


Reinforcement Learning in Process Industries: Review and Perspective

Oguzhan Dogru , Junyao Xie , *Member, IEEE*, Om Prakash , Ranjith Chiplunkar , Jansen Soesanto , Hongtian Chen , *Member, IEEE*, Kirubakaran Velswamy , Fadi Ibrahim , and Biao Huang , *Fellow, IEEE*

Abstract—This survey paper provides a review and perspective on intermediate and advanced reinforcement learning (RL) techniques in process industries. It offers a holistic approach by covering all levels of the process control hierarchy. The survey paper presents a comprehensive overview of RL algorithms, including fundamental concepts like Markov decision processes and different approaches to RL, such as value-based, policy-based, and actor-critic methods, while also discussing the relationship between classical control and RL. It further reviews the wide-ranging applications of RL in process industries, such as soft sensors, low-level control, high-level control, distributed process control, fault detection and fault tolerant control, optimization, planning, scheduling, and supply chain. The survey paper discusses the limitations and advantages, trends and new applications, and opportunities and future prospects for RL in process industries. Moreover, it highlights the need for a holistic approach in complex systems due to the growing importance of digitalization in the process industries.

Index Terms—Process control, process systems engineering, reinforcement learning.

I. INTRODUCTION

REINFORCEMENT learning (RL) has emerged as an effective tool for solving complex decision-making problems in a wide range of fields. It has sparked a growing interest in the process industries in recent years, where it has shown promise in optimizing processes, increasing efficiency, and improving safety [1]. Testing RL in simulated environments, laboratory experiments and pilot-scale setups has yielded significant outcomes and brought RL closer to real-world applications. As a result of these outcomes and rapid developments in computational technologies, numerous technology organizations have created and supported various research institutes to accelerate RL research in robotics and language models. Despite these advancements outside process industries, most of the RL methodologies use a combina-

tion of learning and process control techniques extensively studied in the optimization and control of process industries [2]. On the other hand, operational drifts and varying dynamics of processes require modifications in classical techniques and more sophisticated and adaptable solutions. In order to understand and contribute to RL theory in process industries while improving its applicability in real-time, researchers and practitioners should analyze the operational levels in process control holistically [3]. A systematic outline for these levels (the control hierarchy), was initially given in [4] without considering possible faults in the production, supervision and execution levels. Fig. 1 generalizes the existing presentation of the control hierarchy by considering complex applications in the real world, and this study provides representative RL applications for each level.

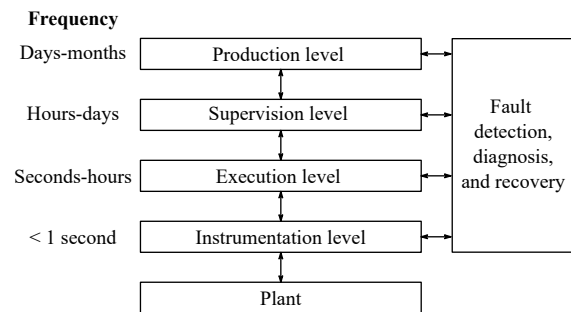


Fig. 1. A schematic of the modern control hierarchy based on [4]. The production level makes high-level decisions based on market demand and political and social events. The supervision level includes real-time optimization of operational targets, control tuning and other hyperparameter optimizations. The execution level observes the process variables and matches the criteria that are defined by the higher levels. These observations are obtained through sensors, and the actuators deliver the control actions to the plant at the instrumentation level.

Despite RL's potential in the control hierarchy, its methodologies face several challenges, including a lack of high-quality data for training, simultaneous learning and control in real-time, interconnectivity and complexity of systems. Moreover, optimal control strategies depend on industries, and finding them through RL involves high risk and cost due to significant trial and error and long-term effects. Nonetheless, it has been reported that several RL methods can solve real-world problems effectively.

Parallel to the ongoing process control and automation research, Industry 5.0 builds upon Industry 4.0, aiming to

Manuscript received July 15, 2023; revised December 6, 2023; accepted December 28, 2023. This work was supported in part by the Natural Sciences Engineering Research Council of Canada (NSERC). Recommended by Associate Editor Qing-Long Han. (*Corresponding author: Biao Huang.*)

Citation: O. Dogru, J. Xie, O. Prakash, R. Chiplunkar, J. Soesanto, H. Chen, K. Velswamy, F. Ibrahim, and B. Huang, "Reinforcement learning in process industries: Review and perspective," *IEEE/CAA J. Autom. Sinica*, vol. 11, no. 2, pp. 283–300, Feb. 2024.

The authors are with the Department of Chemical and Materials Engineering, University of Alberta, Edmonton, T6G 1H9, Canada (e-mail: dogru@ualberta.ca; junyao3@ualberta.ca; om.prakash@ualberta.ca; chiplunk@ualberta.ca; soesanto@ualberta.ca; hc11@ualberta.ca; velswamy@ualberta.ca; fibrahim@ualberta.ca; biao.huang@ualberta.ca).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/JAS.2024.124227

TABLE I
NOMENCLATURE

A2C	Advantage actor-critic	GRU	Gated recurrent units	PDE	Partial differential equation
A3C	Asynchronous advantage actor-critic	HLA/LLA	High/Low level agent	PID	Proportional-integral-derivative
ACER	Actor-critic with experience replay	HVAC	Heating, ventilation, and air conditioning	PIDE	Partial integral differential equation
ACKTR	Actor-critic using Kronecker-factored trust region	IIoT	Industrial Internet-of-Things	PLC	Programmable logic controller
CNN	Convolutional neural network	KL	Kullback-Leibler	PPO	Proximal policy optimization
COD	Chemical oxygen demand	LQR	Linear quadratic regulator	PSE	Process systems engineering
CSTR	Continuous stirred-tank reactor	LSTM	Long short-term memory	PSV	Primary Separation Vessel
DDPG	Deep deterministic policy gradient	LTI	Linear time-invariant	RL	Reinforcement learning
DL	Deep learning	MC	Monte Carlo	RNN	Recurrent neural network
DoS	Denial-of-Service				
DP	Dynamic programming	MDP	Markov decision process	SAC	Soft actor-critic
DPG	Deep policy gradient	MILP	Mixed integer linear programming	SARSA	State-action-reward-state-action
DQN	Deep Q learning	MIMO	Multiple input multiple output	SS	Soft sensor
FD	Fault detection	MINLP	Mixed integer non-linear programming	TD	Temporal difference
FIM	Fisher information matrix	ML	Machine learning	TD3	Twin delayed deep deterministic policy gradient
FTC	Fault tolerant control	MPC	Model predictive controller	TRPO	Trust region policy optimization
GAN	Generative adversarial network	ODE	Ordinary differential equation	VAE	variational auto-encoder

overcome its limitations and prioritize human-machine interaction. Process control, automation, and reinforcement learning are crucial for achieving the goals of Industry 5.0. For example, network automation, a key aspect of Industry 4.0, has supported large-scale Industrial Internet-of-Things (IIoT) and optimized service quality while reducing complexity and cost [5]. However, Industry 5.0 introduces new challenges like cross-layer network optimization and privacy/security protection. Reinforcement learning can address these challenges by enabling comprehensive network optimization and enhancing privacy/security measures. Industry 5.0 also emphasizes societal significance and a human-oriented approach beyond production efficiency and economic goals. SCADA systems have evolved with concepts like event-oriented, data-driven, and model-driven systems. Reinforcement learning can enhance these systems by providing intelligent decision-making capabilities and adaptive control that align with human needs. Supply chain management has transitioned from Supply Chain 4.0 to Supply Chain 5.0, focusing on tailored demand fulfillment and better human-machine relationships. Reinforcement learning is crucial in optimizing supply chain processes, improving demand forecasting, and facilitating effective human-machine collaboration. In summary, reinforcement learning complements process control, automation, and the advancements of Industry 5.0. It enables comprehensive network optimization, adaptive control in SCADA systems, and improved supply chain management. By learning from data and making intelligent decisions, reinforcement learning facilitates human-machine interaction, resilience, sustainability, and societal value in Industry 5.0 [6].

Some studies reviewed machine learning and RL applications in specific domains, including supply chain management [7], process control [8], fault detection and diagnosis [9], [10], etc. However, there is still a lack of a comprehensive

review of the state-of-the-art theory, outstanding challenges in the unified control hierarchy, and possible improvements for practical solutions beyond toy problems and small-scale implementations. A detailed analysis of the methodological development of RL techniques, concerns and obstacles, and future prospects may help researchers and practitioners implement, enhance, and expand the existing accomplishments more rigorously. As a result, the main contributions of this manuscript are to show the developments in deep learning, process control, and RL, introduce advanced RL techniques, present an overview of the recent progress in RL in process industries, and discuss the future prospects in a compact manner. Table I presents the abbreviations used in this manuscript.

The manuscript is organized as follows. Section II progressively introduces the motivation behind RL in process industries, Section III mathematically explores the RL theory, Section IV reviews the recent literature with a focus on soft sensor design, process control, fault detection and diagnosis, fault tolerant control, optimization, planning, scheduling, and supply chain management. Then, Section V discusses the outstanding problems and possible extensions that researchers and professionals of process industries can consider.

Research Methodology: In order to compile the specific literature as extensively as possible, this survey focuses on the relevant keywords in the control hierarchy shown in Fig. 1. An example combination is as follows [“Process control” OR “Fault detection”] AND [“Reinforcement learning”]. After filtering the literature in Google Scholar, Web of Science, Scopus, and IEEE Xplore using these keywords, this article includes peer-reviewed research and review papers written in English. In order to focus on the most recent developments, the survey paper considers the articles published after January 2019 in most of the survey sections (e.g., in process control). However, since this is the first review article that focuses on

the entire control hierarchy, it also considers older articles in sections like automated soft sensor design and fault detection. Moreover, to minimize the overlap and provide a unique resource, the articles covered in the recent related review papers (e.g., in [11], [12]) were excluded from this survey's scope. Fig. 2 presents the number of related articles in the Scopus database to highlight the importance and opportunities in these fields.

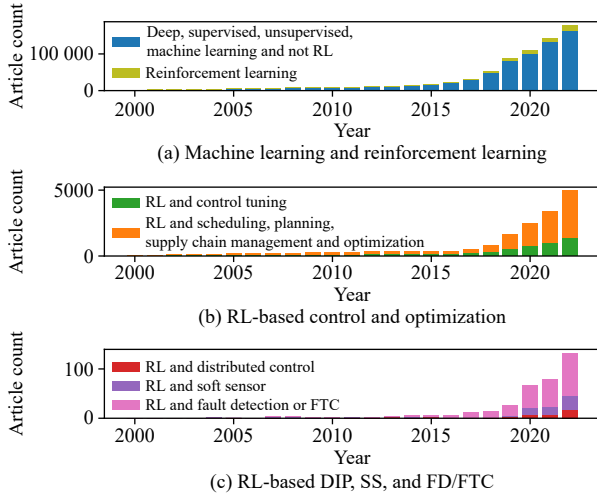


Fig. 2. Scopus literature survey for the related topics. The figure shows an exponential increase in the topics covered in this survey paper since 2000.

II. RECENT INNOVATIONS IN CONTROL AND LEARNING TECHNOLOGIES

A. Advances in Process Control

Process control is a highly interdisciplinary field that has been evolving from the earliest forms of proportional-integral-derivative (PID) controllers to the more sophisticated optimal control schemes. While the PID controllers are effective for low-level, simpler control tasks (execution level in Fig. 1), the optimal control schemes are preferred for complex systems, particularly at the supervisory level. The general optimal control problem is formulated as a cost-minimization problem expressed as follows:

$$\min_u \Phi_0(x(\tau_0), \tau_0) + \int_{\tau_0}^{\tau_f} \zeta(x(\tau), u(\tau), \tau) d\tau + \Phi_\tau(x(\tau_f), \tau_f) \quad (1)$$

where Φ_0 , ζ , and Φ_τ represent the arrival, running, and terminal costs respectively. Variables x and u represent the states and inputs, respectively. The linear quadratic regulator (LQR) and the model predictive controller (MPC) are the most popular optimal control scheme where the cost functions are taken as quadratic functions. The LQR usually solves the optimization problem once in the selected window, whereas the MPC solves the optimization problem in a receding window manner at every time instant. Moreover, LQR assumes a linear model, while the MPC framework can handle both linear and nonlinear system models. Thus, MPC is more widely used as it is more suitable for real process systems. Optimal control schemes, in general, are very versatile and have the ability to

incorporate various system complexities.

One such complexity arises from the fact that modern processes are highly integrated. This leads to complex systems that contain multiple sub-systems which interact with each other. Distributed MPC techniques have been explored in the past couple of decades [13] for such systems. Economic MPC is another widely explored category of MPC that combines the economic objectives of a plant operation with the control objectives [14]. This fuses the production-level decisions and supervisory-level controls depicted in Fig. 1, thus leading to a holistic consideration of various aspects such as economics, time-varying operation, and process dynamics. Robust MPC is another important area that has received substantial attention. This deals with the issues such as model mismatch, uncertainty in the system model, and the presence of unknown disturbances. Some important methods include tube-based MPC [15], stochastic MPC [16], etc.

