieved fast and accurate setpoint tracking that significantly reduced the errors and eliminated offsets; thus, making our RL-based strategy attractive for chemical engineering applications under time-varying uncertainty.

Keywords: Reinforcement Learning, PID control, MIMO System, time-varying uncertainty, adaptive control

1. INTRODUCTION

Proportional-integral-derivative (PID) controllers remain the most widely adopted and popular controllers in the industry because of their simplicity, robustness, low-cost, and broad applicability with satisfactory performance; particularly at the lower level of the control hierarchy (Carlucho et al., 2020). Consequently, a popular topic is the development of adaptive PID tuning strategies for a variety of control task scenarios.

Conventional PID tuning approaches have been well-developed and mainly categorized into two groups: i) rule-based tuning approaches, e.g., Ziegler-Nichols and Cohen-Coon, and ii) model-based tuning approaches, e.g., Internal Model Control (IMC). However, most practical systems in the industry are nonlinear and time-varying. This makes the conventional PID tuning approaches less applicable as they would require frequent re-tuning whenever there are changes in the system. The problem is even more prominent when attempting to implement a PID control scheme for complex systems, such as nonlinear multiple-input, multiple-output (MIMO) systems under time-varying uncertainty. This problem has prompted the development of adaptive PID tuning approaches for such complex systems.

Studies in developing adaptive PID controllers for MIMO systems with time-varying uncertainty are limited. Gopmandal and Ghosh (2021) and Pradhan and Ghosh (2022) considered the MIMO PID design problem for MIMO systems with norm-bounded time-varying uncertainties by transforming it into a static output feedback problem. However, both studies only considered linear MIMO systems. Also, none of those studies explored the controllability of nonlinear MIMO systems with time-varying uncertainty in the form of time-varying parameters with uncertain coefficients.

Reinforcement Learning (RL) has been extensively and successfully employed in various domains, such as adaptive PID control (Yu et al., 2022) and chemical engineering (Mendiola-Rodriguez and Ricardez-Sandoval, 2022). A few studies have considered RL-based PID tuning methods for MIMO systems. Carlucho et al. (2020) proposed a deep RL-based adaptive MIMO PID tuning approach for low-level control in mobile robots using an inverted deep deterministic policy gradient (IDDPG) algorithm. Wang et al. (2021) introduced an adaptive PID tuning scheme for controlling a MIMO nonlinear six-joint manipulator using a deep deterministic policy gradient (DDPG) algorithm. Yu et al. (2022) applied a model-free self-adaptive SAC-PID control for mobile robots based on the soft actor-critic algorithm. RL-based adaptive PID tuning approaches have also been developed for other MIMO systems, e.g., vapour compression cycles (Ding et al., 2022). However, none of them considered nonlinear MIMO systems under time-varying uncertainty.

The present study formulates a deep RL-based adaptive PID tuning strategy for controlling a class of nonlinear MIMO systems subject to time-varying uncertainty, i.e., time-varying parameters with uncertain coefficients with known bounds. The proposed tuning approach modified the IDDPG algorithm proposed by Carlucho et al. (2020) by designing a tailored RL agent (i.e., state vector, action vector, and reward function) combined with a training procedure for the nonlinear MIMO systems subject to time-varying uncertainty. To the authors' knowledge, this study is the first that tackles the PID tuning challenge for nonlinear MIMO systems with time-varying uncertainty using deep RL. This study is organized as follows. Section 2 constructs the tuning problem of the multiloop PID control scheme for a class of nonlinear MIMO systems with time-varying uncertainty. Section 3 describes the proposed mathematical framework. Section 4 applies the mathematical framework to a case study: a nonlinear MIMO continuous stirred-tank reactor (CSTR) with time-varying uncertainty due to catalyst deactivation and regeneration. Concluding remarks and recommendations for future work are presented at the end.

2. PROBLEM STATEMENT

2.1 MIMO system with time-varying uncertainty

Consider the following nonlinear MIMO system with uncertain time-varying parameters with respect to the states and inputs:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}, t, \boldsymbol{\phi}) \tag{1}$$

