Deng, Jifei; Sierla, Seppo; Sun, Jie; Vyatkin, Valeriy

Reinforcement learning for industrial process control

Published in:
Computers in Industry

DOI:
10.1016/j.compind.2022.103748

Published: 01/12/2022

Document Version
Publisher's PDF, also known as Version of record

Computers in Industry 143 (2022) 103748

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.sciencedirect.com/journal/computers-in-industry

Reinforcement learning for industrial process control: A case study in flatness control in steel industry

Jifei Deng [a,*], Seppo Sierla [a], Jie Sun [b,*], Valeriy Vyatkin [a,c]

[a] Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, Espoo, Finland
[b] State Key Laboratory of Rolling and Automation, Northeastern University, Shenyang, China
[c] Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå, Sweden

ARTICLE INFO

Keywords:
Strip rolling
Process control
Reinforcement learning
Ensemble learning

ABSTRACT

Strip rolling is a typical manufacturing process, in which conventional control approaches are widely applied. Development of the control algorithms requires a mathematical expression of the process by means of the first principles or empirical models. However, it is difficult to upgrade the conventional control approaches in response to the ever-changing requirements and environmental conditions because domain knowledge of control engineering, mechanical engineering, and material science is required. Reinforcement learning is a machine learning method that can make the agent learn from interacting with the environment, thus avoiding the need for the above mentioned mathematical expression. This paper proposes a novel approach that combines ensemble learning with reinforcement learning methods for strip rolling control. Based on the proximal policy optimization (PPO), a multi-actor PPO is proposed. Each randomly initialized actor interacts with the environment in parallel, but only the experience from the actor that obtains the highest reward is used for updating the actors. Simulation results show that the proposed method outperforms the conventional control methods and the state-of-the-art reinforcement learning methods in terms of process capability and smoothness.

1. Introduction

Strip rolling is a manufacturing process in which metal strips are passed through rolling mills to reduce their thickness, improve flatness and make the thickness uniform (Ginzburg, 2009). Steel strips are an important raw material in modern industries, such as shipbuilding, automotive, and construction (Deng et al., 2019). In the strip rolling industry, production lines are subject to the influence of various difficult-to-predict factors, resulting in randomly appearing quality issues, even when the production process works stably. These factors are referred to as common cause variation, which is considered to be an inherent part of the production process and cannot be changed without changing the process itself (Qiu, 2013). For example, in the strip rolling field, the vibration of the running mills and the changing temperature of the roller and product surface cannot be fully predicted or avoided by redesigning the production line or control rules, but these factors will influence the quality of products (Ginzburg, 2009). Researchers studied finite element models and pure mathematical analysis (Jin et al., 2020; Wang et al., 2022a, 2022b; Mathieu et al., 2017) to redesign the mill or control models, but these time-consuming approaches could not solve the above problems completely.

Proportional integral (PI) controller is widely applied in control systems of production lines in the current strip rolling factories. It is synthesized based on the first principles and empirical models. For example, in a cold rolling production line, a feedback control loop that receives measured flatness values of the strips and outputs process parameters for mills is used to control flatness quality. When designing control logic, specific mathematical models of the studied problems, which have been proved to be correct in the field, can be predetermined and instantly executed online. However, for common cause variation, control experts need to design specific rules by manually considering potential problems. The trial-and-error approach is widely used in real factories, which is time-consuming and inefficient (Deng et al., 2019). Therefore, new methods that could cover the basic functions of the existing controller, while coping with the common cause variation, would be important for quality improvement.

The new trend of data-driven controllers could be of help. Such controllers are built using machine learning methods by data mining the available process data. The data collected from a stable process in a real production line contains the information of manufacturing principles

* Corresponding authors.
E-mail addresses: jifei.deng@aalto.fi (J. Deng), sunjie@ral.neu.edu.cn (J. Sun).

https://doi.org/10.1016/j.compind.2022.103748
Received 8 March 2022; Received in revised form 17 June 2022; Accepted 13 July 2022
Available online 2 August 2022

and common cause variation. The controller is designed through offline data analysis and deployed for online use. This offline-training-online-practice (OTOP) mode offers more adaptability and accuracy of control, and has been applied in various fields (Nawfel et al., 2021; Zhang et al., 2021a, 2021b; Wang et al., 2022a, 2022b).

In this paper, the OTOP mode is incorporated into model-free reinforcement learning (RL). Recent studies show that RL can cope with high-dimensional stochastic problems (Vanvuchelen et al., 2020; Zhang et al., 2021a, 2021b; He et al., 2021). During the training process, the agent is not told what to do, but instead must discover which actions yield the most reward by trying them (Sutton and Barto, 2018). For model-free RL, on-policy methods attempt to evaluate or improve the policy that is used to make decisions, while off-policy methods evaluate or improve a policy different from that used to generate the data. State-of-the-art on-policy RL methods include Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) and Proximal Policy Optimization (PPO) (Schulman et al., 2017). The off-policy methods include Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2016), Twin Delayed DDPG (TD3) (Fujimoto et al., 2018), and Soft Actor-Critic (SAC) (Haarnoja et al., 2018). In control fields, RL based methods have already been applied. Chen et al. (2021) proposed a model-free control scheme for emergency control in a power system based on multi-Q-learning based emergency plans and DDPG. Liu et al. (2015) presented an RL algorithm with fewer learning parameters for discrete-time multiple-input-multiple-output (MIMO) systems. Based on DDPG and a deep Q-network, a novel autonomous control framework is adopted to autonomous voltage control, supporting grid operators in making effective and timely control actions (Duan et al., 2020). In intelligent transportation systems, an intent-based traffic control system is proposed by investigating RL for a 5G-envisioned Internet of Connected Vehicles, which can dynamically orchestrate edge computing and content caching to improve the profits of the Mobile Network Operator (Ning et al., 2021).

RL based methods have also been adopted in the steel-making industry. In basic oxygen steel making, RL is used to express model behavior and increase transparency to generate short polynomials that can explain the model of the process from what it has learnt from process data (Ståhl et al., 2021). Guo et al. (2020) proposed an ε-greedy RL method for the hybrid flow-shop scheduling problem of a steel production process, outperforming the state-of-the-art genetic algorithm. A data-driven RL method is developed for model parameters identification and roll gap control of a bar (Gamal et al., 2021). Han et al. (2020) proposed an RL approach with Monte-Carlo search and policy gradient to train a hierarchical granular computing-based model to construct long-term prediction intervals for reducing byproduct gas from the iron-steel-making process. However, using RL methods to train a controller for a process control system of a strip rolling line is a new topic.