The standard MPC formulations usually assume the availability of a process model which may be challenging to develop for complex systems. Thus, the recent trend has been implementing MPC based on data-driven techniques for learning system dynamics or controller design [17]. In particular, increasing focus is on approaches that do not learn an explicit parametric model using system identification techniques. Thus, MPC implementations based on approaches such as Gaussian process models [18], direct utilization of historical signal trajectories [19], etc., have been explored. The advent of deep learning has also resulted in the adoption of deep learning into optimal control problems. There has been considerable research on using neural network architectures in MPC for various objectives such as learning the feedback control law [20], and modelling process dynamics [21]. In particular, dynamic network architectures such as recurrent neural networks (RNN), long short-term memory (LSTM), and gated recurrent units (GRU) are commonly used to model process dynamics.

B. Advances in Deep Learning

The modern resurgence in deep learning started in 2006 with works related to greedy layer-wise training using restricted Boltzmann machines [22]. Since then, deep learning has been continuously evolving both in terms of training methods and architectures. The improvements in the training methods include weight initialization, weight regularization, dropout, etc. [23]. Similarly, the architectural improvements have also been significant. Several breakthrough architectures since 2006 include the generative adversarial network, GRU, variational auto-encoder (VAE), residual network, and attention-based neural networks [24]. These developments have vastly improved the fields of computer vision, natural language processing, finance, engineering, etc. Deep learning has also been finding wide acceptance in process industries owing to its ability to model complex nonlinear systems. Deep learning and machine learning, in general, can be broadly classified into three categories: unsupervised learning, supervised learning, and RL. All three facets of deep learning have been utilized in process industries, particularly the unsupervised

and supervised learning techniques, which have been extensively explored.

1) *Unsupervised Learning*: Unsupervised learning has been used in process industries for a variety of process monitoring applications, such as fault detection and diagnosis [25], and fault prognosis [26]. Since modern industrial processes are complex in nature, a deep learning approach is a suitable choice for these tasks. Similar to the general deep learning research, the research for process industrial applications also has been evolving continuously to accommodate more complex structures. These include auto-encoders, GANs, VAE, RNN, and their variants [27], [28], etc. This array of methods spans a wide range of approaches covering the aspects of deterministic and probabilistic methods, and dynamic and static methods. Recently, RL has been explored as an avenue to achieve these traditional unsupervised learning tasks.

2) *Supervised Learning*: As with unsupervised learning, there is extensive research in deep learning for supervised learning applications in process industries. In the process industries, supervised learning is primarily used for tasks such as soft sensing [29] and fault classification. Architectures such as the stacked auto-encoders, LSTM, and, VAE [30], [31] have been explored to achieve these tasks. Besides these traditional nonlinear regression-type tasks, computer vision-based soft sensors also have been developed [32], which rely on CNN to extract essential features from an image for supervised learning.

Both the unsupervised and supervised learning techniques are usually purely data-based and require a vast amount of data to obtain reliable models. RL, on the other hand, is a learning framework where an agent interacts with the system and gets a reward for achieving a certain goal [33]. The agent learns the system information in this process and develops a decision framework to achieve the goals. RL thus can aid the traditional supervised and unsupervised learning tasks where even with the vast amount of data, one still has to manage certain aspects such as the selection of features, selection of network architecture, and dealing with unseen data, etc. Thus, apart from the control applications, RL may also be used to aid the conventional unsupervised and supervised learning tasks in process industries.

C. Advances in Reinforcement Learning

Parallel to the advancements in control and DL, RL has advanced over the past decade, primarily driven by the research institutions supported by the technology industry. Primary developments include but are not limited to the following:

- 1) *Stability* in policy and value function updates [34], [35], which can improve the learning in actor-critic settings,
- 2) *Sample efficient exploration* and *model-based* learning that can speed up learning without requiring lengthy or costly operations [36], [37],
- 3) Interactive RL that utilizes *human feedback* to improve agent's behaviour based on expert knowledge [38].

Although these topics will not be covered in this context, this survey paper will provide insights about leveraging these techniques in process industries. Despite specific advance-

ments in the RL literature (e.g., robust RL, model-based RL, offline RL, etc.), the RL in this article refers to generic online model-free RL, unless otherwise specified. More details about algorithmic improvements in RL have been listed in [39].

D. Relationship Between Classical Control and RL

The theoretical foundations of RL and optimal control originated from the 1950s with dynamic programming (DP). Despite this crucial fact, they were studied separately between the late 1900s and 2010s. For example, the computational science community has focused primarily on the learning stability and sample efficiency of the algorithms on toy problems, while the process industries' main concern was control stability in complex systems. However, learning and control performance often affect each other and should be analyzed jointly. This section connects RL and classical control (specifically MPC) to motivate the necessity of such joint developments. The broad similarities between the two approaches are due to the fact that the stochastic MPC and RL essentially solve the same or a similar objective function. Stochastic MPC minimizes the expected value of the cost function given in (1), while RL maximizes the expected value of a reward function as will be discussed in Section III-A. On the other hand, RL and classical MPC methodologies differ due to several reasons:

- 1) Most RL methodologies utilize episodic learning, whereas MPC focuses on continual calculations.
- 2) Model-free RL [33] uses an explicit model only during training and can operate without a model after training. In contrast, MPC always requires a model.
- 3) General MPC often utilizes a quadratic or linear objective function. This function is used to track a setpoint, reject disturbances, or enhance control stability and smoothness. On the other hand, RL uses a more general reward function that is customized to the particular application. This reward function can be discrete or continuous. As a result, the optimizers that MPC and RL use can differ significantly. Numerous training and evaluation objectives for RL applications will be provided in Table III.
- 4) MPC does not typically take an adaptive form, while RL adapts to environmental changes.
- 5) RL often uses an approximated function to represent the policy, which can describe complex control structures, while MPC relies on a mathematical model, which is less flexible.

The following section discusses the fundamentals of RL to explain these similarities and differences mathematically.

III. REINFORCEMENT LEARNING PARADIGMS AND ALGORITHMS

Unlike MPCs, which control systems using process models directly, machine learning methods require training a neural network based on process data. That is, the data used in MPC is the current feedback from the system, which is an estimate of the state of interest. Although a conventional time-invariant MPC provides a locally optimal controller output at each step, it does not handle variations in the operational conditions since it does not have an update mechanism.

Inspired by model-based dynamic programming [40] and

animal learning, RL provides an alternative framework to optimal control by training an agent in the process environment [33]. The agent optimizes a control policy while interacting with the environment. The policy defines the behaviour of the agent given a state during the interaction. After an action is selected according to the policy, it is implemented in the environment, and the agent receives a reward signal, indicating the goodness of the decision. Although the agent can learn a policy offline (through historical data) [41], this section will cover online/recursive RL algorithms that can learn autonomously without explicit system knowledge or any process model.

This section is divided into four subsections, each explaining a different RL element. Section III-A introduces the concept of Markov decision process (MDP), which is a formalization of sequential decision-making and forms the mathematical foundation of RL. Section III-B presents values-based RL where actions are selected based on their estimated action values. Section III-C highlights policy gradient RL, where the agent directly learns a parameterized policy to select actions. Finally, Section III-D covers various actor-critic-based methods that are also policy-based RL, but they consist of two parameterized functions that work together to select actions.

A. Markov Decision Process

Sequential problems require optimal decisions with temporal order. Considering the complex nature of the environment, the outcomes of the actions of an agent are often uncertain. The MDP is a probabilistic framework for sequential decision-making where the agent observes a state, $x \in \mathcal{X}$, chooses an action $u \in \mathcal{U}$, and receives a reward, r , and the next state, $x' \in \mathcal{X}$. This framework assumes the Markov property, where the information in the next time step depends only on the current time step, as shown in Fig. 3. A transition probability function, $p(x', r|x, u)$, governs the system dynamics. The agent aims to maximize a return function, G , shown in (2).

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad (2)$$

where the capital letters denote that the reward is a stochastic variable, t indicates discrete time steps, and $\gamma \in [0, 1]$ is a weight that controls how much future gains will contribute to the return function. During its interactions with the environment, the agent samples its actions from a stochastic policy, $\pi(u|x)$, the performance of which is tracked using a value function. There are two kinds of value functions: $v(x)$ and $q(x, u)$, and the type depends on the policy evaluation approach.

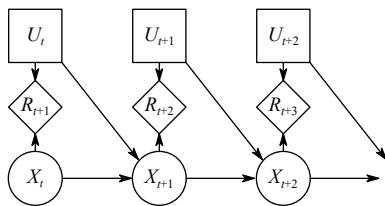


Fig. 3. A graphical representation of the Markov decision process. The next state and the reward depend only on the current state and action. The capital letters indicate that the state, action and reward are random variables.

ach. Learning can be performed by solving the Bellman equations iteratively, as illustrated in (3) and (4) [42].

$$v_{\pi}(x) = \mathbb{E}_{\pi}[G_t|X_t = x] \\ = \sum_u \pi(u|x) \sum_{x'} \sum_r p(x', r|x, u) [r + \gamma v_{\pi}(x')] \quad (3)$$

$$q_{\pi}(x, u) = \mathbb{E}_{\pi}[G_t|X_t = x, U_t = u], \forall x, u \in \mathcal{X} \times \mathcal{U} \\ = \sum_{x'} \sum_r p(x', r|x, u) \left[r + \gamma \sum_{u'} \pi(u'|x') q_{\pi}(x', u') \right] \quad (4)$$

where $\mathbb{E}[\cdot]$ represents the expected value of a random variable. Equations (5) and (6) can be used to find the optimal value functions after the recursions.

$$v^*(x) = \max_{\pi} v_{\pi}(x), \forall x \in \mathcal{X} \quad (5)$$

$$q^*(x, u) = \max_{\pi} q_{\pi}(x, u), \forall x, u \in \mathcal{X} \times \mathcal{U} \\ = \mathbb{E}[R_{t+1} + \gamma v^*(X_{t+1})|X_t = x, U_t = u]. \quad (6)$$

Finally, (7) can be used to calculate the optimum (also known as greedy) policy.

$$\pi^*(x) = \arg \max_u q_{\pi^*}^*(x, u). \quad (7)$$

However, because the system model, $p(\cdot)$, is often unknown, (3) and (4) cannot be solved analytically. To tackle this problem, the agent learns a value (or a policy) function from the data and stores it in memory, whereas, for example, the MPC iteratively optimizes the value function at each step using the system model.

B. Value-Based RL

Initial RL implementations used the value-based (critic-only) methodology [43] to obtain an optimal policy and solve control problems. In these methodologies, actions are derived directly from a value function, which predicts the long-term outcomes of a specific policy. The state value function (shown in (3)) is the expected return obtained from state x while following policy π . The action-value function (shown in (4)) is the expected return after taking action u in state x and following the policy π thereafter. The optimal value functions, $v^*(\cdot)$ and $q^*(\cdot)$ (shown in (5) and (6)) are the unique value functions that maximize the value of every state.

Value-based methodologies, during policy evaluation, estimate $V(x) \approx v_{\pi}(x)$ or $Q(x, u) \approx q_{\pi}(x, u)$ for the current policy and improve the policy iteratively. A common example is greedily selecting actions with respect to the updated value function.

Although dynamic programming (DP) [40] and Monte Carlo (MC) techniques can be used to solve an RL problem, these methods are computationally infeasible, need to wait until an episode ends, have high variance, or require a perfect system model. Temporal difference (TD) methodology provides a practical alternative to these algorithms by combining the bootstrapping (estimating the values by using the previously estimated values) ability of DP and the model-free nature of MC. As a result, TD algorithms can update their policies before an episode ends. Two examples of TD-based

state and action-value function update rules are given in (8) and (9).

$$V(X_t) \leftarrow V(X_t) + \alpha(R_{t+1} + \gamma V(X_{t+1}) - V(X_t)) \quad (8)$$

$$Q(X_t, U_t) \leftarrow Q(X_t, U_t) + \alpha(R_{t+1} + \gamma Q(X_{t+1}, U_{t+1}) - Q(X_t, U_t)) \quad (9)$$

where “ \leftarrow ” represents the update operation, where $(R_{t+1} + \gamma V(X_{t+1}))$ and $V(X_t)$ in TD learning respectively correspond to measurement and prediction in classical control.

Built upon the TD learning methodology, two RL algorithms, namely SARSA (state-action-reward-state-action) and Q-learning [33], have shown promising results in process control applications [11]. These algorithms differ in terms of their policy evaluation/improvement strategies. For example, SARSA is an *on-policy* algorithm since it improves a policy that is used to make decisions. In contrast, the Q-learning algorithm is *off-policy* since it improves a policy different from that used to generate data. The policy improvement step is given by (10). This step shows that the SARSA algorithm uses the next action, U_{t+1} , in the process and updates the Q-function, as shown in (9). However, the Q-learning algorithm performs an additional greedy action selection to find U and updates the action-value (Q) function by using U , as shown in (10).

$$Q(X_t, U_t) \leftarrow Q(X_t, U_t) + \alpha(R_{t+1} + \gamma \max_U Q(X_{t+1}, U) - Q(X_t, U_t)). \quad (10)$$

Since U is not used to control the system (but is used to update the Q-function), it can act as an additional exploration factor in Q-learning. On the other hand, the optimal Q-function can result in aggressive control actions, which can compromise safety, as Sutton and Barto showed [33] through a cliff walking problem. Therefore, the practitioners must select and test the appropriate algorithm considering mathematical and safety requirements.