$$\boldsymbol{y} = h(\boldsymbol{x}) \tag{2}$$

$$\boldsymbol{\phi} = g(t, \boldsymbol{\alpha}) \tag{3}$$

where $\boldsymbol{x} \in R^{n_x}$, $\dot{\boldsymbol{x}} \in R^{n_x}$, $\boldsymbol{u} \in R^{n_u}$, and $\boldsymbol{y} \in R^{n_y}$ are the state vector, the differential state vector, the input vector, and the output vector, respectively. It is assumed that $n_y \leq n_u$. $\boldsymbol{\phi} \in R^{n_\phi}$ represents the time-varying parameters with an uncertain bounded vector of coefficients $\boldsymbol{\alpha} \in R^{n_\alpha}$. We assume the bound of vector $\boldsymbol{\alpha}$ is known a prior, but their specific realizations are unknown. Examples that lead to such time-varying parameters in the chemical engineering field are fouling on heat transfer surfaces, catalyst deactivation and regeneration, and time-varying kinetic parameters in bioprocesses. These factors often render the processes to behave as non-stationary. $f: R^{n_x} \times R^{n_u} \times R^{n_\phi} \mapsto R^{n_x}$ represents the set of nonlinear differential equations that describe the system's dynamics; $h: R^{n_x} \mapsto R^{n_y}$ and $g: R^{n_\alpha} \mapsto R^{n_\phi}$ denote the sets of algebraic equations for the system's outputs and the time-varying parameters, respectively. The control objective is to conduct setpoint tracking of $\boldsymbol{y}$ regarding a setpoint vector $\boldsymbol{y_{sp}} \in R^{n_y}$ using a multiloop PID control scheme in the presence of time-varying uncertainty.

2.2 Multiloop PID control scheme

It is assumed that multi-loop PID control pairings between the outputs (controlled variables, CVs) and the inputs (manipulated variables, MVs) are defined prior. Hence, there are in total n_y control loops (CV-MV pairings) in response to n_y outputs (CVs). For each control loop i, a PID controller is considered with proportional gain (K_c^i), integral time constant (τ_I^i), and derivative time constant (τ_D^i). Any form of the PID algorithm can be considered. Thus, the PID parameters to be tuned are denoted as a vector $\boldsymbol{k}$, such that $\boldsymbol{k} = (\boldsymbol{k^1}, \boldsymbol{k^2}, \dots, \boldsymbol{k^{n_y}})$, where $\boldsymbol{k^i} = (K_c^i, \tau_I^i, \tau_D^i)$ for $i = 1, 2, \dots, n_y$. Traditionally, $\boldsymbol{k}$ is tuned offline using conventional PID tuning methods such as Zeigler-Nichols or IMC. However, such tuning approaches usually lead to poor control performance due to a lack of real-time response to the non-stationary and nonlinear behaviour of the MIMO system under time-varying uncertainty. One solution is to make use of an adaptive RL-based PID tuning approach, as presented in the next section.

3. MATHEMATICAL FRAMEWORK

In this section, we formulate our mathematical framework based on the IDDPG algorithm proposed by Carlucho et al. (2020) and design our own RL agent to solve the tuning problem we presented in the previous section. The DDPG algorithm is a model-free, off-policy, actor-critic algorithm for continuous action space, and it is inherently suitable for handling uncertainty (Mendiola-Rodriguez and Ricardez-Sandoval, 2022). The IDDPG algorithm is selected since Carlucho et al. (2020) further inverted the critic's gradients in the DDPG algorithm to constrain within bounds the output actions and prevent saturation. Nevertheless, we design our own state vector, action vector, and reward function to adapt our RL agent to the proposed PID tuning task while considering the time-varying uncertainty by incorporating realizations of uncertain coefficients into RL agent training.

3.1 IDDPG algorithm

The IDDPG algorithm is an improved version of the DDPG algorithm; hence, the following components in the IDDPG algorithm are the same as in the DDPG (Lillicrap et al., 2015):

(1) Reply buffer: As an off-policy algorithm, the DDPG uses a reply buffer $\boldsymbol{R}$ to store transitions. At a time t, the transition $(\boldsymbol{s_t}, \boldsymbol{a_t}, r_t, \boldsymbol{s_{t+1}})$ includes the current state vector ($\boldsymbol{s_t}$), the action vector ($\boldsymbol{a_t}$), the reward (r_t), and the next state vector ($\boldsymbol{s_{t+1}}$).

(2) Actor-network: The actor-network is estimated with a deterministic policy $\mu(\boldsymbol{s}|\theta^\mu)$ with weight θ^μ.

(3) Critic-network: The critic-network is estimated with a state-action value function $Q(\boldsymbol{s}, \boldsymbol{a}|\theta^Q)$ with weight θ^Q.