In this paper, to improve the quality of the steel strips, a new controller which is a learnt policy of RL is developed. Without upgrading the machines and modifying the control logic, the controller which sets the process parameters of the final mill is studied. Our focus is limited to the final mill, because real-time flatness control is determined by the final mill. Currently, in the control system at our case study factory, a general PI controller is used. However, such a controller requires control experts to tune the parameters (e.g., by trial-and-error), and the controller is not adaptive to the rolling process, because it cannot learn from the process. Therefore, RL which is easy to implement and learn from the data is adapted to train a new controller. The real rolling data collected from the control system were analyzed offline using RL methods. This paper proposed a novel method that is based on multi-actor PPO. Because safety is crucial in the strip rolling industry, stable performance is required for the new method to reduce risks. Based on the random initialization of actors, given the same states, actors will output different actions with different new states and rewards. For the proposed method, only the actions with the highest rewards will be stored to update the actors, therefore, the pressure of local optimum can be relieved. Using the new method, the process is expected to produce strips of higher quality.

The developed RL based control method will be compared to an industry state-of-the-art PI controller. In the simulation, time-series data will be collected separately from the process using the PI and our proposed RL controller. The flatness values will be compared and analyzed, strips with lower flatness at each point are considered to have higher quality. The main intended contribution of this paper is to develop a novel RL based controller for a real industrial control system substantially outperforming the existing methods. The approach aims at the following results:

1. To propose an RL based intelligent controller for a control system to replace the existing controller.
2. To propose a novel method for training the RL controller, by combining PPO with ensemble learning that can reduce the influence of falling into the local optimum.
3. To confirm the superiority of the proposed method in simulation, using data from a real steel plant. Metrics of process capability and smoothness of controlled flatness values are used for the performance comparison of the RL and PI controllers.

The rest of the paper is organized as follows. Conventional methods and recent approaches to RL in control are reviewed in Section 2. Section 3 introduces the studied strip rolling line and system model. Section 4 illustrates the proposed method. Results are described and analyzed in Section 5. Section 6 discusses the limitations of the current work, the corresponding solutions, and future plans. Section 7 concludes this paper and describes the next step.

2. Literature review

The conventional Proportional (P) or Proportional Integral (PI) controllers have been used in process industries for more than two decades. These controllers require a mathematical expression of the process to be controlled, and their application involves controller tuning (Nian et al., 2020). However, designing a mathematical model (first principles or empirical) and deriving the control law require extensive knowledge from an expert with relevant domain knowledge (Spielberg et al., 2017). Thus, the application of such controllers on complex systems could be computationally demanding, and maintenance is difficult (Shin et al., 2019). Moreover, conventional approaches are not adaptive in nature, because these controllers can only obey certain rules preset by a control expert instead of taking intelligent decisions based on the real changeable condition.

On the contrary, the self-learning controller learns to control a process just by interacting with the process using an intelligence algorithm. The learning process enables controllers to understand the plant dynamics and then to act optimally to control actuators. It does not require the mathematical model of the process, does not involve controller tuning and it is fast since it does not have the optimization step (Bao et al., 2021).

The goal of the RL is to make the controller learn the optimal mapping of situations to actions through interaction guided by a scalar reward signal (Sutton and Barto, 2018). Moriyama et al. (2018) applied

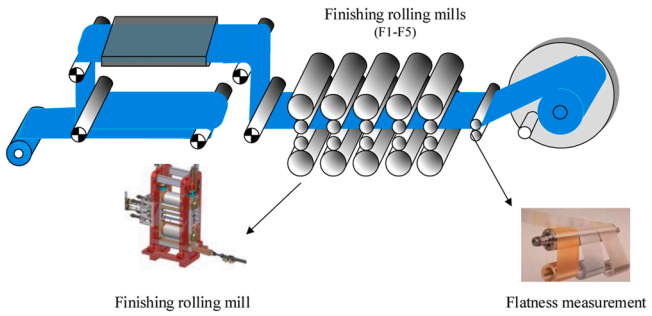

Fig. 1. Example of a cold rolling line.

an RL algorithm to a data-center cooling application where accurate system models are very difficult to identify even with sufficient data. Fan et al. (2019) achieved increased energy efficiency using deep RL by allowing the agent to learn the optimal policy online instead of mathematically modelling the system. Combining the intelligent data-based models of random forest and a human knowledge-based multi-criteria structure of the analytical hierarchical process, a deep Q-Network-based decision support system was proposed to optimize the subjective factors of the textile manufacturing process (He et al., 2021). Based on the actor-critic algorithm, Liu et al. (2021) proposed a multi-actor network ensemble is proposed for decision-making in a mineral processing plant.

Before starting a real production line, experts and engineers will perform parameter tuning. The trial-and-error method is used, but it does not mean the engineers will groundlessly set parameters. Based on the engineers' experience, knowledge, and availability of the production line, they will first give regular values (within an acceptable region, and not far away from the optimal values). Then trial-and-error is adopted to adjust the parameters. This process is time-consuming, risky, and requires the collaboration of experts in various domains. Moreover, steel strips will be wasted because machines need to produce the strips to evaluate the parameters. However, this method has advantages, since safety can be guaranteed because the experts have full knowledge of the production, which can effectively address potential problems. And this proven technique has complete plans for system updates and problem solving.

Our motivation for applying RL to strip rolling control is threefold. Firstly, strip rolling is a classic example of a manufacturing process, where rolling mills are distributed across the production line. Each mill has multiple actuators that control the rolling force, bending force, and roll shifting (Deng et al., 2019). Feedback control with a PI controller is the current state-of-the-art in such systems, and the RL policy has the ability of working as a controller. Secondly, according to various specifications and steel grades of the strips, controllers require different setpoints. The predesigned controller is not adaptive to variations and changing rolling conditions, resulting in irregular strip quality. For example, data collected from a real rolling production line contains significant noise, making information extraction with common machine learning methods difficult, so specific methods were required to process the data before modeling (Sun et al., 2021). However, an RL policy which learns from samples collected from interactions is able to solve these problems. Thirdly, unlike conventional methods that rely on complex mathematical models, RL has a simpler structure that is easier to implement.