The terminal state in RL can be defined based on an event or time for process industries. Some examples of the terminal state include but are not limited to an “unsafe” state (e.g., when the simulated temperature of a hydrocracking reactor reaches 10 000 K), a physically impossible state (e.g., in a simulated environment, when the temperature reaches -1 K), a specific time step (e.g., when the plant has been operated for ten hours), and average return value (e.g., when the agent has achieved an integral absolute error of 100 bar). If the hyperparameters are selected appropriately, SARSA and Q-learning algorithms will asymptotically converge to their optimal values [33]. Successful SARSA and Q-learning applications have been reported in [44], [45].

A challenge with SARSA and Q-learning algorithms is that the action value function is stored as a look-up table, where the Q-value is represented explicitly for each state-action pair. On one hand, large discretization steps can reduce the accuracy of the Q-table. However, selecting small discretization steps makes it infeasible to store and update the Q-table for large or continuous state/action spaces. Therefore, for large state/action spaces, a practical solution is to use approximate value functions, such as $V(x|\omega)$ or $Q(x, u|\omega)$, instead of storing $v(x)$ for each state value or $q(x, u)$ for each state-action

pair. The parameters of the value functions are specified by ω in this case. A variety of applications have utilized deep neural networks to train RL agents in large/continuous state spaces [46], [47]. Nevertheless, the value-based RL algorithms often generate discrete and deterministic actions (which can be insufficient for continuous state-action space control problems) and have been reported to be divergent for large-scale problems [48].

C. Policy-Based RL

Process industries often involve large/continuous action spaces with stochastic state transitions. Policy-based (actor-only) methods [49], [50] learn stochastic and continuous actions by parameterizing a policy, $\pi_\theta(u|x, \theta)$, and directly optimizing it by using a performance metric, as demonstrated in (11).

$$J_2(\theta) = \max_\theta \mathbb{E}_{\pi_\theta}[G_t|\theta] \quad (11)$$

where $J_2(\theta)$ is the agent’s objective function, and θ represents the policy weights. As stated in (12), the policy update rule can be obtained by using the policy gradient theorem [33].

$$\Delta\theta = \theta_{t+1} - \theta_t = \alpha G_t \nabla_\theta \ln \pi_\theta(U_t|X_t, \theta_t) \quad (12)$$

where α denotes the learning rate. Although policy gradient methods can converge to at least locally optimal policies, learn continuous actions and “fuzzy” strategies that are a mixture of different actions, and often converge better than the value-based methods, they generally suffer from higher variance than the value-based methods. To reduce the variability during learning, REINFORCE algorithm modifies the policy gradient algorithm as shown in (13) [51].

$$\Delta\theta = \alpha(G_t - b(X_t)) \nabla_\theta \ln \pi_\theta(U_t|X_t, \theta_t) \quad (13)$$

where $b(X_t)$ is a baseline independent on the action, U_t . Sutton *et al.* [52] have modified this methodology further by replacing the actual return with the action value function, $q_\pi(X_t, U_t)$ as shown in (14).

$$\Delta\theta = \alpha(Q_\pi(X_t, U_t) - b(X_t)) \nabla_\theta \ln \pi_\theta(U_t|X_t, \theta_t). \quad (14)$$

Although (12) and (13) update θ in the direction of high return values, α and G_t remain crucial for θ to converge. For example, an indication of convergence is $\Delta\theta \rightarrow 0$ [52]. However, this convergence condition may not be satisfied if $|G_t| > \xi$ or $|\nabla \ln \pi| > \xi$, with ξ being a small threshold value, which can result in non-convergence. Nevertheless, the definition of convergence depends on the application. Moreover, Q_π in (14) is the expected return, which is initially unknown and can take a long time for the agent to learn it. Despite these challenges, policy-gradient methods have been applied to continuous action spaces in various domains.

D. Actor-Critic RL

Similar to a student-teacher pair or generative adversarial networks (GANs) [53] that utilize generative and discriminative networks, actor-critic algorithms generate control actions and examine the outcomes by using a scalar reward signal without any labels [54]. As shown in Fig. 4, these algorithms combine policy and value-based methods via an actor and a critic, respectively. The actor and the critic can be represented as two neural networks, $\pi(u|x, \theta)$ and $V(x|\omega)$ (or $Q(x, u|\omega)$),

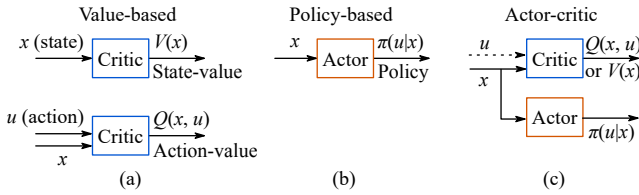


Fig. 4. Comparison of value, policy and actor-critic based RL. The value-based methods derive the policy based on the value functions (which estimate the future return values), the policy-based methods directly optimize the policy, and the actor-critic methods simultaneously learn the policy and the value functions.

respectively. This value-assisted policy learning methodology reduces θ variability while promoting convergence to optimal policies [33], [55]. Although various actor-critic algorithms have been proposed to solve the optimal control/RL problems, an early example combines the policy gradient with state-value estimation [33], as shown in (15).

$$\Delta\theta = \alpha(G_t - V(X_t, \omega)) \nabla_{\theta} \ln \pi_{\theta}(U_t | X_t, \theta_t) \\ = \alpha(R_{t+1} + \gamma V(X_{t+1}, \omega) - V(X_t, \omega)) \nabla_{\theta} \ln \pi_{\theta}(U_t | X_t, \theta_t). \quad (15)$$

Initially, $V(X_t, \omega) \neq (R_{t+1} + \gamma V(X_{t+1}, \omega))$ since ω is often a set of randomly initialized parameters. As a result, high values of $(R_{t+1} + \gamma V(X_{t+1}, \omega))$ will result in large $\Delta\theta$. However, as $V(X_t, \omega)$ approaches $(R_{t+1} + \gamma V(X_{t+1}, \omega))$, the variability in θ decreases over time [56].

This subsection focuses on the most commonly used model-free algorithms that are represented in Table II. Some of these methods use entropy regularization, whereas others take advantage of heuristic methods. A common example of these methods is the ε -greedy approach, where the agent takes a random action with a probability $\varepsilon \in [0, 1]$. $\varepsilon = 1$ corresponds to random search since it learns a policy but does not utilize the learned policy in the decision-making process. Other exploration techniques include but are not limited to introducing additive noise to the action space, introducing noise to the parameter space, utilizing the upper confidence bound, etc. The readers can see [11] for more detail. The actor-critic algorithms are summarized as follows:

1) *Deep Deterministic Policy Gradient (DDPG)*: The DDPG algorithm [57] has been proposed to generalize discrete, low-dimensional value-based approaches [62] to contin-

uous action spaces. This algorithm uses two deep neural networks, namely the deep policy gradient (DPG) and deep Q-learning algorithms, to map the states into actions and estimate the action-value function (Q-function), respectively. The resulting architecture is shown in Fig. 5.

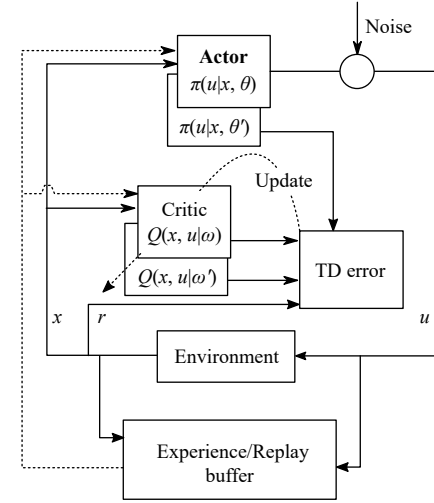


Fig. 5. A schematic of the DDPG algorithm. The solid lines show the data flow, and the dashed lines show the update mechanism.

Similar to the policy update methodology shown in (15), this algorithm updates the policy by using the derivative of the Q-function with respect to ω . This update rule helps the agent maximize the expected return while improving the value estimation and policy. In addition to this improvement, this algorithm utilizes copies of the actor ($\pi(u|x, \theta)$) and critic ($Q(x, u|\omega)$) as target networks ($\pi(u|x, \theta')$, $Q(x, u|\omega')$). After observing a state, real-valued actions are sampled from the actor-network and are mixed with a random process (e.g., Ornstein-Uhlenbeck process, [63]) to encourage exploration as shown in (16).

$$U_t = \pi(U_t | X_t, \theta) + OU_t \quad (16)$$

where OU is the Ornstein-Uhlenbeck process, $dOU = -(OU_{\eta})OU dt + \sigma dW_t$, $OU_{\eta} > 0$ and $\sigma > 0$ are tuning parameters, and W_t is the Wiener process with d representing the ordinary differential symbol. Note that the OU process is a correlated noise, but the use of white noise has also been reported for exploration purposes [61]. The agent stores state, action, and reward samples in an experience replay buffer to break the correlation between consecutive samples to improve learning. It minimizes the mean square error of the loss function to optimize its critic, as shown (for one sample) in (17), and updates the policy parameters using the policy gradient shown in (18).

$$L = (R_t + \gamma Q(X_{t+1}, \pi(X_{t+1} | \theta')) - Q(X_t, U_t, \omega))^2 \quad (17)$$

$$\nabla_{\theta}(Q(X, \theta)) = \nabla_{U_t} Q(X_t, \pi(U_t | X_t, \theta) | \omega) \nabla_{\theta} \pi(X_t | \theta). \quad (18)$$

The target networks are updated using a low-pass filter, as shown in (19).

$$[\omega', \theta']^T \leftarrow \tau_{\text{ddpg}} [\omega, \theta]^T + (1 - \tau_{\text{ddpg}}) [\omega', \theta']^T \quad (19)$$

TABLE II

A COMPARISON OF THE ACTOR-CRITIC ALGORITHMS BASED ON THE TYPE OF ACTION SPACES & THE EXPLORATION METHOD. THE STATE SPACE CAN BE EITHER DISCRETE OR CONTINUOUS FOR ALL ALGORITHMS

Algorithm	Action space	Exploration
DDPG [57]	Continuous	Noisy actions
A2C or A3C [33], [56]	Discrete/Continuous	Entropy regularization
ACER [36]	Discrete/Continuous	Entropy regularization
PPO [58]	Discrete/Continuous	N/A
ACKTR [59]	Discrete/Continuous	N/A
SAC [60]	Continuous	Entropy regularization
TD3 [61]	Continuous	Noisy actions

where τ_{ddpg} is the filter coefficient that adjusts the contribution of the observation (ω and θ , which are a function of the empirical/observed reward) and the previous values of the target parameters. Since the value function is learned for the target policy based on a different behaviour policy, DDPG is an **off-policy** method.

2) *Asynchronous Advantage Actor-Critic (A2C/A3C)*: Instead of storing the experience in a replay buffer that requires memory, this scheme involves local workers that interact with their environments and update a global network asynchronously, as shown in Fig. 6. This update scheme inherently increases exploration since the individual experience of the local workers is independent [56]. Instead of minimizing the error based on the Q function, this scheme minimizes the mean square error of the advantage function (A or δ) for critic update, as shown in (20).

$$A_t = \delta = R_t + V(X_{t+1}|\omega) - V(X_t|\omega). \quad (20)$$

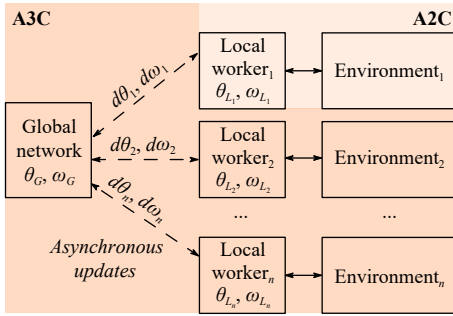


Fig. 6. Multiple worker scheme in the A3C algorithm. Local workers interact with their environment and update a global network. Using a single A3C worker results in an A2C agent.

In this scheme, the global critic is updated by using (21) and the entropy of the policy is used as a regularizer in the actor loss function to increase exploration, as shown below:

$$d\omega_G \leftarrow d\omega_G + \alpha_c \nabla_{\omega_L} \delta(x_t|\omega_L)^2 \quad (21)$$

$$d\theta_G \leftarrow d\theta_G + \alpha_a \nabla_{\theta_L} \delta(x_t|\omega_L) \ln \pi(u_t|x_t, \theta_L) + \beta \pi(u_t|x_t, \theta_L) \ln \pi(u_t|x_t, \theta_L) \quad (22)$$

where initially $d\theta_G = d\omega_G = 0$. α_c and α_a are the learning rates for critic and actor, respectively, and β is a fixed entropy term that is used to encourage exploration. Subscripts L and G stand for the local and the global networks respectively. Multiple workers (A3C) can be used in an off-line manner and the scheme can be reduced to a single worker (A2C) to be implemented online. Even though the workers are independent, they predict the value function based on the behaviour policy of the global network, which makes A3C an **on-policy** method.