(4) Soft update: To prevent instability during training, the DDPG also employs a target critic-network $Q'(\boldsymbol{s}, \boldsymbol{a}|\theta^{Q'})$ with weight $\theta^{Q'}$ and a target actor-network $\mu'(\boldsymbol{s}|\theta^{\mu'})$ with weight $\theta^{\mu'}$ to soft update the learned networks. The target networks are updated as follows:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} \tag{4}$$

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'} \tag{5}$$

where $\tau \ll 1$

(5) Exploration policy: In the DDPG, the exploration is performed by adding noise $\boldsymbol{\epsilon_{a_t}}$ sampled from a noise process to the actor policy, such that at any time t:

$$\boldsymbol{a_t} = \mu(\boldsymbol{s_t}|\theta^\mu) + \boldsymbol{\epsilon_{a_t}} \tag{6}$$

In addition to the main components of the DDPG discussed above, the IDDPG algorithm further limits the critic gradients by inverting them to keep the actions selected by the actor-network within the specified bounds and prevent saturation. This is formulated as follows (Carlucho et al., 2020):

$$\nabla_a Q^{inverted} = \begin{cases} \nabla_a Q\left(\dfrac{\boldsymbol{a_{max}} - \boldsymbol{a}}{\boldsymbol{a_{max}} - \boldsymbol{a_{min}}}\right), & \forall\ \nabla_a Q > 0 \\ \nabla_a Q\left(\dfrac{\boldsymbol{a} - \boldsymbol{a_{min}}}{\boldsymbol{a_{max}} - \boldsymbol{a_{min}}}\right), & otherwise \end{cases} \tag{7}$$

$$\nabla_a Q = \frac{\partial Q(\boldsymbol{s_t}, \mu(\boldsymbol{s_t}|\theta^\mu))}{\partial \boldsymbol{a}} \tag{8}$$

where $\nabla_a Q$ is the gradient of the state-action value function; $\boldsymbol{a_{max}}$ and $\boldsymbol{a_{min}}$ are the upper and lower bounds of the action vector, respectively. The inverted critic's gradients are implemented into the critic- and actor-networks update procedure as follows. First, randomly sample a minibatch of N transitions $(\boldsymbol{s_j}, \boldsymbol{a_j}, r_j, \boldsymbol{s_{j+1}})$ from the reply buffer $\boldsymbol{R}$. With this minibatch, the critic is updated by minimizing the loss L, i.e.,

$$L = \frac{1}{N}\sum_{j}^{N}(y_j - Q(\boldsymbol{s_j}, \boldsymbol{a_j}|\theta^Q))^2 \quad (9)$$

$$y_j = r_j + \gamma Q'\left(\boldsymbol{s_{j+1}}, \mu'(\boldsymbol{s_{j+1}}|\theta^{\mu'})|\theta^{Q'}\right) \quad (10)$$

where $\gamma \in [0, 1]$ is the discounting factor. Then, obtain the critic's gradients $\nabla_a Q$ using (8) and calculate the inverted gradients $\nabla_a Q^{inverted}$ using (7). The actor policy is then updated using the sampled policy gradient as follows:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_{j}\Big[\nabla_{\boldsymbol{a}} Q^{inverted}(\boldsymbol{s}, \boldsymbol{a}|\theta^Q)|_{\boldsymbol{s}=\boldsymbol{s_j}, \boldsymbol{a}=\mu(\boldsymbol{s_j})} \nabla_{\theta^\mu}\mu(\boldsymbol{s}|\theta^\mu)|_{\boldsymbol{s_j}}\Big] \quad (11)$$

where J is the objective function of the actor function.

3.2 RL agent design

In this section, we present the RL agent's design to address the problem statement presented in section 2, i.e., tuning of the multiloop PID control scheme for the nonlinear MIMO system under time-varying uncertainty. A pseudo-code that illustrates how we incorporate the design of the RL agent into the IDDPG algorithm is presented in Algorithm 1.