In this paper, based on a state-of-the-art RL method (PPO), ensemble learning is adapted to develop a novel method (EPPO) for the strip rolling control system. An industrial problem (flatness control) is studied, and the proposed method was compared to the existing PI control method at the case study steel mill.

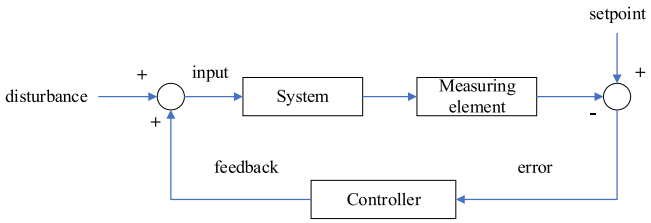

Fig. 2. Feedback control.

3. Strip rolling control

3.1. Background

Fig. 1 shows a cold rolling line with five mills. From left to right, the strip is rolled by the mills in sequence. A mill is propelled by motors and cylinders. In practice, the system controls the flatness through two steps. In step 1, system computes the process parameters based on the measured flatness. Based on the given process parameters, in step 2, system controls multiple actuators (cylinders). In this paper, we studied the step 1 to build a model between the flatness and process parameters. After the final mill, sensors measure the flatness of the strip every 20 ms and send the data to the control system. Considering the space between the mills and costs, only the final flatness is available for flatness control. As shown in Fig. 2, a feedback controller controls the final mill. After receiving the flatness measurement from the sensors, the controller computes and sends new process parameters for the mill. The flatness index (Paakkari, 1998), for an individual strip is defined as:

$$I = \left(\Delta L / L_{\text{ref}}\right) \times 10^5 \tag{1}$$

where L_{ref} is the length of the strip as a reference and ΔL is the difference between the length of a given strip and the reference strip.

In industrial practice, the strip rolling controller is based on pure mathematical models, considering the relevant physical principle of flatness and process parameters. The PI controller designed by experts is not adaptive, because the performance is determined by the experts' experience and understanding of strip rolling. Upgrading the control system requires new first-principles or empirical models, which is inefficient. Therefore, developing a fast and effective method for replacing or improving the existing controllers is a demanding task.

3.2. Problem formulation

As shown in Fig. 3, the strip is moving from left to right. The flatness measurement is pegged on the production line and measures the flatness of 4 reference points at each time. The flatness of the strip at time t can be defined as$S_t = (s_t^0, s_t^1, s_t^2, s_t^3)$. Process parameters of the mill include rolling force, bending force of the work roll and intermediate roll, and the roll gap tilting, which are defined as$U_t = (u_t^0, u_t^1, u_t^2, u_t^3)$. During the rolling process, a controller receives S_t, then computes the output U_t. This process takes place every 20 ms to control flatness.

Since RL has already been proved to perform well on some control

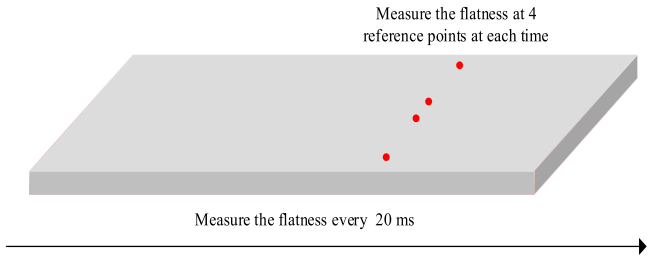

Fig. 3. Example of measuring flatness.

problems (Vanvuchelen et al., 2020; He et al., 2021), this paper applies RL to strip rolling control problems. Using model-free RL methods, an agent will be trained through interaction. The obtained policy is proposed to replace the existing PI controller in the industrial strip rolling control system.

The data description of the studied production line is shown in Table 1. The inputs to the RL controller are the states S_t (flatness of the strip at four reference points). The outputs are actions U_t (process parameters of the mill). The control target is to compute the optimal control actions for the final mill, producing high quality strips for which the flatness values are close to zero. The flatness determines the quality of the strips. Lower flatness is a sign of higher quality. Therefore, the reward function is shown as follows:

$$r_t = -\left(|s_t^0| + |s_t^1| + |s_t^2| + |s_t^3|\right)/4. \tag{2}$$

Designing this reward function has three main reasons. Firstly, in a factory, to evaluate the flatness of a strip, flatness values at different reference points at each timestep are averaged, and this averaged value is used to represent the flatness at the current timestep. Secondly, in practice, even at different reference points, the flatness values have a similar trend without trip points. This situation is determined by the method of adjusting the flatness (Bemporad et al., 2010). Thirdly, for the customer's demand, two-sigma rule is used, meaning that a strip is qualified if 95 % of the averaged values are within a desired boundary. During the training process, this reward function can guide the policy to output actions that can generate a higher reward. In other words, the reward will increase if the policy is correctly trained. Finally, when it converges to a stable value, only the actor will be saved for testing.

4. Proposed method

4.1. Proximal policy optimization

In this paper, PPO was adopted as the base algorithm, which was proven to be effective and efficient for many application fields (Vanvuchelen et al., 2020). PPO is an approximate version of TRPO that relies only on a first-order gradient, which is significantly simpler to implement. It relies on specialized clipping in the objective function to remove incentives for the new policy to get far from the old policy (Schulman et al., 2017). Using an actor-critic framework, PPO has a policy and a value function. The policy network is trained with the clipped surrogate objective:

$$L^{clip}(\theta) = \hat{\mathbb{E}}_t\left[\min\left(\frac{\pi_\theta(u|s)}{\pi_{\theta_{old}}(u|s)}A_t, \mathrm{clip}\left(\frac{\pi_\theta(u|s)}{\pi_{\theta_{old}}(u|s)}, 1-\varepsilon, 1+\varepsilon\right)A_t\right)\right] \tag{3}$$

where π_θ and $\pi_{\theta_{old}}$ are a new and old policy parameterized by θ and θ_{old}, u and s are the action and state and ε is a hyperparameter, $\varepsilon = 0.2$. Given state as input, action is the output of actor. A is an estimator of the advantage function which is computed with generalized advantage estimation (GAE) (Schulman et al., 2016).