3) *Actor-Critic With Experience Replay (ACER)*: ACER was proposed to address sample inefficiency of A3C and improve learning stability [36]. The algorithm utilizes ‘truncated importance sampling with bias correction, trust region policy optimization, stochastic “duelling” network architectures, and the Retrace algorithm [64]. the ACER algorithm modifies the policy update rule shown in (15) for a trajectory $\{X_0, U_0, R_1, \mu(\cdot|X_0), \dots, X_k, U_k, R_{k+1}, \mu(\cdot|X_k)\}$ and calculates

the importance weighted policy shown in (23).

$$\Delta\theta = \left(\prod_{i=0}^k \rho_i \right) \sum_{i=0}^k \left(\sum_{j=0}^k \gamma^j r_{t+1+i} \right) \nabla_{\theta} \log \pi_{\theta}(U_t|X_t) \quad (23)$$

where $\rho_i = \frac{\pi(U_i|X_t)}{\mu(U_i|X_t)}$ is the important weight, and μ and π are the behaviour and the target policies respectively. To reduce the variance, the algorithm estimates the action value of the policy by using the Retrace algorithm shown in (24).

$$Q^{\text{ret}}(s_t, a_t) = R_t + \gamma \bar{\eta}_{t+1} [Q^{\text{ret}}(X_{t+1}, U_{t+1}) - Q(X_{t+1}, U_{t+1})] + \gamma V(X_{t+1}) \quad (24)$$

where the truncated importance weight, $\bar{\eta}_t = \min\{c, \rho_t\}$, and c is a clipping constant. The algorithm updates the actor and critic using the clipped trust region policy optimization technique and the Retrace algorithm shown below:

$$Q^{\text{ret}} = R_{t+1} + \gamma V(X_t|\theta') \quad (25)$$

$$g = \min\{c, \rho_t\} \nabla_{\phi_{\theta'}(X_t)} \log f(U_t|\phi_{\theta'}(X_t)) \times (Q^{\text{ret}}(X_t, U_t) - V_{\omega}(X_t)) + \left[1 - \frac{c}{\rho'_t} \right]_+ (Q(X_t, U'_t|\omega') - V(X_t|\omega')) \times \nabla_{\phi_{\theta'}(X_t)} \log f(U'_t|\phi_{\theta'}(X_t)) \quad (26)$$

$$k = \nabla_{\phi_{\theta'}(X_t)} D_{KL}[f(\cdot|\phi_{\theta'}(X_t))|f(\cdot|\phi_{\theta'}(X_t))] \quad (27)$$

$$\Delta\theta = \nabla_{\theta'} \phi_{\theta'}(x) \left(g - \max \left\{ 0, \frac{k^T g - \delta}{\|k\|_2^2} \right\} k \right) \quad (28)$$

$$\Delta\omega' = (Q^{\text{ret}} - Q(X_t, U_t|\omega')) \nabla_{\omega'} Q(X_t, U_t|\omega') + \min\{1, \rho_t\} (Q^{\text{ret}}(X_t, U_t) - Q(X_t, U_t|\omega')) \times \nabla_{\omega'} V(X_t|\omega') \quad (29)$$

where f represents a sampling distribution, ϕ is a neural network that generates the statistics of f , $\rho'_t = \frac{f(U'_t|\phi_{\theta'}(X_t))}{\mu(U'_t|X_t)}$, and D_{KL} is the KL divergence. Because of its Retrace algorithm, ACER is an off-policy method.

4) *Proximal Policy Optimization (PPO)*: The trust region policy optimization (TRPO) algorithm suggests maximizing a soft-constrained objective function shown in (30),

$$\max_{\theta} J^{\text{KL}} = \max_{\theta} \mathbb{E} \left[\frac{\pi_{\theta}(U_t|X_t)}{\pi_{\theta_{\text{old}}}(U_t|X_t)} A_t - \beta D_{KL}[\pi_{\theta_{\text{old}}}(U_t|X_t)|\pi_{\theta}(U_t|X_t)] \right] \quad (30)$$

where β is a weight for the KL divergence term, and θ_{old} represents the old policy parameters. A_t is the advantage estimate that represents how good the agent’s actions are, as shown in (20), and the KL divergence creates a lower bound on the performance of π . However, using a fixed weight, β , in the objective function shown in (30), can result in large policy updates. To avoid abrupt changes in the policy, the PPO algorithm suggests [58] clipping the surrogate objective function as shown in (31).

$$J^{\text{CLIP}}(\theta) = \mathbb{E}[\min(r(\theta)A_t, \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)A_t)] \quad (31)$$

where $r(\theta) = \frac{\pi_{\theta}(u|x)}{\pi_{\theta_{old}}(u|x)}$, and ϵ is the clipping constant. Then, the algorithm includes a KL penalty and a value loss function in the objective function, which results in (32)

$$J^{\text{PPO}} = \mathbb{E}[J^{\text{CLIP}} - c_1 J^{\text{VF}} + c_2 \mathcal{H}] \quad (32)$$

where c_1 and c_2 are weight coefficients, $\mathcal{H} = \mathbb{E}[-\log \pi(U_t|x_t, \theta)]$, and $J^{\text{VF}} = (V(X_t|\omega) - V(X_t|\omega'))^2$. The resulting architecture of the PPO algorithm is shown in Fig. 7. Due to the practical advantages of these modifications, PPO and its variants have been one of the most commonly used algorithms to solve control problems [65].

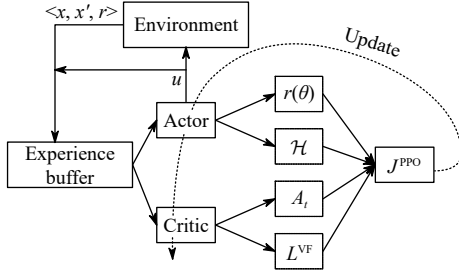


Fig. 7. A schematic of the PPO algorithm. The solid lines show the data flow, and the dashed lines show the update mechanism.

5) *Actor-Critic Using Kronecker-Factored Trust Region (ACKTR)*: Classical gradient descent/ascent algorithms, such as the general policy gradient algorithm (shown in (15)), update the parameters by solving the optimization function shown in (33).

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \Psi^t \nabla_{\theta} \log \pi(U_t|x_t|\theta) \right] \quad (33)$$

where Ψ is the magnitude of the update, which is often selected as G_t , $Q(X_t, U_t|\omega)$, or A_t . Such RL algorithms aim to maximize a non-convex function, $J(\theta)$, in the steepest ascent direction while calculating $\Delta\theta$ such that $J(\theta + \Delta\theta)$ is maximized. In general, the goal is to keep $\|\Delta\theta\|_z = (\Delta\theta^T z \Delta\theta)^{0.5} < 1$, and z is a positive semidefinite matrix. The result of this optimization problem is in the form of $\Delta\theta \propto z^{-1} \nabla_{\theta} J(\theta)$, with $z = I$, as shown in (15). Instead of a gradient descent algorithm to optimize the actor and the critic networks, ACKTR [66] utilizes second-order optimization, which provides more information during learning. That is, $z = F \neq I$, where F is the Fisher information matrix, which is challenging to calculate, store and invert. ACKTR overcomes the computational complexity by using Kronocker-factored approximation [59] to approximate the inverse of Fisher information matrix (FIM) that, otherwise, scales exponentially with respect to the neural network parameters. Moreover, ACKTR keeps track of the Fisher statistics, which yields better curvature estimates. The resulting algorithm updates the parameters as shown in (34).

$$\Delta\theta = \alpha F^{-1} \nabla_{\theta} \delta \quad (34)$$

where $F = \mathbb{E}_{p(\tau)} [\nabla_{\theta} \log \pi(U_t|x_t, \theta) (\nabla_{\theta} \log \pi(U_t|x_t, \theta))^T]$, and τ is shown in (35).

$$\tau = p(X_0) \prod_{t=0}^T \pi(U_t|x_t) p(X_{t+1}|x_t, U_t) \quad (35)$$

where p denotes the probability distribution functions. As a result of these improvements, ACKTR has shown successful results in various applications [67].

6) *Soft Actor-Critic (SAC)*: Unlike methods such as A3C and PPO, which use the entropy of the policy as a loss regularizer [56], [58], [68], SAC augments the reward function with the entropy term (as shown in (36)) to encourage exploration while maintaining learning stability.

$$J(\theta) = \sum_{t \in T} \mathbb{E}_{(x_t, u_t) \sim \pi} [R(x_t, u_t) + \alpha \mathcal{H}(\pi_{\theta}(\cdot))] \quad (36)$$

where θ represents the parameters of the policy, and α is a user-defined (fixed or time-varying) weight to adjust the contribution of the entropy, \mathcal{H} . This scheme relies on both $Q(\cdot, \phi)$ and $V(\cdot, \omega)$ functions to utilize the soft-policy iteration. The parameters of the neural networks are updated, as shown in (37)–(39).

$$\begin{aligned} \nabla_{\omega} J_V(\omega) &= \mathbb{E}_{X_t \sim \mathcal{D}} [0.5(V(X_t|\omega) \\ &\quad - \mathbb{E}_{U_t \sim \pi_{\theta}} [Q_{\phi}(X_t, U_t) - \log \pi_{\theta}(U_t|x_t)])^2] \\ &= \nabla_{\omega} V(X_t|\omega) (V(X_t|\omega) - Q_{\phi}(X_t, U_t) \\ &\quad - \log \pi_{\theta}(U_t|x_t)) \end{aligned} \quad (37)$$

$$\begin{aligned} \nabla_{\phi} J_Q(\phi) &= \mathbb{E}_{(X_t, U_t) \sim \mathcal{D}} [0.5(Q_{\phi}(X_t, U_t) - \hat{Q}(X_t, U_t))^2] \\ &= \nabla_{\phi} Q_{\phi}(X_t, U_t) (Q_{\phi}(X_t, U_t) \\ &\quad - R_{t+1} - \gamma V_{\omega'}(X_{t+1})) \end{aligned} \quad (38)$$

$$\begin{aligned} \nabla_{\theta} J_{\pi}(\theta) &= \nabla_{\theta} \log \pi_{\theta}(X_t|U_t) + (\nabla_{U_t} \log \pi_{\theta}(X_t|U_t) \\ &\quad - \nabla_{U_t} Q(X_t, U_t)) \nabla_{\theta} f_{\theta}(\epsilon|x_t) \end{aligned} \quad (39)$$

where U_t is evaluated at $f_{\theta}(\epsilon|x_t)$, ϵ is a noise vector, and $\hat{Q}(X_t, U_t) = R_{t+1} + \gamma \mathbb{E}_{X_{t+1}} [V_{\omega'}(X_{t+1})]$. Similar to DDPG and PPO, SAC stores the transitions in a replay buffer, indicated as \mathcal{D} , to address sample efficiency. This approach has also been reported in [55], [60] to improve the robustness of the policy against model and estimation errors. Besides enhancing the exploration, this off-policy training methodology has been used in several control and optimization applications [69] and reported to improve stability since it utilizes target networks.

7) *Twin Delayed Deep Deterministic Policy Gradient (TD3)*: TD3 is an extension to the DDPG algorithm [61]. It addresses error propagation (which is a non-trivial challenge in statistics and control) due to function approximation and **bootstrapping** (i.e., instead of an exact value, using an estimated value in the update step). To reduce the overestimation bias, the scheme predicts two separate action-value functions and prefers the pessimistic value to update the network parameters, avoiding sub-optimal policies. TD3 utilizes target networks, delays the update to the policy function, and uses an average target value estimate by sampling N -transitions from a replay buffer to reduce variance during learning. The scheme introduces exploration by adding Gaussian noise to the sampled actions and performs policy updates using the deterministic policy gradient [49]. As a result of these modifications,

TD3 has been considered one of the state-of-the-art RL algorithms in control and optimization [70], [71].

IV. RL APPLICATIONS IN PROCESS INDUSTRIES

Now that various RL algorithms have reviewed, this section presents applications of RL in various activities relevant to process industries.

A. Soft Sensors

Soft sensors, also known as inferential or virtual sensors, are widely used for quality estimation, process monitoring, and feedback control [29]. When building a data-driven soft sensor model, there are mainly four procedures, including data selection, model selection, model training and validation, and model maintenance. Typically, data-driven soft sensor models individually consider the aforementioned modelling procedures and process knowledge is applied to each procedure separately and in a non-autonomous manner. Moreover, most existing data-driven soft sensor methods are based on traditional statistical methods and/or (semi) supervised learning methods that eventually lead to specific regression modelling problems.

Compared to traditional statistical methods and/or (semi) supervised machine learning-based data-driven soft sensor methods, RL enables autonomous soft sensor design. Thanks to the nature of the MDP, RL can be naturally applied to address dynamic soft sensor design problems. The crucial aspects of soft sensor modelling tasks, including data selection and regression model training, validation, and maintenance, may be addressed as a sequential decision-making problem via RL. Recently, one interesting work proposed a framework based on deep RL for autonomous data selection and soft sensor modelling [72], where the authors formulated the soft sensor as an MDP problem and showed that the proposed RL methods outperform some commonly utilized data-driven soft sensor methods, including statistical methods, transfer learning methods, and just-in-time learning methods. Reinforcement learning and particle filtering were jointly used in [73] for remaining useful life estimation of sensor-monitored degrading systems. In addition, [74] suggested using soft sensing techniques for quality prediction and data collection in RL-based control optimization of wastewater treatment processes. To the best of the authors' knowledge, the research results in this direction are very limited. The applications of RL in autonomous soft sensing deserve future research.