Algorithm 1: Proposed RL-based tuning strategy

line	Pseudo-code
1	Random initialize actor μ and critic Q networks with weights θ^μ and θ^Q
2	Initialize target networks μ' and Q' with weights $\theta^{\mu'} \leftarrow \theta^\mu$ and $\theta^{Q'} \leftarrow \theta^Q$
3	Initialize replay buffer $\boldsymbol{R}$
4	**for** episode = 1 to M **do**
5	Initialize a random noise process for exploration
6	Receive initial state $\boldsymbol{s_1}$ and setpoint vector $\boldsymbol{y_{sp}}$
7	Randomly and uniformly sample a vector of uncertain coefficient $\boldsymbol{\alpha}$ from given bounds
8	**for** z = 1 to $t_{max}/\Delta t/P$ **do**
9	Select action $\boldsymbol{a_z}$ based on $\boldsymbol{s_z}$ and $\boldsymbol{\epsilon_{a_z}}$ using (6), then compute $\boldsymbol{k_z}$ using (12)
10	**for** p = 1 to P **do**
11	Execute $\boldsymbol{k_z}$ with $\boldsymbol{\alpha}$ and observe $\boldsymbol{CV_{z_p}}$
12	**end for**
13	Obtain $\boldsymbol{s_{z+1}}$ and calculate r_z
14	Store transition $(\boldsymbol{s_z}, \boldsymbol{a_z}, r_z, \boldsymbol{s_{z+1}})$ in $\boldsymbol{R}$
15	**if** $\|\boldsymbol{R}\| > N$ **then**
16	Sample a random minibatch of N transitions $(\boldsymbol{s_j}, \boldsymbol{a_j}, r_j, \boldsymbol{s_{j+1}})$ from $\boldsymbol{R}$
17	Set y_j using (10)
18	Update critic by minimizing loss L in (9)
19	Obtain critic's gradients $\nabla_a Q$ using (8)
20	Obtain inverted critic's gradients using (7)
21	Update actor using (11)
22	Update target networks using (4) and (5)
23	**end if**
24	**end for**
25	Calculate episodic return $= -\sum_{z=1}^{z=t_{max}/\Delta t/P} r_z$
26	**end for**

In line 7, to consider the time-varying uncertainty in the RL agent's design, a vector $\boldsymbol{\alpha}$ of the uncertain coefficients is sampled randomly and uniformly from their given bounds at the beginning of each training episode. The selected realization in $\boldsymbol{\alpha}$ is then introduced into the environment for RL agent training within that episode (lines 8 to 24). Note that the realizations in $\boldsymbol{\alpha}$ remain constant throughout each episode. To achieve real-time adaptive PID tuning and track the setpoint under time-varying conditions, the agent needs to interact with its environment frequently. As shown in lines 8 to 24, within a training episode with t_{max} simulation time and $t_{max}/\Delta t$ sampling time steps (Δt), the RL agent interacts with the environment every P sampling time steps. Note that z specifies the total number of P sampling steps considered in an episode. As shown in line 9, the agent samples an action vector $\boldsymbol{a_z}$ based on the current state vector $\boldsymbol{s_z}$ and $\boldsymbol{\epsilon_{a_z}}$. Although the IDDPG does not use any probability distribution, a normalized action space with a range of $[-1, 1]$ is used as a convention. Thus, the normalized $\boldsymbol{a_z}$ is used to compute $\boldsymbol{k_z}$ to test the multiloop PID control scheme for the next P sampling time steps. This procedure is as follows:

$$\boldsymbol{k_z} = \left[\underset{-1;1}{clip}\left(\boldsymbol{a_z} + \beta\boldsymbol{\epsilon_{a_z}}\right)\right]\frac{\boldsymbol{k_{max}} - \boldsymbol{k_{min}}}{2} + \frac{\boldsymbol{k_{max}} + \boldsymbol{k_{min}}}{2} \quad (12)$$

$$\boldsymbol{\epsilon_{a_z}} \sim \mathcal{N}(\varphi, \boldsymbol{I}\cdot\sigma) \quad (13)$$

$$\beta = max\,(\beta_{min}, \kappa^{episode\ number}) \quad (14)$$

where $\boldsymbol{k_{max}}$ and $\boldsymbol{k_{min}}$ are the upper and lower bounds of vector $\boldsymbol{k}$. The action noise vector $\boldsymbol{\epsilon_{a_z}}$ is sampled from a Gaussian distribution as it generally provides a higher return than the Ornstein–Uhlenbeck noise process (Hollenstein et al., 2022). φ and σ are the mean and standard deviation of the Gaussian noise. Typically, $\varphi = 0$ and $\sigma = 0.1$. $\boldsymbol{I}$ is the identity matrix. β is an action noise scaling factor and is used for the scheduled reduction of action noise to improve the robustness of the learning process. β decays with the increase in the episode number and a decay factor of κ, until reaching the minimum value β_{min}, as shown in (14). As shown in Algorithm 1, lines 10 to 12, the RL agent executes $\boldsymbol{k_z}$ with the realizations in $\boldsymbol{\alpha}$ for P sampling time steps and records sensor measurements of CVs: $\boldsymbol{CV_{z_1}}$, $\boldsymbol{CV_{z_2}}$, …, $\boldsymbol{CV_{z_p}}$. As shown in line 13, the RL agent obtains the next state vector $\boldsymbol{s_{z+1}}$ and calculates the reward r_z from the past P sampling time steps. The next state vector $\boldsymbol{s_{z+1}}$ is composed of the sensor measurements of CVs in the past P sampling time steps ($\boldsymbol{CV_{z_1}}$, $\boldsymbol{CV_{z_2}}$, …, $\boldsymbol{CV_{z_p}}$), the action vector ($\boldsymbol{a_z}$) at step z, and the last error vector ($\boldsymbol{\varepsilon_{z_p}} = \boldsymbol{y_{sp}} - \boldsymbol{CV_{z_p}}$) to provide the agent with information about the setpoint tracking errors. Note that for the initial state vector $\boldsymbol{s_1}$ (line 6), the initial CVs' conditions in the system were used with an initial action vector based on tuning parameters obtained from conventional PID tuning methods.