Considering an entropy bonus $\widehat{S}[\pi_\theta]$, two terms can be combined to the following objective:

$$L^{clip+S}(\theta) = \hat{\mathbb{E}}_t[L^{clip}(\theta) + cS[\pi_\theta]] \tag{4}$$

where c is the hyperparameter for entropy, and $c = 0.01$ in this paper. To train the value function network, mean squared error of the value and return is computed, the objective is given by:

$$L^{value}(\phi) = \hat{\mathbb{E}}_t\left[\left(V_\phi(s_t) - R_t\right)^2/2\right] \tag{5}$$

where V_ϕ is the state-value function parameterized by ϕ and R is the return. Using state as input, the state-value is the output of the critic. During the training process, gradient descent will be used to update the actor and critic.

Table 1
Data description of states and actions.

	Parameters	Values
Actions	Rolling force	7056–7448 kN
	Bending force of work roll	65–104 kN
	Bending force of intermediate roll	77–180 kN
	Roll gap tilting	-0.01 to 0.18 mm
States	Flatness 1	-4.5 to 21 I
	Flatness 2	-10 to 15 I
	Flatness 3	-10 to 7.5 I
	Flatness 4	-20 to 0.5 I

4.2. Proximal policy optimization with multiple actors

In this paper, ensemble learning is combined with PPO by designing multiple actors. The original PPO has an actor and a critic. Ensemble PPO with N actors (EPPO) defined as $\{\pi_{\theta_1}, \pi_{\theta_2}, \ldots, \pi_{\theta_N}\}$ an output $(u^1, u^2, \ldots, u^N)$. The networks are updated in parallel, and their initializations are set randomly. For each timestep, based on reward function Eq. (2), reward of each state-action pair is computed separately. The overall optimal action is selected based on the following equation:

$$u_t^* = \underset{u_t \in \{u^1, u^2, \ldots, u^N\}}{\arg\max}\ r(s_t, u_t) \tag{6}$$

The optimal action is the action that has the highest reward.

Algorithm 1

Initialize actor networks π_{θ_i}, critic network π_ϕ with random parameters θ_i, ϕ

Initialize replay buffer B

For $t = 1$ **to** T **do**

  For $i = 1$ **to** N_{actor} **do**

    Run policy π_{θ_i} observe reward r_i^t and next state s_i^{t+1}

    Compute value function V_t

  End for

  $o = \arg\max_{i\in\{1,2,\ldots,N_{actor}\}} r_i^t$

  Define the policy π_{θ_o} with the highest reward as the best policy at each timestep

  Store transition $\left(s_o^t, u_o^t, r_o^t, s_o^{t+1}\right)$ in B

  Compute advantage estimates $\hat{A}^t$

  For $ep = 1$ **to** N_{epoch} **do**

    Sample mini-batch of N_{batch} transitions (s, u, r, s') from B

    For $i = 1$ **to** N_{actor} **do**

      Update actor θ_i by gradient method w.r.t. $L^{clip+S}(\theta_i)$

    End for

    Update critic π_ϕ by gradient method w.r.t. $L^{value}(\phi)$

  End for

End for

The pseudocode of the proposed method (EPPO) is shown in Algorithm 1. The methods will be run for T timesteps. In the beginning, given an initial state, N_{actor} randomly initialized actors will compute and output different actions based on the state. The environment will output the rewards and new states after receiving the actions. In the original PPO, the state, action, and reward for each timestep will be stored in the experience replay memory and will be used to update the actor every m timesteps. The samples will not be reused. For EPPO, because of multiple actors, each timestep will generate N_{actor} action-reward-new state tuples, but only the tuple with the highest reward will be stored in the experience replay memory and used to update all the actors. In other words, during the training process, samples with the most contribution are used to update the actors. For each update, the actors and critic will be updated for N_{epoch} times based on the same data sampled from buffer B. Moreover, considering the safety, a constraint avoids dangerous actions from the actors. Based on the machine capabilities, actions are mathematically restricted in an acceptable region. EPPO is intended to overcome the weakness of the original PPO method, in which the random initialization of the actors and the dynamics of the environment can cause the policy to easily fall into different local optima, making the performance unstable when repeating the training with random seeds.

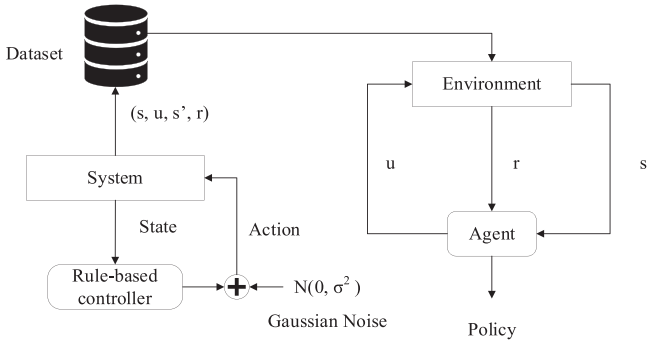

Fig. 4. Data collection, system modelling, and policy training.

4.3. Environment modelling

For RL methods in our paper, an environment is required. As shown in Fig. 4, the environment is used to interact with an agent for training a policy. Depending on the RL application, the environment can be a mathematical model, a real control system, a black box model (e.g., neural networks), etc. In other words, the environment is a model of the studied problem.

For research targets like robots or video games, the agent can interact with the real environment. For example, with the famous AlphaGo, the researchers make the agent play GO games tens of thousands of times to gain experience. During this process, the agent could lose or win the games, but it is not risky or costly, since this process is fully happening on a computer.

However, for industrial cases, such as strip rolling control in our paper, it is impossible to make the agent interact with the real system. Because the system needs to take actions given by the agent, and the

agent will try to explore the action space during the training process. In other words, the agent could give random and unsafe actions. Although we can try to constrain actions to a reliable region, this process is extremely risky in a safety critical industrial environment. For this reason, we first build an environment manually.

The real system has mathematical models of solid mechanics, mechanical engineering, and abrasion, so it is hard to build a physics-based model. Moreover, some of the models are sensitive in factories. Therefore, as shown in Fig. 4, using MATLAB system identification toolbox, we designed the environment by building a black box model (state-space model) based on the real data collected from the factory. Then, either the training process or the evaluation of the RL agent are based on this approximated model.

For the black box model mentioned above, we directly adopted a state-space model in MATLAB. Based on its general framework, the real data were used to compute parameters of the state-space model. Our dataset from the strip rolling line of the case study factory is time-domain data with 4 inputs and 4 outputs. In our paper, a third-order state-space model was estimated. Our RL algorithms are based on the PyTorch platform in Python, so we ported that state-space model developed in MATLAB to Python by recording the parameters of the state-space model and building a new state-space model in Python using the recorded parameters. After that, based on the Python version state-space model, an RL environment was developed, including a step function, reward function, and reset function. All the RL methods in our paper were trained and evaluated on this environment.