B. Process Control

As motivated in Fig. 1, the primary process control (after instrumentation and monitoring) occurs in the Execution and Supervision levels, where the controllers drive the controlled variables to the desired operating points. Successful RL implementations can cater to maintaining safe, optimal and environmentally-friendly operations if these desired points are optimally determined and if the controller parameters are selected appropriately. This section divides process control into two levels. Low-level control refers to the controllers at the Execution level (e.g., PID, LQR, and MPC), high-level control refers to tuning these controllers' hyperparameters

(e.g., PID parameters, MPC parameters).

1) *Low-Level Control*: Classical control theory has limitations in achieving complex desired behaviours, even with variants of the quadratic cost function. To address this, the model predictive control (MPC) cost function can be automatically learned through inverse reinforcement learning (IRL). This method could particularly be useful for large-scale systems with intricate interactions, as demonstrated in [75].

Traditional controllers depend on accurate system models to achieve optimal performance, which may not always be available. In contrast, RL offers more adaptable and versatile solutions. In [76], RL and MPC were combined to create an optimal controller that addresses safety and stability issues in linear systems. The resulting controller provides better adaptability and reliability than traditional approaches.

Safety is a crucial consideration when implementing RL agents, especially in process industries where operational variables are critical. State constraints can be added to the RL learning objective to maintain safety, as shown in [77]. Additionally, input constraints proposed in [78] can promote operational safety by limiting the actions of the RL agent.

Additionally, it was demonstrated that multi-agent RL systems could be trained to control multiloop processes in a feed-forward feedback configuration to promote closed-loop stability and disturbance rejection, which can significantly enhance control performance [79]. However, the potential benefits of multi-agent RL systems come with the cost of potentially time-consuming training processes. Despite this, the benefits of multi-agent RL systems in promoting closed-loop stability and disturbance rejection make them a promising area of research for researchers in process control and process automation.

Another promising approach for heating, ventilation, and air conditioning (HVAC) system control combined deep learning and MPC to utilize the benefits of modern and classical control techniques [80]. This algorithm effectively improves system performance by assigning the update target as the cumulative reward in the prediction horizon of MPC. The proposed method has been validated through a case study involving a two-zone data center in a simulated environment, where it achieves the largest average rewards with four weather data sets out of five. These results demonstrate the potential for this approach to enhance HVAC system performance, making it a valuable addition to the toolkit of process industries.

A novel RL-based approximate optimal control method that guarantees state-constraint handling abilities on attitude forbidden zones and angular velocity limits has been proposed in [81]. In this work, barrier functions were used to encode constraint information into the cost function, and a simplified critic-only neural network was used to replace the conventional actor-critic structure. The authors also demonstrated promising estimation error given limited excitation instead of persistent excitation. The effectiveness and advantages of the proposed controller were verified by numerical simulations and experimental tests.

In addition to these developments in the low-level control settings, the controller performance often depends on parameter tuning, also known as high-level control, as will be dis-

cussed in the following subsection.

2) *High-Level Control*: A meta-RL approach to tuning fixed structure controllers in closed-loop without explicit system identification has been presented in [82]. It was shown that the proposed algorithm was able to tune fixed-structure process controllers online with no process-specific training and no process model. Despite this algorithm's incremental nature, which can result in divergent control performance, the authors proved the concept of control tuning using meta-learning schemes.

Another model-free goal-based algorithm for self-tuning MIMO PID systems for real-time mobile robots has been presented in [83]. The proposed adaptive architecture allows for a direct representation of the state space, and the network used for parameterizing the policy function is able to directly output the parameters of the MIMO system without any reparameterization.

In addition, a contextual bandit-based approach for PID tuning has been proposed in [84] to handle parameter-varying and nonlinear systems. This approach first trains an agent on a step response model to address sample efficiency and then fine-tunes the agent to learn the model-plant mismatch online. This type of learning can benefit industries with thousands of PID loops where the systems cannot be excited persistently.

An automated tuning framework for the weights of nonlinear MPC to reduce the time and effort of users has been proposed in [85]. The proposed agent was implemented in two stages. In the first stage, the agent explored the weights that could maintain the vehicle position at the desired points. In the second stage, the agent was trained to find the best parameters for trajectory tracking. The empirical results validated computational efficiency, direct applicability, and satisfactory trajectory tracking performance. Therefore, this two-step methodology can be helpful for practical implementations in process industries that involve setpoint tracking problems. Similarly, a dynamic weights-varying MPC was developed to autonomously learn a policy for online weight modifications of an MPC cost function [86]. This approach used deep RL and allowed users to specify additional objectives in a multi-objective cascaded Gaussian reward function. The agent observed the low-level states and their deviations from their setpoints while outputting the controller weight parameters. Empirical findings showed that the proposed methodology outperformed a manually-tuned MPC. In addition to the weights of controllers, finding appropriate lengths of prediction and control horizons is challenging. It was demonstrated that for simple systems, RL could be used to automatically modify and adapt the prediction horizon of the MPC controllers using only a few minutes of data collection [87]. These tuning methods have shown that RL could be used for autonomous controller optimization.

C. Distributed Industrial Processes and Control

Distributed industrial processes, also known as distributed parameter systems, are often considered another major class of industrial processes in addition to lumped parameter industrial processes. Compared to lumped parameter processes modelled by ordinary differential equations (ODE), as cov-

ered in the earlier sections, distributed parameter processes are modelled by partial differential equations (PDE) or partial integral-differential equations (PIDE), which makes them capable of describing spatial-temporal dynamics in numerous industrial applications, e.g., processes that involve chemical reactions, heat transfer, crystal growth, or irrigation, fluid dynamic and flexible mechanical systems (e.g., flexible aircraft wing design) utilize PDEs and PIDEs to describe the system dynamics comprehensively [88]. However, due to the complexity of PDE (or PIDE) models, applications of RL in distributed parameter processes are more challenging and complex than that in ODE-based processes from both theoretical and practical viewpoints.

Over the past decades, various contributions have been reported in the literature on RL-based control synthesis of distributed parameter systems. An adaptive-critic-based optimal neuro control strategy was proposed for distributed parameter systems in [89], where an approximate discrete dynamic programming was used for the problem formulation, and necessary conditions of optimality were derived. By using proper orthogonal decomposition and approximate dynamic programming, a single-network-adaptive-critic scheme was designed (as a stabilizing state-feedback controller) to control the heave dynamics of a flexible aircraft wing in [90]. Based on the empirical eigenfunctions (computed from the Karhunen-Loeve decomposition method) and neural networks, approximate optimal controller designs were considered for dissipative PDE systems [91] and one-dimensional parabolic PDE systems [92]. Apart from optimal control designs, various RL-based methods have been applied to address other types of control problems, including an off-policy RL method developed for the data-driven H_∞ control in [93], a convolutional RL framework proposed for distributed stabilizing feedback controller design in [94], an off-policy integral RL algorithm proposed for dealing with the nonzero-sum game in [95], and so on. Recently, a network-based policy gradient RL algorithm (i.e., the proximal policy optimization) was compared with the Lyapunov-based controllers in a traffic PDE system [96], and the RL controller showed compatible performance with the Lyapunov-based controllers, especially in learning (i.e., adaptation) potential under changing and uncertain conditions while the training time was long and convergence of the value function was not guaranteed. Moreover, a value-iteration-based RL method was proposed for sensor placement in the spatial domain of distributed parameter systems in [97]. Most aforementioned methods first lump distributed parameter systems (with infinite-dimensional state spaces) into lumped parameter systems (with finite-dimension state spaces) via various reduced-order models and then perform the RL-based control designs. This implies that these RL-based PDE control designs often depend on specific approximation schemes and lead to approximate control laws [98]. The direct RL-based control design on PDE systems constitutes future work.

D. Fault Detection and Fault-Tolerant Control

This section examines reinforcement learning (RL) algorithms used to address fault detection (FD) and fault-tolerant

control (FTC) problems. The primary objective of FD is to detect faults and determine their source promptly. FTC is designed to enable the system to persistently function in the presence of failures while maintaining a minimum level of performance. In such cases, the system may operate at a reduced capacity compared to a fault-free scenario.

1) *RL-Based Fault Detection*: Research on RL-based FD can be mainly divided into model-based and data-driven approaches [99], [100]:

a) Model-based RL FD methods rely heavily on accurate mathematical modelling knowledge [10]. As part of this approach, diagnostic models are constructed to analyze the system output and compare the state of the machine in terms of health [9], [101], [102]. This approach is derived from the traditional fault diagnosis.

b) The data-driven approach directly applies deep RL techniques for encapsulated processing of the acquired signals to output fault identification results, as shown in references [103], [104]. This “end-to-end” approach directly realizes the output from input to target, promoting the joint optimization of feature extraction and pattern classification parameters in multi-hidden-layer networks. This approach also adopts a strategy of self-learning to automatically discover effective features associated with the target in large datasets.

2) *RL-Based Fault-Tolerant Control*: Fault tolerant control (FTC) has been widely developed to improve the performance of industrial systems in the face of unexpected faults, accidental events, and aging components. RL has emerged as a novel approach for FTC, offering the advantage of requiring fewer design efforts compared to traditional model-based and learning-based methods. By using powerful neural networks and self-adjusted methods, RL-based strategies can effectively address problems such as disturbance elimination, reference tracking, and fault tolerance [105], [106]. In addition, RL provides a feasible method for learning in unknown environments with a large amount of prior data, making it a promising approach for FTC in complex industrial systems. Recent studies have reported successful results in RL-based FTC, including FTC tracking control of MIMO discrete-time systems [107], [108], FTC design for a class of nonlinear MIMO discrete-time systems [109], [110], and RL-based FTC for linear discrete-time dynamic systems [111], [112]. These promising findings suggest that RL-based FTC can significantly benefit industrial systems, and further research in this area should be encouraged. To motivate a novel FTC perspective, the following example illustrates the principle of linear model-based FTC. Taking a linear time-invariant (LTI) system as an example, a fault results in the difference (denoted as $S_F(z)$) between the nominal plant and the faulty counterpart. Then, a separate control structure can be equivalently described where $Q_r(z)$ is a filter to be tuned by using RL. If S_F does not affect the system stability, tuning $Q_r(z)$ is sufficient to tolerate the unpredictable faults.

Besides these advancements and possible improvement areas, it is challenging to implement RL in FTC due to the need for a large amount of training data. This lack of effective training data presents a significant hurdle for RL-based

methods, making it challenging to design real-time agents that can handle unforeseen faults. Another challenge to implementing these algorithms is the high false-positive alarm rates. Overcoming these challenges is critical for successfully implementing RL-based FD and FTC, which can improve system performance and reduce downtime. Therefore, developing novel training acceleration methods to mitigate the impact of errors and improve training efficiency is a worthwhile research direction for RL-based FD and FTC.

E. Optimization

The hierarchical nature of industrial processes involving multiple decision-making levels, such as process design, planning, scheduling, and supply chain, presents challenges in coordinating decisions within each level. Traditionally, these problems were modelled as MILP or MINLP that are NP-hard [113] and are solved using computationally expensive optimization techniques such as Branch and bound and cutting plane methods [114].

RL has recently attracted attention to this domain as its utilization in industrial process optimization offers several advantages. First, industrial processes are inherently complex, making the implementation of model-based optimization challenging [115]. Whereas RL algorithms learn the process behaviour and optimal actions through continuous interaction with the environment, eliminating the need for a process model. Moreover, RL can learn from model mismatches and compensate for them in their actions, leading to improved decision-making [116]. Additionally, in time-critical industrial processes, fast optimization is crucial for timely decision-making [117]. RL offers the advantage of training an optimal policy instead of optimizing actions at every time step. Once the optimal policy is found, an inexpensive forward pass through the function can instantly generate an online solution, leading to faster decision-making. Hierarchical or multi-agent RL has also gained attention for its ability to model complex systems and optimize decisions at different levels of the production process. Applying hierarchical RL to hierarchical and combinatorial optimization problems can simplify the large-scale optimization problem into smaller, more tractable subproblems, leading to improved efficiency and effectiveness [118]. Moreover, the design of RL allows for the incorporation of relevant information related to decision-making that may be hard to model in traditional optimization techniques, providing flexibility in capturing important process dynamics. Process systems engineering (PSE) communities have been contributing to solving some of the decision-making problems. A literature review on a specific application of RL in supply chain management can be found in [7]. Further, Table III lists some significant RL applications in this domain.

In terms of future work, providing any optimality guarantee to RL-based solutions is an open problem. Also, there is an opportunity to combine RL with conventional optimization approaches to leverage the advantages of both techniques.