The design of the reward function follows a well-known PID controller tuning criterion: the integral of the time-weighted absolute error (ITAE) criterion in the integral error criteria. ITAE is selected as it is conservative and penalizes errors that exist for long periods of time. Accordingly, the reward r_z for every P sampling time step is constructed as follows:

$$r_z = \sum_i^{n_y} \omega_i \int_{t_{z_1}}^{t_{z_P}} t\left|\varepsilon^i(t)\right| dt \quad (15)$$

$$\varepsilon^i = y_{sp}^i - CV^i \quad (16)$$

where ε^i and ω_i are the error and the weighting factor for each control loop, respectively.

4. RESULTS AND DISCUSSIONS

4.1 Cast study

The approach presented in the previous section is tested on a case study that features the control of a MIMO nonlinear non-isothermal CSTR process with a first-order irreversible reaction (A→B) and time-varying activation energy. This MIMO CSTR process was modified from a single-input, single-output CSTR process (Nikravesh et al., 2000).

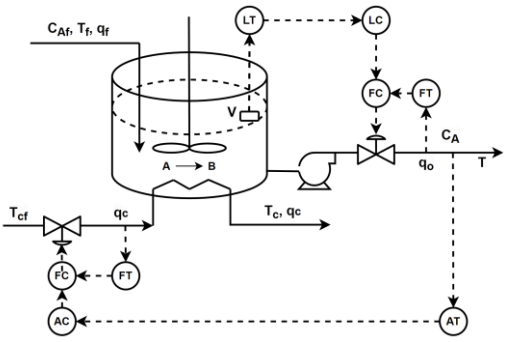

Figure 1. An illustration of the MIMO CSTR process

As shown in Fig. 1, two control loops are considered: the first control loop pairs the effluent concentration (C_A) and the coolant flow rate (q_c), whereas the second control loop pairs the reactor's hold-up (V) with the outlet flowrate (q_o). The objective is to maintain C_A and V to a specific setpoint, $C_{A,sp} = 70\ mol/m^3$ and $V_{sp} = 0.085\ m^3$. Equation (1) is represented as follows in this MIMO CSTR process model:

$$\frac{dV}{dt} = q_f - q_o \quad (17)$$

$$\frac{dC_A}{dt} = \frac{q_f}{V}\left(C_{Af} - C_A\right) - k_0 C_A\, exp\left(-\frac{E}{RT}\right)\phi_c(t) \quad (18)$$

$$\frac{dT}{dt} = \frac{q_f}{V}\left(T_f - T\right) + \frac{(-\Delta H)k_0 C_A}{\rho C_p} exp\left(-\frac{E}{RT}\right)\phi_c(t) + \frac{\rho_c C_{pc}}{\rho C_p V} q_c \left[1 - exp\left(-\frac{hA}{q_c \rho_c C_{pc}}\right)\right] \times (T_{cf} - T) \quad (19)$$

where C_{Af}, q_f, and T_f are the feed concentration, flow rate, and temperature, respectively. T_{cf} and T_c are the coolant inlet and outlet temperatures, respectively. At the nominal operating condition, q_f, q_o, and V are set to $0.001667\ m^3/s$, $0.001667\ m^3/s$, and $0.1\ m^3$, respectively. The rest of the model parameters and nominal conditions for this process can be found in Nikravesh et al. (2000).

In this MIMO CSTR process, the time-varying parameters $\boldsymbol{\phi}$ in (3) are in the form of $\phi_c(t)$ in (18) and (19). The time-varying uncertainty considered in this case study is the time-varying activation energy due to catalyst conditions. There are two catalyst conditions of interest: i) catalyst deactivation due to poisoning with time-varying parameter $\phi_{c,d}(t)$, and ii) catalyst regeneration with time-varying parameter $\phi_{c,r}(t)$. Since theoretical expressions do not exist for the catalyst deactivation and regeneration, empirical correlations are considered (Nikravesh et al., 2000):

$$\phi_{c,d}(t) = exp\left(-\alpha_{c,d}\frac{E}{RT}t\right) \quad (20)$$

$$\phi_{c,r}(t) = exp\left(-\alpha_{c,r}\frac{E}{RT}t\right) \quad (21)$$

where $\alpha_{c,d}$ and $\alpha_{c,r}$ are the deactivation and regeneration constants, respectively, with known specific bounds determined based on the nature of the case study, i.e., $\alpha_{c,d} \in [0.00306, 0.00374]$ and $\alpha_{c,r} \in [-0.003674,\ -0.003006]$.