4.4. Experiment settings

In this paper, the proposed method is evaluated with a strip rolling problem. The studied strip rolling production line has five rolling mills and sensors, and each mill has four process parameters to control the flatness of strips. Only the normal condition of the process is studied. Our RL policy works as a feedback controller for the final mill to keep the flatness close to the target value. The work has two steps. First, based on the data from the case study factory, offline training was carried out in the lab using high performance computers. Second, the policy was evaluated after being trained. For this evaluation step, currently it was done in the lab, but online evaluation is possible. The trained policy is a neural network, it can be added to PLCs.

Considering the safety and costs, it is impossible to directly make any experiments on the real production line, so this paper introduced an offline analysis. Firstly, based on the strip rolling related knowledge and real data collected from the production line, experts from the factory designed a simulator, which is also called environment for RL in this paper. Secondly, the new controller was trained based on this environment using the EPPO algorithm. Finally, the new controller was evaluated using this environment.

The environment is a model of the real system by simplifying the connection between the machines and products. In practice, the real system has pure mathematical models of abrasion, pressure, etc. However, these models are sensitive and complex, which cannot be directly used for the experiments in this paper. Using data-driven principles and rolling knowledge, partners from the factory designed the environment.

Five methods are evaluated and compared in this paper, including DDPG, SAC, TD3, PPO, and EPPO. The hyperparameters were selected based on the recommended values in the original papers (Lillicrap et al., 2016; Schulman et al., 2017; Fujimoto et al., 2018; Haarnoja et al., 2018), and grid research was adopted for key hyperparameters (e.g., act noise). The hyperparameters of the models are shown in Table 2. In order to objectively compare and analyze the methods, the shared hyperparameters such as total steps and optimizer will have the same values. Each method will be repeated five times with random seeds.

The proposed method (EPPO) is analyzed in simulation by adding Gaussian noise to model the disturbance. The data of an existing strip rolling process with PI control were collected from a real production line and our EPPO based control was implemented. For the proposed and existing methods, given initial states, the policy or controller computes and output actions. Based on the actions, new states will be generated. The time-series states were collected to analyze quality performance. For the studied strip rolling problem, the optimal state values are 0, which corresponds to ideal flatness. In other words, the method that generates new states to be closer to 0 is better.

4.5. Quality metrics

For the studied strip rolling problem, the purpose is to control the flatness of the strips, keeping the values within a boundary designed by engineers, and close to the optimal value of 0. In this paper, capability and smoothness will be studied to evaluate the performance of the proposed method.

Capability of a process is the ability to produce output within specification limits. Capability is measured with capability ratios such as C_p and C_{pk} (Qiu, 2013). C_p and C_{pk}measure how consistent the data are around the average performance. C_p indicates whether the process can produce products to specifications.C_{pk}indicates whether the process is capable of producing within specifications and it is also an indicator of the ability of the process to adhere to the target specification. The equations of C_p and C_{pk} are given by Qiu (2013):

$$C_p = \left(USL - LSL\right)/6\sigma \tag{7}$$

$$C_{pk} = \min[(USL-\mu)/3\sigma, (\mu - LSL)/3\sigma] \tag{8}$$

where USL and LSL are the upper and lower specification limits of the process, μ and σ are sample mean and standard deviation.

The coefficient of correlation between two values in a time series is called the autocorrelation function (ACF) (Box et al., 2015; Liu et al., 2019). Instead of correlation between two different variables, ACF

Table 2
Hyperparameters of RL methods.

Shared		**TD3**		**SAC**	
Hyperparameters	Value	Hyperparameters	Value	Hyperparameters	Value
Total steps	270,000	Start step	10,000	Start step	10,000
Optimizer	Adam	Update after	1000	Update after	1000
Learning rate for actor	1e-3	Act noise	0.1	Entropy coefficient	0.2
Learning rate for critic	1e-3	Target noise	0.2	Polyak	0.95
Activation function	ReLU	Noise clip	0.5		
Update every	300	Policy delay	2	**DDPG**	
Batch size	100	Polyak	0.95	Hyperparameters	Value
gamma	0.99			Start step	10,000
Hidden nodes of nets	64	**PPO/EPPO**		Update after	1000
		Hyperparameters	Value	Act noise	0.1
		Clip range (ε)	0.2	Polyak	0.995
		Lambda	0.98		
		EPPO actors	3		

measures the self-similarity of the variables. The ACF for a time series (the length is N) is given by Box et al. (2015):

$$corr_k = \frac{\sum_{t=k+1}^{N}(y_t-\mu)(y_{t-k}-\mu)}{\sum_{t=1}^{N}(y_t-\mu)^2} \tag{9}$$

where k is the time gap being considered and is called the lag. A lag 1 ACF ($k = 1$ in the above) is the correlation between values that are one time period apart. More generally, a lag k autocorrelation is a correlation between values that are k time periods apart. y_t and y_{t-k} are the variables measured at time t and $t - k$, and μ is the mean of all the samples in the time series. For ACF, possible values range from + 1 to − 1. An ACF of + 1 indicates a perfect positive correlation, which means that both variables move in the same direction.

5. Results

5.1. Training performance

Training results of four state-of-the-art methods are shown in Fig. 5.

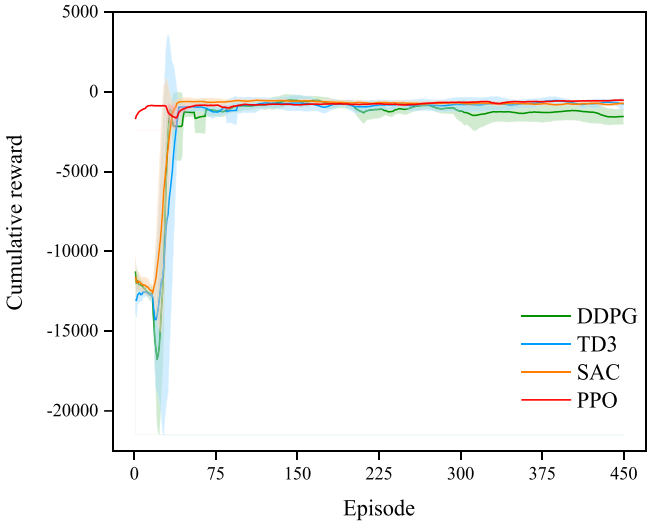

Fig. 5. Comparison of DDPG, SAC, TD3, PPO.