V. DISCUSSION

Besides these developments in process industries, the RL literature is evolving rapidly in various fields. To accelerate

TABLE III
RL APPLICATIONS IN PROCESS OPTIMIZATION, PLANNING AND SCHEDULING

Application	Environment	Objective	Action	Algorithm
Water treatment [119]	Dynamic model	COD reduction in the effluent	Dilution rate	DDPG
CSTR [120]	CSTR model	Maximize real-time profit while accounting for changing environmental conditions	Reactant flowrate and heat input	A2C
Electrical and heating [116]	Integrated energy system	Optimize operating costs under uncertainties	Heat conversion and power distribution ratio	PPO
Biopharmaceutical chromatography [121]	Simulator and experiment	Maximize product yield and purity	Flowrate	Q learning
Oil sands bitumen extraction [118]	High-fidelity PSV model	HLA: improve bitumen recovery LLA: setpoint tracking, sanding prevention	HLA: controlled variable setpoint LLA: manipulated variables output	Hierarchical A3C
Industrial mixing [122]	Experiment	Minimize mixing variance	Flowrate	DQN
Waterflooding optimization [123]	Matlab Reservoir Simulation Toolbox	Maximize total oil production with minimum cost	Manipulates water injection rates	Q learning
Dynamic sensor planning [124]	Experimental oil & gas testbed	Infer two phase flow rates with minimum cost	Dynamically deploy sensors	Deep Q learning
Chemical production scheduling [125]	Small single stage reactor	Design production schedule for a planning horizon with minimum cost	Dynamically assign production	A2C
Computer aided synthesis planning [126]	Reaxys database	Find synthesis routes with minimum steps and eco-friendly solvents	Retrosynthetic disconnection	Monte-Carlo tree search

this expansion safely and realistically, researchers and practitioners should consider RL's advantages, limitations, and opportunities while examining the trends and new applications. This section discusses these crucial topics to guide the reader.

A. Limitations and Advantages

The concept of autonomy tries to decipher decision-making, especially during adverse situations. The nature of such situations is both sparse and sporadic in nature. Such scenarios are usually based on plants' apparent behaviours unique to each site/unit operation. Such sparse events and adverse interactions have led to potential loss of production and, in the worst cases, fatal injuries. Combined with the nature of the scenarios as well as their sparsity of occurrence, no standard operating procedures can streamline the experience an operator may have in suitable countermeasures developed over time.

While a multitude of models, such as soft sensors, estimators, and forecasting, have been successfully developed and deployed, their usage is limited to normal operations. Fault tolerance or prevention is rarely inherently a part of their design. Such a modal approach to the machine learning models and their poor adoption of sparse scenarios renders them impractical in capturing the operator's instincts. This impracticality is mainly owing to the fact that these models are mostly spatial, limiting them in understanding the long-term implications of a decision. This segues into an opportunity to understand the long-term impact of certain decisions. A time-tested approach for plant optimization is to utilize model predictive controllers to provide appropriate decisions considering the states/measurements from the unit operation. These methods heavily rely on the plant dynamics that are ever-changing and nonlinear in standard templates. The challenges in linearization as well as the heavily invasive approach of plant perturbations, limit this approach to depend on plant inertia. These issues render the inherent characters in the plant, which otherwise are available in the plant data, useless.

RL, as a tool to identify and understand operator instincts, can ensure the preservation of operator instincts, which otherwise would be lost with the personnel over time. The approach involves the utilization of the RL platform to develop a data-driven approach to understanding the long-term implications of operator decisions. The approach can be realized in two steps. The first is an imitation learning-based approach, which aims at exploiting the operator experience using a one-to-one map of decisions from operator data. This approach would allow the RL agent to comprehend the spatial nature of the decisions. During this scenario, if an actor-critic setup is utilized, the value of such decisions can also be empirically assessed, and the critic can be calibrated. The second and crucial step is to deploy the RL agent into an exploration phase. Here, the decisions can be provided to the operators as recommendations. These recommendations can be constrained/flagged/validated by the operator. Based on the number of times the operator accepts the decision, further deliberation of such learning platforms can be commenced. Operator intuition being a crucial part of industrial autonomy, RL provides a robust platform for learning such sparse information.

B. Trends and New Applications

Classical control theory often utilizes an average cost function as shown in (1). On the other hand, most of the RL implementations utilize episodic learning even if they are finally used for continual processes. Since the optimal policies in episodic problems can significantly differ in continual processes, an emerging research topic is *continual learning* [127]. In this setting, the agent/learner utilizes non-discounted variables, similar to the classical-control-theoretic applications. Moreover, *safe learning* techniques showed promising results in the robotics and control fields [84], [128]. These techniques update the actor and critic functions based on the parameter and system constraints. In addition to these advancements, multi-agent methodologies [129] and agents that utilize human feedback to improve their policies [130]

can speed up learning and improve exploration and interpretability of the control system by incorporating human expertise and knowledge. This interaction can also help overcome data availability limitations and uncertainty in the process by incorporating the intuition and experience of human operators. Overall, applying RL in complex process industries can lead to more efficient, reliable, and sustainable processes with improved performance and reduced operational costs.

C. Opportunities and Future Prospects

1) *RL Applications in Educational Settings*: The relative recency of using RL in real-life applications, particularly in process industries/engineering, created a need for qualified engineers equipped with such ML expertise typically not covered by the engineering curriculum. Only computer science departments teach RL and deliver qualified computer scientists to join the demanding market. However, using ML tools in the process industry requires targeted knowledge in specific engineering domains. Hence, engineers and expert-domain matters are more suited to use RL theory and tailor it for every application independently. Therefore, engineering students should be educated and trained with RL to apply it in process industries. However, teaching such specialized and advanced ML tools (RL) in the classroom without pedagogical applications has been considered a challenge, and some experimental applications were proposed in [131].

2) *Fault Detection, Tolerance, and Recovery*: Industrial fault recovery is crucial for the safe and reliable operation of systems. While this survey presents promising RL-based approaches for fault detection and tolerant control, there is a need for resilient agents to recover from unsafe operations. For example, a malfunctioning sensor or actuator in a chemical plant can lead to catastrophic accidents. Designing an RL-based fault recovery agent that adjusts parameters, reroutes flows, or safely shuts down systems can greatly benefit plant operations. By leveraging RL's ability to learn and make intelligent decisions, these agents enhance reliability and safety in Industry 5.0. Future research should focus on effective RL fault recovery strategies, considering interconnected components and potential cascading effects. Integrating RL into fault recovery achieves greater fault tolerance, ensuring continued operation, minimizing unexpected events, enhancing system reliability, and promoting safe human-machine collaboration in Industry 5.0.

3) *Complex Systems and Holistic Designs*: This survey has identified that previous research has predominantly focused on using RL to improve individual levels in the control hierarchy, as shown in Fig. 1. However, this approach fails to consider the interconnectedness of multiple levels in most process industries, which can significantly impact the overall performance of the control system. To address this issue, future research should focus on developing advanced RL algorithms that can optimize the control system design by considering the interactions between multiple levels of the process control hierarchy. Moreover, integrating reinforcement learning into Industry 5.0 holds immense potential for advancing process control and automation. New evaluation metrics and benchmarks should be developed to assess the performance of algo-

rithms in more realistic and complex industrial settings. Pursuing these research directions can improve process control systems' efficiency, reliability, and sustainability and enable the transition toward autonomous and intelligent manufacturing.

4) *Autonomous Control Systems*: Designing sequential interactions in the control hierarchy can be complex and challenging. Due to limited considerations, the traditional approach of using programmable logic controllers (PLCs) may compromise process optimality and safety. To address this, a better alternative is to define higher-level objectives and explore various scenarios comprehensively, enhancing both safety and optimality. Implementing hard constraints requires process knowledge and models while incorporating expert information into system identification methodologies leads to successful applications. Data-driven methods face challenges with extensive training time, but this can be mitigated through offline pre-training, model-based RL agents, long historical data, or sparse agents. Smooth transitions during shutdown and startup phases are crucial for handling nonlinear dynamics. Commissioning an autonomous process automation system is a complex task that requires collaboration across disciplines. RL implementations offer an efficient trial-and-error approach, automating the sequence design process, reducing design times, optimizing control system design, and improving process performance with speed and accuracy. This integration of reinforcement learning, autonomous control systems, and Industry 5.0 presents opportunities for advanced process automation, enabling efficient and adaptive control in complex industrial environments.

5) *Cyber Attacks*: RL can be used to mitigate cyber attacks or enhance cyber security. However, several types of challenges need to be addressed when deploying RL agents. These challenges include the security and privacy challenges of cyber-physical systems, the unpredictability and modelling challenges of human behaviour when mitigating human-related vulnerabilities, the challenges of handling system and performance constraints in the learning process, and improving the learning speed. Moreover, non-stationary environments should also be considered in cyber systems. The success of RL algorithms depends heavily on accurate and consistent feedback from the environment. However, such feedback is challenging to guarantee under various cyber attacks, such as denial-of-service (DoS) attacks, jamming attacks, spoofing attacks, data injection attacks, data poisoning attacks, test-item attacks, etc. To understand RL under cyber attacks, it is necessary to understand the attacking behaviours and how specific attacks influence the learning results of RL. For example, designing learning rules requires further design considerations in the presence of cyber attacks. Otherwise, the learning results (e.g., the policy parameters) could be corrupted due to adversaries causing misleading information (e.g., the reward, measurement, and control signals) since a naive agent might not be able to notice the attacks [132]–[134]. Although several research articles address adversary detection and mitigation through RL solutions [135]–[137],

deploying RL agents in process industries requires further research for robustness and safety [138].

REFERENCES

- [1] C. Liu, J. Ding, and J. Sun, "Reinforcement learning based decision making of operational indices in process industry under changing environment," *IEEE Trans. Ind. Inf.*, vol. 17, no. 4, pp. 2727–2736, Apr. 2021.
- [2] D. P. Bertsekas, *Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control*. Belmont, USA: Athena Scientific, 2022.
- [3] T. Kegyes, Z. Süle, and J. Abonyi, "The applicability of reinforcement learning methods in the development of Industry 4.0 applications," *Complexity*, vol. 2021, p. 7179374, Nov. 2021.
- [4] D. E. Seborg, T. F. Edgar, and D. A. Mellichamp, *Process Dynamics and Control*. 3rd ed. Hoboken, USA: Wiley, 2011.
- [5] H. R. Chi, A. Radwan, N.-F. Huang, and K. F. Tsang, "Guest editorial: Next-generation network automation for industrial internet-of-things in Industry 5.0," *IEEE Trans. Ind. Inf.*, vol. 19, no. 2, pp. 2062–2064, Feb. 2023.
- [6] T. S. Chu, A. B. Culaba, and J. A. C. Jose, "Robotics in the fifth industrial revolution," in *Proc. IEEE 14th Int. Conf. Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management*, Boracay Island, Philippines, 2022, pp. 1–6.
- [7] B. Rolf, I. Jackson, M. Müller, S. Lang, T. Reggelin, and D. Ivanov, "A review on reinforcement learning algorithms and applications in supply chain management," *Int. J. Prod. Res.*, vol. 61, no. 20, pp. 7151–7179, Nov. 2022.
- [8] J. Shin, T. A. Badgwell, K.-H. Liu, and J. H. Lee, "Reinforcement learning—Overview of recent progress and implications for process control," *Comput. Chem. Eng.*, vol. 127, pp. 282–294, Aug. 2019.
- [9] P. Kumar and A. S. Hati, "Review on machine learning algorithm based fault detection in induction motors," *Arch. Comput. Methods Eng.*, vol. 28, no. 3, pp. 1929–1940, May 2021.
- [10] Y. Lei, B. Yang, X. Jiang, F. Jia, N. Li, and A. K. Nandi, "Applications of machine learning to machine fault diagnosis: A review and roadmap," *Mech. Syst. Signal Process.*, vol. 138, p. 106587, Apr. 2020.
- [11] R. Nian, J. Liu, and B. Huang, "A review on reinforcement learning: Introduction and applications in industrial process control," *Comput. Chem. Eng.*, vol. 139, p. 106886, Aug. 2020.
- [12] H. Yoo, H. E. Byun, D. Han, and J. H. Lee, "Reinforcement learning for batch process control: Review and perspectives," *Annu. Rev. Control*, vol. 52, pp. 108–119, Oct. 2021.
- [13] R. R. Negenborn and J. M. Maestre, "Distributed model predictive control: An overview and roadmap of future research opportunities," *IEEE Control Syst. Mag.*, vol. 34, no. 4, pp. 87–97, Aug. 2014.
- [14] M. Ellis, J. Liu, and P. D. Christofides, *Economic Model Predictive Control: Theory, Formulations and Chemical Process Applications*. Cham, Germany: Springer, 2017.
- [15] S. Mata, A. Zubizarreta, and C. Pinto, "Robust tube-based model predictive control for lateral path tracking," *IEEE Trans. Intell. Veh.*, vol. 4, no. 4, pp. 569–577, Dec. 2019.
- [16] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Syst. Mag.*, vol. 36, no. 6, pp. 30–44, Dec. 2016.
- [17] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 3, pp. 269–296, May 2020.
- [18] T. Zhang, S. Li, and Y. Zheng, "Implementable stability guaranteed Lyapunov-based data-driven model predictive control with evolving Gaussian process," *Ind. Eng. Chem. Res.*, vol. 61, no. 39, pp. 14681–14690, Sept. 2022.
- [19] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Data-driven model predictive control with stability and robustness guarantees," *IEEE Trans. Automat. Control*, vol. 66, no. 4, pp. 1702–1717, Apr. 2021.
- [20] P. Kumar, J. B. Rawlings, and S. J. Wright, "Industrial, large-scale model predictive control with structured neural networks," *Comput. Chem. Eng.*, vol. 150, p. 107291, Jul. 2021.
- [21] Y. M. Ren, M. S. Alhajer, J. Luo, S. Chen, F. Abdullah, Z. Wu, and P. D. Christofides, "A tutorial review of neural network modeling approaches for model predictive control," *Comput. Chem. Eng.*, vol. 165, p. 107956, Sept. 2022.
- [22] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. 19th Int. Conf. Neural Information Processing Systems*, Vancouver, Canada, 2006, pp. 153–160.
- [23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, USA: MIT Press, 2016.
- [24] S. Sengupta, S. Basak, P. Saikia, S. Paul, V. Tsalavoutis, F. Atiah, V. Ravi, and A. Peters, "A review of deep learning with special emphasis on architectures, applications and recent trends," *Knowl.-Based Syst.*, vol. 194, p. 105596, Apr. 2020.
- [25] J. Yu and Y. Zhang, "Challenges and opportunities of deep learning-based process fault detection and diagnosis: A review," *Neural Comput. Appl.*, vol. 35, no. 1, pp. 211–252, Jan. 2023.
- [26] L. Biggio and I. Kastanis, "Prognostics and health management of industrial assets: Current progress and road ahead," *Front. Artif. Intell.*, vol. 3, p. 578613, Nov. 2020.
- [27] S. Zhang and T. Qiu, "Semi-supervised LSTM ladder autoencoder for chemical process fault diagnosis and localization," *Chem. Eng. Sci.*, vol. 251, p. 117467, Apr. 2022.
- [28] J. Qian, Z. Song, Y. Yao, Z. Zhu, and X. Zhang, "A review on autoencoder based representation learning for fault detection and diagnosis in industrial processes," *Chem. Intell. Lab. Syst.*, vol. 231, p. 104711, Dec. 2022.
- [29] Q. Sun and Z. Ge, "A survey on deep learning for data-driven soft sensors," *IEEE Trans. Ind. Inf.*, vol. 17, no. 9, pp. 5853–5866, Sept. 2021.
- [30] X. Yuan, B. Huang, Y. Wang, C. Yang, and W. Gui, "Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE," *IEEE Trans. Ind. Inf.*, vol. 14, no. 7, pp. 3235–3243, Jul. 2018.
- [31] X. Yuan, L. Li, Y. A. W. Shardt, Y. Wang, and C. Yang, "Deep learning with spatiotemporal attention-based LSTM for industrial soft sensor model development," *IEEE Trans. Ind. Electron.*, vol. 68, no. 5, pp. 4404–4414, May 2021.
- [32] F. Amjad, S. K. Varanasi, and B. Huang, "Kalman filter-based convolutional neural network for robust tracking of froth-middling interface in a primary separation vessel in presence of occlusions," *IEEE Trans. Instrum. Meas.*, vol. 70, p. 5007308, Feb. 2021.
- [33] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. 2nd ed. Cambridge, USA: MIT Press, 2018.
- [34] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *Proc. 4th Int. Conf. Learning Representations*, San Juan, Puerto Rico, 2016.
- [35] S. D.-C. Shashua and S. Mannor, "Kalman meets Bellman: Improving policy evaluation through value tracking," arXiv preprint arXiv: 2002.07171, 2020.
- [36] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas, "Sample efficient actor-critic with experience replay," in *Proc. 5th Int. Conf. Learning Representations*, Toulon, France, 2017.
- [37] M. Okada and T. Taniguchi, "DreamingV2: Reinforcement learning with discrete world models without reconstruction," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Kyoto, Japan, 2022, pp. 985–991.
- [38] N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano, "Learning to summarize from human feedback," in *Proc. 34th Conf. Neural Information Processing Systems*, Vancouver, Canada, 2020, pp. 3008–3021.
- [39] O. Dogru, K. Velswamy, and B. Huang, "Actor-critic reinforcement learning and application in developing computer-vision-based interface tracking," *Engineering*, vol. 7, no. 9, pp. 1248–1261, Sept. 2021.
- [40] R. Bellman, "The theory of dynamic programming," *Bull. Amer. Math. Soc.*, vol. 60, no. 6, pp. 503–515, Nov. 1954.