4.2 Preliminaries

For this case study, the velocity form of the PID algorithm with anti-proportional and anti-derivative kicks was considered. This algorithm is often used in the industry to prevent proportional and derivative kicks due to a sudden step change in the setpoint, as the proportional and derivative kicks may damage the final control element, i.e., control valves. For the i^{th} control loop, this PID algorithm is as follows:

$$\Delta MV^i = K_c^i\left[\left(y_{t-1}^i - y_t^i\right) + \frac{\Delta t}{\tau_I^i}\varepsilon_t^i - \frac{\tau_D^i}{\Delta t}\left(y_t^i - 2y_{t-1}^i + y_{t-2}^i\right)\right] \quad (22)$$

where ΔMV^i is the change in MV compared to the previous output (MV_{t-1}^i).

Before training the RL agent, the tuning parameters of the conventional PID controllers were obtained. The first and second control loops were tuned using the IMC tuning method and the level controller tuning method (Smuts, 2011). Hence, $[K_c^1, \tau_I^1, \tau_D^1] = [190.1, 0.556, 0.827]$ and $[K_c^2, \tau_I^2, \tau_D^2] = [-12.0, 0.4, 0.05]$. These PID tuning parameters were also used as the initial action vector in the initial state vector $\boldsymbol{s_1}$ (line 6 in Algorithm 1). $\boldsymbol{k_{max}}$ and $\boldsymbol{k_{min}}$ of the vector $\boldsymbol{k}$ to be tuned were set to $\boldsymbol{k_{max}} = [[500, 10, 5], [-0.5, 10, 1]]$ and $\boldsymbol{k_{min}} = [[150, 0.01, 0.01], [-15, 0.1, 0.01]]$.

Deep neural networks were used for critic- and actor-networks function approximations in the RL-based strategy. Similar to Carlucho et al. (2020), fully connected layers were used to construct deep neural networks with leaky rectified linear units (ReLU) as activation functions. The neural networks contained two hidden layers with 400 and 300 neurons, respectively. This configuration was chosen as it can handle the complexity of the case study without causing a high computational burden. Actions were introduced to the second layer of the critic-network. Other parameters and hyperparameters used in the RL agent settings were obtained from trial-and-error and are as follows: the critic and actor learning rates were set to 10^{-3} and 10^{-4}, respectively; the sizes of the minibatch and replay buffer were 64 and 10^5, respectively; moreover, γ, τ, κ, and

β_{min} were set to 0.99, 10^{-3}, 0.999, and 0.01, respectively. To match the magnitude difference between effluent concentration and reactor's hold-up, the weighting factors chosen for the first and second control loops in (15) were 0.1 and 1000, respectively.

4.3 RL agent training

Two scenarios were considered, i.e., catalyst deactivation and catalyst regeneration. Hence, an RL agent was trained for each scenario. To account for time-varying uncertainty in the catalyst deactivation scenario, as illustrated in line 7 of Algorithm 1, at the beginning of each training episode, an $\alpha_{c,d}$ value was randomly and uniformly sampled from its given bounds and applied to the MIMO CSTR environment for RL agent training. The realization of $\alpha_{c,d}$ remained constant for the RL agent's training within that episode. The same procedure was used for RL agent training in the catalyst regeneration scenario, except that an $\alpha_{c,r}$ value was sampled from its given bounds instead. To ensure prompt response to the time-varying system while tracking the setpoint, Δt, P, and t_{max} were set to 0.45 seconds, 15, and 1800 seconds (30 minutes), respectively. Each RL agent underwent 50,000 episodes to provide sufficient time for the agent to learn. Both RL agents' training was conducted on a PC with Intel® Core™ i7-9700K CPU @ 3.60 GHz and 64 GB of RAM using TensorFlow 1 in Python. As shown in Fig. 2, RL agents for both scenarios converged after 30,000 training episodes.

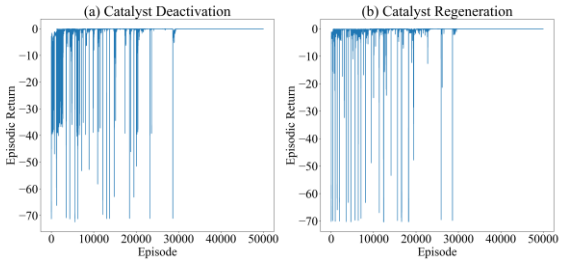

Figure 2. Episodic returns of the two RL agents: (a) catalyst deactivation and (b) catalyst regeneration.