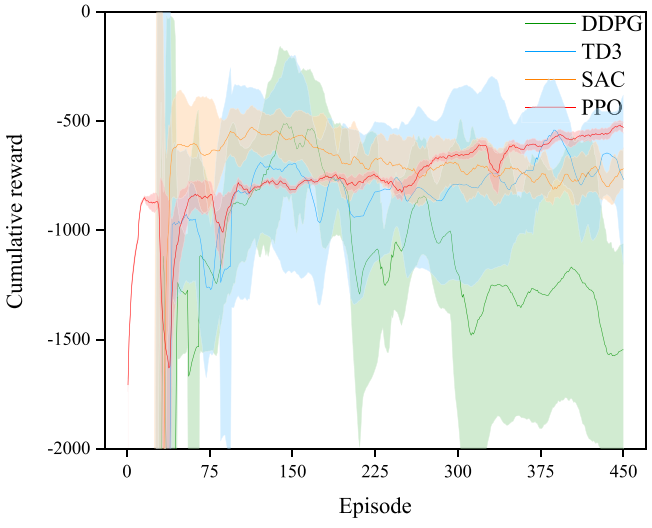

Fig. 6. Comparison of DDPG, SAC, TD3, PPO in terms of cumulative reward over − 2000.

According to the cumulative reward of each episode, each method can converge within 150 episodes. After being trained for 450 episodes, DDPG (green line) has the lowest reward, while PPO (red line) has the highest one. Because each method was repeated 5 times, the mean and standard deviation are used for plotting, and shadow areas in Figs. 5 and 6 represent the standard deviation. As shown in Fig. 5, at 30th episode, TD3 (blue line) and SAC (orange line) have shadow area beyond 0, that means the rewards vary dramatically, resulting in a large the standard deviation. In other words, performance of TD3 and SAC are unstable at that episode. Fig. 6 shows the zoomed in cumulative reward, so that the minimum value of the vertical axis has been set to − 2000. During the training process, the cumulative reward for SAC increased during the first 150 episodes and then decreased to − 720. DDPG has a similar trend as SAC, increasing first, and finally decreasing to − 1500. The average reward for PPO is increasing and approaching − 530. Like PPO, the reward for TD3 has an increasing trend, but converged to around − 720, which is lower than that of PPO. PPO has less shadow area than other methods have, so with different values for random seeds, PPO has a smaller fluctuation in cumulative reward.

The training performance of EPPO (gray line) is shown in Fig. 7. The purpose of adding multiple actors to PPO is to obtain better and stable performance. The cumulative reward of PPO has an obvious increasing trend and converges to − 530, which is the highest reward among all the methods implemented in this paper, but there are three drops at 35th, 80th, 330th episode. For EPPO, the final reward is − 660, which is lower than that of PPO. However, the cumulative reward of EPPO increases without any obvious drops. Moreover, EPPO has a high convergence speed. After only 20 episodes, the cumulative reward reaches − 700, while PPO has a reward of − 870. Therefore, EPPO outperforms PPO in terms of cumulative reward.

5.2. Simulation analysis

In addition to the training performance, the control performance will be evaluated by a simulation. The studied problem has four states (flatness values at four reference points). The control performance at these four states is analyzed separately. In the studied factory, the actual USL and LSL are set as 15 I and − 15 I. However, in our paper, a strict requirement is designed, USL and LSL are set as 5 I and − 5 I. In Fig. 8, all the state values generated by PI and EPPO are in the range of − 15 I and 15 I, which meets the demands in our case study factory. However, given a strict requirement, only state 2 and state 3 of the data collected from two processes using PI and EPPO are between the boundaries of USL and LSL. The optimal flatness is 0, and EPPO has more samples much closer to 0 than PI has. For state 1, the PI based controller has 50 % of the data beyond the USL, while the EPPO based controller has all the data meeting the requirement. Similar to the performance of state 3, for state 4, the EPPO based controller performed well, but the PI based controller failed to effectively control the flatness, because 99 % of the data is beyond the LSL.

Moreover, in Table 3, C_p and C_{pk} are calculated based on the data shown in Fig. 8. The EPPO based controller has higher C_p than what the PI based controller has in all the states, which means that the EPPO controller is more capable of producing products to specifications. The larger C_{pk} is, the less likely it is that any sample will be outside the specification limits. For the EPPO based controller, all the values of C_{pk} in four states are over 2. For the PI based controller, C_{pk} values are much lower, and lower than 1. For state 4, C_{pk} is even negative, meaning that the process will produce output that is outside the customer specification limits. With the EPPO based controller, for each state, C_{pk} is close to C_p, which means the average of the specification is close to the target value.

For the flatness measured at a specific frequency, the values are expected to be close to the optimal value 0 without huge sudden changes. As shown in Fig. 8, the process based on the PI controller (black lines)

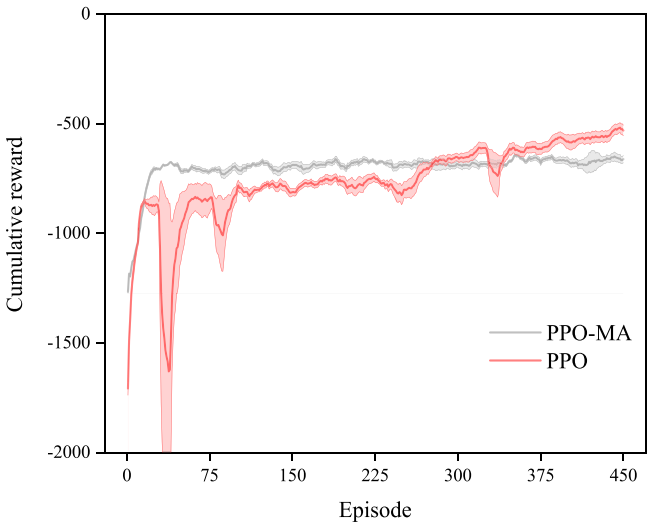

Fig. 7. Comparison of PPO and EPPO.

Table 3
Capability data of PI and EPPO.