- [41] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," arXiv preprint arXiv: 2005.01643, 2020.
- [42] D. P. Bertsekas, *Reinforcement Learning and Optimal Control*. Belmont, USA: Athena Scientific, 2019.
- [43] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, pp. 3–4, May 1992.
- [44] Q. Wei, T. Li, and D. Liu, "Learning control for air conditioning systems via human expressions," *IEEE Trans. Ind. Electron.*, vol. 68, no. 8, pp. 7662–7671, Aug. 2021.
- [45] A. Kekuda, R. Anirudh, and M. Krishnan, "Reinforcement learning based intelligent traffic signal control using n-step SARSA," in *Proc. Int. Conf. Artificial Intelligence and Smart Systems*, Coimbatore, India, 2021, pp. 379–384.
- [46] B. Ning, F. H. T. Lin, and S. Jaimungal, "Double deep Q-learning for optimal execution," *Appl. Math. Finance*, vol. 28, no. 4, pp. 361–380, Oct. 2021.
- [47] P. Casgrain, B. Ning, and S. Jaimungal, "Deep Q-learning for Nash equilibria: Nash-DQN," *Appl. Math. Finance*, vol. 29, no. 1, pp. 62–78, Nov. 2022.
- [48] V. Konda, *Actor-Critic Algorithms*. Cambridge, USA: Massachusetts Institute of Technology, 2000, pp. 1008–1014.
- [49] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. 31st Int. Conf. Machine Learning*, Beijing, China, 2014.
- [50] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1334–1373, Jan. 2016.
- [51] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3–4, pp. 229–256, May 1992.
- [52] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. 12th Int. Conf. Neural Information Processing Systems*, Denver, USA, 1999, pp. 1057–1063.
- [53] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. 27th Int. Conf. Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 2672–2680.
- [54] I. Grondman, L. Busoniu, G. A. D. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Trans. Syst., Man, Cybern., Part C (Appl. Rev.)*, vol. 42, no. 6, pp. 1291–1307, Nov. 2012.
- [55] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. 35th Int. Conf. Machine Learning*, Stockholm, Sweden, 2018.
- [56] V. Mnih, A. P. Badia, M. Mirza, T. Harley, A. Graves, T. Lillicrap, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd Int. Conf. Machine Learning*, New York, USA, 2016, pp. 1928–1937.
- [57] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. 4th Int. Conf. Learning Representations*, San Juan, Puerto Rico, 2016.
- [58] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv preprint arXiv: 1707.06347, 2017.
- [59] R. Grosse and J. Martens, "A kronecker-factored approximate fisher matrix for convolution layers," in *Proc. 33rd Int. Conf. Machine Learning*, New York, USA, 2016, pp. 573–582.
- [60] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," arXiv preprint arXiv: 1812.05905, 2018.
- [61] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. 35th Int. Conf. Machine Learning*, Stockholm, Sweden, 2018.
- [62] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
- [63] P. Langevin, "Sur la theorie du mouvement brownien," *C. R. Acad. Sci.*, vol. 146, pp. 530–533, 1908.
- [64] R. Munos, T. Stepleton, A. Harutyunyan, and M. G. Bellemare, "Safe and efficient off-policy reinforcement learning," in *Proc. 30th Int. Conf. Neural Information Processing Systems*, Barcelona, Spain, 2016, pp. 1054–1062.
- [65] T. Shuprajhaa, S. K. Sujit, and K. Srinivasan, "Reinforcement learning based adaptive PID controller design for control of linear/nonlinear unstable processes," *Appl. Soft Comput.*, vol. 128, p. 109450, Oct. 2022.
- [66] Y. Wu, E. Mansimov, S. Liao, R. Grosse, and J. Ba, "Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation," in *Proc. 31st Int. Conf. Neural Information Processing Systems*, Long Beach, USA, 2017, pp. 5285–5294.
- [67] V. Taboga, A. Bellahsen, and H. Dagdougui, "An enhanced adaptivity of reinforcement learning-based temperature control in buildings using generalized training," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 6, no. 2, pp. 255–266, Apr. 2022.
- [68] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, "Trust region policy optimization," in *Proc. 32nd Int. Conf. Machine Learning*, Lille, France, 2015, pp. 1889–1897.
- [69] G. Campos, N. H. El-Farra, and A. Palazoglu, "Soft actor-critic deep reinforcement learning with hybrid mixed-integer actions for demand responsive scheduling of energy systems," *Ind. Eng. Chem. Res.*, vol. 61, no. 24, pp. 8443–8461, Apr. 2022.
- [70] X. Yuan, Y. Wang, R. Zhang, Q. Gao, Z. Zhou, R. Zhou, and F. Yin, "Reinforcement learning control of hydraulic servo system based on TD3 algorithm," *Machines*, vol. 10, no. 12, p. 1244, Dec. 2022.
- [71] D. Dutta and S. R. Upreti, "A survey and comparative evaluation of actor-critic methods in process control," *Can. J. Chem. Eng.*, vol. 100, no. 9, pp. 2028–2056, Sept. 2022.
- [72] J. Xie, O. Dogru, B. Huang, C. Godwaldt, and B. Willms, "Reinforcement learning for soft sensor design through autonomous cross-domain data selection," *Comput. Chem. Eng.*, vol. 173, p. 108209, May 2023.
- [73] E. Skordilis and R. Moghaddass, "A deep reinforcement learning approach for real-time sensor-driven decision making and predictive analytics," *Comput. Ind. Eng.*, vol. 147, p. 106600, Sept. 2020.
- [74] H. C. Croll, K. Ikuma, S. K. Ong, and S. Sarkar, "Reinforcement learning applied to wastewater treatment process control optimization: Approaches, challenges, and path forward," *Critical Reviews in Environmental Science and Technology*, vol. 53, no. 20, pp. 1775–1794, Mar. 2023.
- [75] K. Lee, D. Isele, E. A. Theodorou, and S. Bae, "Spatiotemporal costmap inference for MPC via deep inverse reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3194–3201, Apr. 2022.
- [76] M. Zanon and S. Gros, "Safe reinforcement learning using robust MPC," *IEEE Trans. Automat. Control*, vol. 66, no. 8, pp. 3638–3652, Aug. 2021.
- [77] O. Dogru, N. Wiecek, K. Velswamy, F. Ibrahim, and B. Huang, "Online reinforcement learning for a continuous space system with experimental validation," *J. Process Control*, vol. 104, pp. 86–100, Aug. 2021.
- [78] B. Zhao, D. Liu, and C. Luo, "Reinforcement learning-based optimal stabilization for unknown nonlinear systems subject to inputs with uncertain constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4330–4340, Oct. 2020.
- [79] Y. Yifei and S. Lakshminarayanan, "Multi-agent reinforcement learning system for multiloop control of chemical processes," in *Proc. IEEE Int. Symp. Advanced Control of Industrial Processes*, Vancouver, Canada, 2022, pp. 48–53.
- [80] L. Chen, F. Meng, and Y. Zhang, "MBRL-MC: An HVAC control approach via combining model-based deep reinforcement learning and model predictive control," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 19160–19173, Oct. 2022.
- [81] H. Dong, X. Zhao, and H. Yang, "Reinforcement learning-based approximate optimal control for attitude reorientation under state constraints," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 4, pp. 1664–1673, Jul. 2021.