4.4 Validation of proposed PID tuning strategy

The setpoint tracking performances of the RL agent in each of the two scenarios were validated using 1000 testing runs. For the catalyst deactivation scenario, the 1000 testing runs were conducted with different $\alpha_{c,d}$ values randomly and uniformly drawn from their corresponding bounds. Whereas for the catalyst regeneration scenario, different values of $\alpha_{c,r}$ were used instead. The RL agents' setpoint tracking performances are compared with those of conventional PID controllers in terms of ITAEs and the offsets estimated at the end of the simulation time. Table 1 lists the mean and standard deviation of ITAEs over 1000 testing runs under catalyst deactivation and regeneration time-varying scenarios. As shown in Table 1, the RL agents perform significantly better than the conventional PID controllers for the effluent concentration setpoint tracking performances in both time-varying scenarios. In the catalyst deactivation scenario ($\phi_{c,d}$), the mean and standard deviation of ITAEs of the RL agent were reduced by 94.70% and 98.33 %, respectively, compared to conventional PID controllers. Similarly, in the catalyst regeneration scenario ($\phi_{c,r}$), the RL agent reduced the mean and standard deviation of ITAEs by 97.95% and 99.12%, respectively. The large reductions in standard deviations suggest that the RL agents exhibit better generalization capabilities than the conventional PID controllers in the face of time-varying uncertainty.

Table 1. Comparison of setpoint tracking performances in terms of the mean and standard deviation of ITAEs (mean ± standard deviation).

CV	**ITAE for Catalyst Deactivation ($\phi_{c,d}$)**	
	RL-PID	**PID**
Effluent Concentration	132.37 ± 1.37	2496.20 ± 82.23
Reactor's Hold-up	0.7426 ± 0.0001	1.6969 ± 0
CV	**ITAE for Catalyst Regeneration ($\phi_{c,r}$)**	
	RL-PID	**Conventional PID**
Effluent Concentration	148.08 ± 5.90	7224.23 ± 670.41
Reactor's Hold-up	1.5145 ± 0.0189	1.6969 ± 0

On the other hand, the RL agents perform slightly better than conventional PID controllers in both time-varying scenarios for the reactor's hold-up setpoint tracking. Compared to conventional PID controllers, the means of ITAEs of RL agents were reduced by 56.24% and 10.75% in catalyst deactivation and regeneration scenarios, respectively. However, the standard deviations of conventional PID controllers are 0, which suggests that the time-varying uncertainty only affects the first control loop, and there is no interaction between the two control loops. In contrast, the RL agents display very small standard deviations. This behaviour is likely because the RL agents are optimizing for higher rewards that consist of ITAEs for both control loops and lead to trade-offs between better effluent concentration setpoint tracking and better reactor's hold-up setpoint tracking. Note that the ITAEs used to evaluate the reactor's hold-up setpoint tracking performances were magnified by 1000 to match the magnitude difference between the effluent concentration and this output. One instance from each of the 1000 testing runs for catalyst deactivation and regeneration is also illustrated in Fig. 3 and Fig. 4, respectively.

As shown in Fig. 3a and 4a, for conventional PID controllers in both time-varying scenarios, the time-varying uncertainty leads to non-stationary effluent concentration setpoint tracking performances. This behaviour also results in large offsets at the end of the simulation time; particularly for the catalyst regeneration time-varying scenario shown in Fig. 4a. Although some oscillations are observed at the beginning, RL agents in both scenarios are able to track and remain at the setpoint without offsets. On the other hand, for the reactor's hold-up setpoint tracking depicted in Fig. 3b and 4b, the performances of the RL agents are slightly better than conventional PID controllers with faster responses and smaller ITAEs, as shown in Table 1. The superior performance of the RL agents is likely

due to the online adaptation of their PID parameters, as shown in Fig. 3c-d and 4c-d, while the parameters for conventional PID controllers stayed constant.

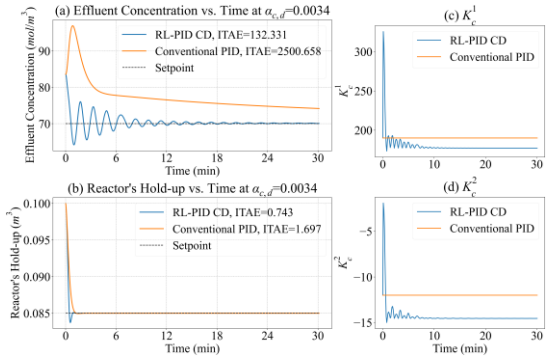

Figure 3. Setpoint tracking performance and PID gains of the RL agent and conventional PID controllers under time-varying catalyst deactivation with $\alpha_{c,d} = 0.0034$.