Flatness	PI		EPPO	
	C_p	C_{pk}	C_p	C_{pk}
State 1	0.90009	0.07261	2.89666	2.86143
State 2	0.96115	0.72811	2.27545	2.21477
State 3	1.03520	0.90036	6.44785	6.41915
State 4	0.80144	-0.71414	4.45139	4.42305

Table 4
Lag 1 autocorrelation of time series collect from PI and EPPO.

Flatness	PI	EPPO
State 1	0.95024	0.99780
State 2	0.96815	0.99778
State 3	0.94819	0.99781
State 4	0.94662	0.99781

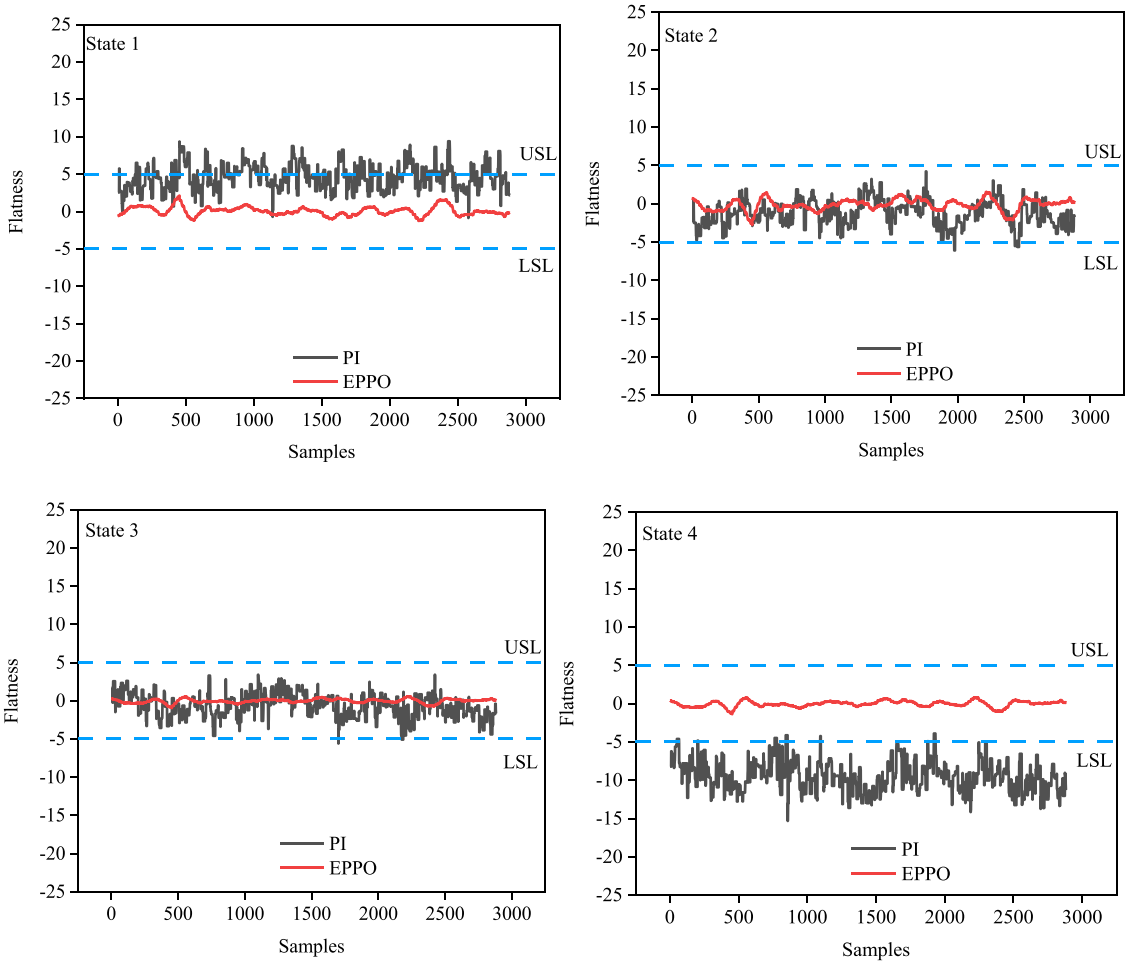

Fig. 8. Capability analysis of the proposed and the existing methods.

produced strips having notable fluctuations in flatness, while the red lines are smoother. Mathematically, based on Eq. (9), lag 1 ACF is computed for smoothness analysis, which is shown in Table 4. For lag 1 ACF, close to 1 means a smoothly varying series. Like the capability analysis, performance of PI and EPPO controllers is analyzed separately by observing the four states. For PI, it has the highest lag 1 ACF of 0.96815 for state 2, while state 3 and state 4 have lower values of 0.94819 and 0.94662. However, for EPPO, state 3 and state 4 have the highest lag 1 ACF values of 0.99781, and state 1 has the lowest value of 0.99780. Moreover, all the states of the EPPO based controller have higher values of lag 1 ACF compared to the PI based controller. It can be concluded that the proposed method (EPPO) outperforms the existing method (PI).

6. Discussion

This paper tried to address real industrial control problems using RL methods, and the results in Section 5 showed that RL can be used for industrial control system. However, this topic also has three main limitations that need to be studied further.

1. The simplified data-driven model used for training can represent the real system only to a limited extent, there could be a bias between the model and the real system. Thus, it is risky to directly use the policy for real control.
2. Using approximated environment on computers for training and evaluation is normal practice in RL literature. As a feasibility test, this paper proved the method is possible to be used for industrial application, bur real tests in the factory are needed in the future.
3. In addition to the stability and capability of the proposed method discussed in Section 5, functionalities, e.g., computational efficiency, are needed for an industrial application. Although the policy (neural network) can be added to (PLCs), a real test is significant.

Although limitations are existing, the current results in the paper demonstrated that RL can be adapted to strip rolling control. Such work is motivated because few publications about RL in strip rolling can be found. Based on the first work proposed in this paper, in the future, we have three main plans to address the above limitations.

1. For the first limitation about the simplified environment model:
   a) Collaborate with experts from the factory, design a near optimal environment model, considering all the possible factors (mentioned in Paragraph 4 of Section 4.3) which are existing in the real system.
   b) Train a policy without an environment using offline reinforcement learning (OLRL). This is done by collecting data from the real production line, and OLRL can directly learn from the data without interacting with an environment model. Because the assumption behind OLRL is that the real data are considered to have all the underlying information generated by the factors mentioned in Paragraph 4 of Section 4.3. Therefore, the simplification problem does not exist in OLRL.