- [82] D. G. McClement, N. P. Lawrence, J. U. Backström, P. D. Loewen, M. G. Forbes, and R. B. Gopaluni, "Meta-reinforcement learning for the tuning of PI controllers: An offline approach," *J. Process Control*, vol. 118, pp. 139–152, Oct. 2022.
- [83] I. Carlucho, M. De Paula, and G. G. Acosta, "An adaptive deep reinforcement learning approach for MIMO PID control of mobile robots," *ISA Trans.*, vol. 102, pp. 280–294, Jul. 2020.
- [84] O. Dogru, K. Velswamy, F. Ibrahim, Y. Wu, A. S. Sundaramoorthy, B. Huang, S. Xu, M. Nixon, and N. Bell, "Reinforcement learning approach to autonomous PID tuning," *Comput. Chem. Eng.*, vol. 161, p. 107760, May 2022.
- [85] M. Mehndiratta, E. Camci, and E. Kayacan, "Automated tuning of nonlinear model predictive controller by reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Madrid, Spain, 2018, pp. 3016–3021.
- [86] B. Zarrouki, V. Klöes, N. Heppner, S. Schwan, R. Ritschel, and R. Voßwinkel, "Weights-varying MPC for autonomous vehicle guidance: A deep reinforcement learning approach," in *Proc. European Control Conf.*, Delft, Netherlands, 2021, pp. 119–125.
- [87] E. Bohn, S. Gros, S. Moe, and T. A. Johansen, "Reinforcement learning of the prediction horizon in model predictive control," *IFAC-PapersOnLine*, vol. 54, no. 6, pp. 314–320, Feb. 2021.
- [88] W. H. Ray, *Advanced Process Control*. New York: McGraw-Hill, 1981.
- [89] R. Padhi, S. N. Balakrishnan, and T. Randolph, "Adaptive-critic based optimal neuro control synthesis for distributed parameter systems," *Automatica*, vol. 37, no. 8, pp. 1223–1234, Aug. 2001.
- [90] M. Kumar, K. Rajagopal, S. N. Balakrishnan, and N. T. Nguyen, "Reinforcement learning based controller synthesis for flexible aircraft wings," *IEEE/CAA J. Autom. Sinica*, vol. 1, no. 4, pp. 435–448, Oct. 2014.
- [91] B. Luo, H.-N. Wu, and H.-X. Li, "Data-based suboptimal neuro-control design with reinforcement learning for dissipative spatially distributed processes," *Ind. Eng. Chem. Res.*, vol. 53, no. 19, pp. 8106–8119, Apr. 2014.
- [92] B. Luo and H.-N. Wu, "Approximate optimal control design for nonlinear one-dimensional parabolic PDE systems using empirical eigenfunctions and neural network," *IEEE Trans. Syst., Man, Cybern., Part B (Cybern.)*, vol. 42, no. 6, pp. 1538–1549, Dec. 2012.
- [93] B. Luo, T. Huang, H.-N. Wu, and X. Yang, "Data-driven H_∞ control for nonlinear distributed parameter systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2949–2961, Nov. 2015.
- [94] S. Peitz, J. Stenner, V. Chidananda, O. Wallscheid, S. L. Brunton, and K. Taira, "Distributed control of partial differential equations using convolutional reinforcement learning," arXiv preprint arXiv: 2301.10737, 2023.
- [95] H. Ren, J. Dai, H. Zhang, and K. Zhang, "Off-policy integral reinforcement learning algorithm in dealing with nonzero sum game for nonlinear distributed parameter systems," *Trans. Inst. Meas. Control*, vol. 42, no. 15, pp. 2919–2928, Jul. 2020.
- [96] H. Yu, S. Park, A. Bayen, S. Moura, and M. Krstic, "Reinforcement learning versus PDE backstepping and PI control for congested freeway traffic," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 4, pp. 1595–1611, Jul. 2022.
- [97] Z. Wang, H.-X. Li, and C. Chen, "Reinforcement learning-based optimal sensor placement for spatiotemporal modeling," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2861–2871, Jun. 2020.
- [98] E. N. Evans, M. A. Pereira, G. I. Boutselis, and E. A. Theodorou, "Variational optimization based reinforcement learning for infinite dimensional stochastic systems," in *Proc. 3rd Annu. Conf. Robot Learning*, Osaka, Japan, 2019, pp. 1231–1246.
- [99] S. Fan, X. Zhang, and Z. Song, "Imbalanced sample selection with deep reinforcement learning for fault diagnosis," *IEEE Trans. Ind. Inf.*, vol. 18, no. 4, pp. 2518–2527, Apr. 2022.
- [100] Y. Zhu, X. Liang, T. Wang, J. Xie, and J. Yang, "Multi-information fusion fault diagnosis of bogie bearing under small samples via unsupervised representation alignment deep Q-learning," *IEEE Trans. Instrum. Meas.*, vol. 72, p. 3503315, 2023.
- [101] G. Xu, M. Liu, Z. Jiang, W. Shen, and C. Huang, "Online fault diagnosis method based on transfer convolutional neural networks," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 2, pp. 509–520, Feb. 2020.
- [102] Y. Zhao, T. Li, X. Zhang, and C. Zhang, "Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future," *Renew. Sustain. Energy Rev.*, vol. 109, pp. 85–101, Jul. 2019.
- [103] J. Zhou, L. Zheng, Y. Wang, C. Wang, and R. X. Gao, "Automated model generation for machinery fault diagnosis based on reinforcement learning and neural architecture search," *IEEE Trans. Instrum. Meas.*, vol. 71, p. 3501512, Jan. 2022.
- [104] F. Lv, C. Wen, and M. Liu, "Representation learning based adaptive multimode process monitoring," *Chemom. Intell. Lab. Syst.*, vol. 181, pp. 95–104, Oct. 2018.
- [105] W. Zhao, H. Liu, and F. L. Lewis, "Data-driven fault-tolerant control for attitude synchronization of nonlinear quadrotors," *IEEE Trans. Autom. Control*, vol. 66, no. 11, pp. 5584–5591, Nov. 2021.
- [106] H. Li, Y. Wu, and M. Chen, "Adaptive fault-tolerant tracking control for discrete-time multiagent systems via reinforcement learning algorithm," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1163–1174, Mar. 2020.
- [107] L. Liu, Z. Wang, and H. Zhang, "Data-based adaptive fault estimation and fault-tolerant control for MIMO model-free systems using generalized fuzzy hyperbolic model," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 6, pp. 3191–3205, Dec. 2018.
- [108] K. Zhao and J. Chen, "Adaptive neural quantized control of MIMO nonlinear systems under actuation faults and time-varying output constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3471–3481, Sept. 2020.
- [109] Y. Du, B. Jiang, Y. Ma, and Y. Cheng, "Robust ADP-based slidingmode fault-tolerant control for nonlinear systems with application to spacecraft," *Appl. Sci.*, vol. 12, no. 3, p. 1673, Feb. 2022.
- [110] I. A. Zamfirache, R.-E. Precup, R.-C. Roman, and E. M. Petriu, "Reinforcement learning-based control using Q-learning and gravitational search algorithm with experimental validation on a nonlinear servo system," *Inf. Sci.*, vol. 583, pp. 99–120, Jan. 2022.
- [111] D. P. Zhang and Z. W. Gao, "Reinforcement learning-based fault-tolerant control with application to flux cored wire system," *Meas. Control*, vol. 51, no. 7–8, pp. 349–359, Jul. 2018.
- [112] Q. Chen, Y. Jin, and Y. Song, "Fault-tolerant adaptive tracking control of Euler-Lagrange systems—An echo state network approach driven by reinforcement learning," *Neurocomputing*, vol. 484, pp. 109–116, May 2022.
- [113] N. V. Sahinidis, "Mixed-integer nonlinear programming 2018," *Optim. Eng.*, vol. 20, no. 2, pp. 301–306, Apr. 2019.
- [114] A. P. Barbosa-Póvoa, "Process supply chains management—Where are we? Where to go next?" *Front. Energy Res.*, vol. 2, p. 23, Jun. 2014.
- [115] K. Alhazmi, F. Albalawi, and S. M. Sarathy, "A reinforcement learning-based economic model predictive control framework for autonomous operation of chemical reactors," *Chem. Eng. J.*, vol. 428, p. 130993, Jan. 2022.
- [116] B. Zhang, W. Hu, D. Cao, Q. Huang, Z. Chen, and F. Blaabjerg, "Deep reinforcement learning-based approach for optimizing energy conversion in integrated electrical and heating system with renewable energy," *Energy Convers. Manage.*, vol. 202, p. 112199, Dec. 2019.
- [117] D.-H. Oh, D. Adams, N. D. Vo, D. Q. Gbadago, C.-H. Lee, and M. Oh, "Actor-critic reinforcement learning to estimate the optimal operating conditions of the hydrocracking process," *Comput. Chem. Eng.*, vol. 149, p. 107280, Jun. 2021.
- [118] H. Shafi, K. Velswamy, F. Ibrahim, and B. Huang, "A hierarchical constrained reinforcement learning for optimization of bitumen recovery rate in a primary separation vessel," *Comput. Chem. Eng.*, vol. 140, p. 106939, Sept. 2020.
- [119] T. A. Mendiola-Rodríguez and L. A. Ricardez-Sandoval, "Robust control for anaerobic digestion systems of tequila vinasses under uncertainty: A deep deterministic policy gradient algorithm," *Digit. Chem. Eng.*, vol. 3, p. 100023, Jun. 2022.
- [120] B. K. M. Powell, D. Machalek, and T. Quah, "Real-time optimization using reinforcement learning," *Comput. Chem. Eng.*, vol. 143, p. 107077, Dec. 2020.
- [121] S. Nikita, A. Tiwari, D. Sonawat, H. Kodamana, and A. S. Rathore, "Reinforcement learning based optimization of process chromatography for continuous processing of biopharmaceuticals," *Chem. Eng.*

- Sci.*, vol. 230, p. 116171, Feb. 2021.
- [122] M. Konishi, M. Inubushi, and S. Goto, "Fluid mixing optimization with reinforcement learning," *Sci. Rep.*, vol. 12, no. 1, p. 14268, Aug. 2022.
- [123] F. Hourfar, H. J. Bidgoly, B. Moshiri, K. Salahshoor, and A. Elkamel, "A reinforcement learning approach for waterflooding optimization in petroleum reservoirs," *Eng. Appl. Artif. Intell.*, vol. 77, pp. 98–116, Jan. 2019.
- [124] A. Tewari, K.-H. Liu, and D. Papageorgiou, "Information-theoretic sensor planning for large-scale production surveillance via deep reinforcement learning," *Comput. Chem. Eng.*, vol. 141, p. 106988, Oct. 2020.
- [125] C. D. Hubbs, C. Li, N. V. Sahinidis, I. E. Grossmann, and J. M. Wassick, "A deep reinforcement learning approach for chemical production scheduling," *Comput. Chem. Eng.*, vol. 141, p. 106982, Oct. 2020.
- [126] X. Wang, Y. Qian, H. Gao, C. W. Coley, Y. Mo, R. Barzilay, and K. F. Jensen, "Towards efficient discovery of green synthetic pathways with Monte Carlo tree search and reinforcement learning," *Chem. Sci.*, vol. 11, no. 40, pp. 10959–10972, Sept. 2020.
- [127] K. Khetarpal, M. Riemer, I. Rish, and D. Precup, "Towards continual reinforcement learning: A review and perspectives," *J. Artif. Intell. Res.*, vol. 75, pp. 1401–1476, Dec. 2022.
- [128] Z. Yuan, A. W. Hall, S. Zhou, L. Brunke, M. Greeff, J. Panerati, and A. P. Schoellig, "Safe-control-gym: A unified benchmark suite for safe learning-based control and reinforcement learning in robotics," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 11142–11149, Oct. 2022.
- [129] Y. Jiang, W. Gao, J. Wu, T. Chai, and F. L. Lewis, "Reinforcement learning and cooperative H_∞ output regulation of linear continuous-time multi-agent systems," *Automatica*, vol. 148, p. 110768, Feb. 2023.
- [130] F.-Y. Wang, Q. Miao, X. Li, X. Wang, and Y. Lin, "What does ChatGPT say: The DAO from algorithmic intelligence to linguistic intelligence," *IEEE/CAA J. Autom. Sinica*, vol. 10, no. 3, pp. 575–579, Mar. 2023.
- [131] M. Tokic and H. B. Ammar, "Teaching reinforcement learning using a physical robot," in *Proc. Workshop on Teaching Machine Learning at the 29th Int. Conf. Machine Learning*, 2012.
- [132] H. Song, D. Ding, H. Dong, and X. Yi, "Distributed filtering based on Cauchy-kernel-based maximum correntropy subject to randomly occurring cyber-attacks," *Automatica*, vol. 135, p. 110004, Jan. 2022.
- [133] Y. Huang, L. Huang, and Q. Zhu, "Reinforcement learning for feedback-enabled cyber resilience," *Annu. Rev. Control*, vol. 53, pp. 273–295, Jul. 2022.
- [134] M. Xie, D. Ding, X. Ge, Q.-L. Han, H. Dong, and Y. Song, "Distributed platooning control of automated vehicles subject to replay attacks based on proportional integral observers," *IEEE/CAA J. Autom. Sinica*, 2022. DOI: 10.1109/JAS.2022.105941
- [135] F. Wei, Z. Wan, and H. He, "Cyber-attack recovery strategy for smart grid based on deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 3, pp. 2476–2486, May 2020.
- [136] M. N. Kurt, O. Ogundijo, C. Li, and X. Wang, "Online cyber-attack detection in smart grid: A reinforcement learning approach," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5174–5185, Sept. 2019.
- [137] I. Ortega-Fernandez and F. Liberati, "A review of denial of service attack and mitigation in the smart grid using reinforcement learning," *Energies*, vol. 16, no. 2, p. 635, Jan. 2023.
- [138] S. Parker, Z. Wu, and P. D. Christofides, "Cybersecurity in process control, operations, and supply chain," *Comput. Chem. Eng.*, vol. 171, p. 108169, Mar. 2023.

Oguzhan Dogru Bio of Oguzhan Dogru can be found at <https://ieeexplore.ieee.org/author/37089293475>.

Junyao Xie (Member, IEEE) Bio of Junyao Xie can be found at <https://ieeexplore.ieee.org/author/37089777001>.

Om Prakash His research interests include machine learning and its application to process systems engineering.

Ranjith Chiplunkar Bio of Ranjith Chiplunkar can be found at <https://ieeexplore.ieee.org/author/37270341700>.

Jansen Soesanto His research interests involve process modeling, control, and optimization using first principles, machine learning, etc.

Hongtian Chen (Member, IEEE) Bio of Hongtian Chen can be found at <https://ieeexplore.ieee.org/author/37086392741>.

Kirubakaran Velswamy Bio of Kirubakaran Velswamy can be found at <https://ieeexplore.ieee.org/author/37089514521>.

Fadi Ibrahim Bio of Fadi Ibrahim can be found at <https://ieeexplore.ieee.org/author/37580427800>.

Biao Huang (Fellow, IEEE) Bio of Biao Huang can be found at <https://ieeexplore.ieee.org/author/37270341700>.