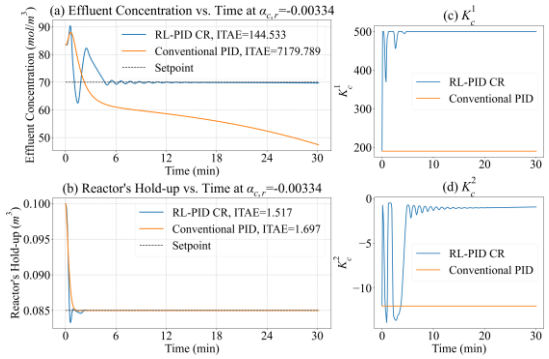

Figure 4. Setpoint tracking performances and PID gains of the RL agent and conventional PID controllers under time-varying catalyst regeneration with $\alpha_{c,r} = -0.00334$.

5. CONCLUSIONS

This study presented a mathematical framework of a deep RL-based tuning strategy to solve the PID tuning problem of a class of nonlinear MIMO systems with time-varying uncertainty in the form of time-varying parameters with uncertain bounded coefficients. The key novelty of this work is the improvements to the IDDPG algorithm through the RL agent design. By designing our own state vector, action vector, and reward function, we can adapt our RL agent to achieve real-time adaptive MIMO PID tuning to track setpoints while addressing the time-varying uncertainty by incorporating it into the RL agent training. A case study was conducted on a nonlinear MIMO CSTR system with two time-varying catalyst scenarios: catalyst deactivation and regeneration. The results indicated that conventional PID controllers failed to address the time-varying uncertainty and led to non-stationary effluent concentration setpoint tracking performances with large ITAEs and offsets. Conversely, the RL agents were able to track the setpoints with small ITAEs, no offsets, and low variability under time-varying uncertainty. Future work includes a comparison of this framework to PID gain scheduling and testing the robustness of the method using more complex MIMO systems with time-varying uncertainty, especially for systems involving strong interactions between different PID control loops.

ACKNOWLEDGMENT

The financial support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the insights by Dr. Qinqin Zhu are gratefully acknowledged.

REFERENCES

Carlucho, I., De Paula, M., and Acosta, G.G. (2020). An adaptive deep reinforcement learning approach for MIMO PID control of mobile robots. *ISA Transactions*, 102, 280–294.

Ding, T.L., Norris, S., and Subiantoro, A. (2022). Adaptive reinforcement learning PI controllers for vapor compression cycle control.

Gopmandal, F. and Ghosh, A. (2021). A hybrid search H∞ based synthesis of static output feedback controllers for uncertain systems with application to multivariable PID control. *International Journal of Robust and Nonlinear Control*, 31(12), 6069-6090.

Hollenstein, J., Auddy, S., Saveriano, M., Renaudo, E., and Piater, J. (2022). Action noise in off-policy deep reinforcement learning: Impact on exploration and performance. *arXiv preprint arXiv:2206.03787.*

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971.*

Mendiola-Rodriguez, T.A. and Ricardez-Sandoval, L.A. (2022). Robust control for anaerobic digestion systems of Tequila vinasses under uncertainty: A deep deterministic policy gradient algorithm. *Digital Chemical Engineering*, 3, 100023.

Nikravesh, M., Farell, A.E., and Stanford, T.G. (2000). Control of nonisothermal CSTR with time varying parameters via dynamic neural network control (DNNC). *Chemical Engineering Journal*, 76(1), 1-16.

Pradhan, J.K. and Ghosh, A. (2022). Multivariable robust proportional-integral-derivative control for linear quadratic compensation of a class of norm-bounded uncertain systems. *Journal of Dynamic Systems, Measurement, and Control*, 144(10), 101003.

Smuts, J.F. (2011). *Process control for practitioners: How to tune PID controllers and optimize control loops*. OptiControls, League City, TX.

Wang, J., Zhu, J., Zou, C., Ou, L., and Yu, X. (2021). Robust adaptive PID control based on reinforcement learning for MIMO nonlinear six-joint manipulator. In *2021 China Automation Congress (CAC)*, 2633-2638. IEEE.

Yu, X., Fan, Y., Xu, S., and Ou, L. (2022). A self-adaptive SAC-PID control approach based on reinforcement learning for mobile robots. *International journal of robust and nonlinear control*, 32(18), 9625–9643.