2. For the second limitation about the evaluation:
   a) As mentioned above, if a near optimal environment is available, that will also address the evaluation limitation.
   b) The trained policy is a neural network, and the evaluation does not have complex calculation. In fact, it is functionally possible to add a neural network to PLCs, making it involve in real control. New RL methods with safety precautions are needed.
3. For the limitations in RL algorithm development:
   a) Currently, most of the general RL algorithms were designed purely on computers using simulation environments. Considering specific functionalities, e.g., safety, stability, new algorithms for industrial control are needed.
   b) For industrial applications, the algorithms need to be practicable. Based on high performance computers in the lab, we can modify the existing state-of-the-art algorithms to meet the demands mentioned above. It is important to keep a balance between functionality and practicality for algorithm development. For example, PLCs may not be able to execute complex models, especially if the cycle time is short. The tradeoff between simplicity and real-time performance can be determined empirically for alternative RL algorithms.

7. Conclusions and future work

This paper proposed a novel intelligent policy for flatness control of a strip rolling line. Without first-principles or empirical models, the policy was trained with a model-free reinforcement learning method, using an environment that was generated based on data from a real industrial strip rolling line. To enhance performance, ensemble learning was adopted. A proximal policy optimization with multiple actors was developed to stabilize performance. For the proposed method, capability and smoothness analysis were implemented based on a simulation. Results showed that the proposed method outperformed the existing method.

Our next work will focus on the practical application of the proposed method. By improving the environment and reinforcement learning algorithm, the gap between the experiments and a factory environment can be narrowed. For that, the policy (neural network) will be implemented in a Programmable Logic Controller, aiming not to replace the current controller but rather to work together. Based on the real industrial experiment, the performance of the policy will be improved.

CRediT authorship contribution statement

Jifei Deng: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft, Visualization. **Seppo Sierla:** Investigation, Validation, Writing – review & editing **Jie Sun:** Investigation, Resources, Data Curation, Writing – review & editing. **Valeriy Vyatkin:** Writing – review & editing, Supervision, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by China Scholarship Council (No. 202006080008), the National Natural Science Foundation of China (Grant Nos. 52074085 and U21A20117), the Fundamental Research Funds for the Central Universities (Grant No. N2004010), and the LiaoNing Revitalization Talents Program (XLYC1907065).

References

Bao, Yaoyao, Zhu, Yuanming, Qian, Feng, 2021. A deep reinforcement learning approach to improve the learning performance in process control. Ind. Eng. Chem. Res. 60 (15), 5504–5515. https://doi.org/10.1021/ACS.IECR.0C05678.

Bemporad, Alberto, Bernardini, Daniele, Cuzzola, Francesco Alessandro, Spinelli, Andrea, 2010. Optimization-based automatic flatness control in cold tandem rolling. J. Process Control 20 (4), 396–407. https://doi.org/10.1016/J.JPROCONT.2010.02.003.

Box, George E.P., Jenkins, Gwilym M., Reinsel, Gregory C., Ljung, Greta M., 2015. Time Series Analysis: Forecasting and Control. John Wiley & Sons.

Chen, Chunyu, Cui, Mingjian, Li, Fangxing, Yin, Shengfei, Wang, Xinan, 2021. Model-free emergency frequency control based on reinforcement learning. IEEE Trans. Ind. Inform. 17 (4), 2336–2346. https://doi.org/10.1109/TII.2020.3001095.

Deng, Jifei, Sun, Jie, Peng, Wen, Hu, Yaohui, Zhang, Dianhua, 2019. Application of neural networks for predicting hot-rolled strip crown. Appl. Soft Comput. J. 78, 119–131. https://doi.org/10.1016/j.asoc.2019.02.030.

Duan, Jiajun, Shi, Di, Diao, Ruisheng, Li, Haifeng, Wang, Zhiwei, Zhang, Bei, Bian, Desong, Yi, Zhehan, 2020. Deep-reinforcement-learning-based autonomous voltage control for power grid operations. IEEE Trans. Power Syst. 35 (1), 814–817. https://doi.org/10.1109/TPWRS.2019.2941134.

Fan, Haoren, Zhu, Lei, Yao, Changhua, Guo, Jibin, Lu, Xiaowen, 2019. Deep reinforcement learning for energy efficiency optimization in wireless networks. In: Proceedings of the 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analytics, ICCCBDA 2019, April. Institute of Electrical and Electronics Engineers Inc., pp. 465–71. ⟨https://doi.org/10.1109/ICCCBDA.2019.8725683⟩.

Fujimoto, Scott, Hoof, Herke Van, Meger, David, 2018. Addressing function approximation error in actor-critic methods. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, 4, pp. 2587–601.

Gamal, Omar, Mohamed, Mohamed Imran Peer, Patel, Chirag Ghanshyambhai, Roth, Hubert, 2021. Data-driven model-free intelligent roll gap control of bar and wire hot rolling process using reinforcement learning. Int. J. Mech. Eng. Robot. Res. 10 (7), 349–356. https://doi.org/10.18178/ijmerr.10.7.349-356.

Ginzburg, Vladimir B. (Ed.), 2009. Flat-Rolled Steel Processes: Advanced Technologies. CRC Press.

Guo, Fang, Li, Yongqiang, Liu, Ao, Liu, Zhan, 2020. A reinforcement learning method to scheduling problem of steel production process. J. Phys. Conf. Ser. 1486 (7) https://doi.org/10.1088/1742-6596/1486/7/072035.

Haarnoja, Tuomas, Zhou, Aurick, Abbeel, Pieter, Levine, Sergey, 2018. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, 5, pp. 2976–89.

Han, Zhongyang, Pedrycz, Witold, Zhao, Jun, Wang, Wei, 2020. Hierarchical granular computing-based model and its reinforcement structural learning for construction of long-term prediction intervals. IEEE Trans. Cybern. 52 (1), 666–676. https://doi.org/10.1109/TCYB.2020.2964011.

He, Zhenglei, Tran, Kim Phuc, Thomassey, Sebastien, Zeng, Xianyi, Xu, Jie, Yi, Changhai, 2021. A deep reinforcement learning based multi-criteria decision support system for optimizing textile chemical process. Comput. Ind. 125 (February) https://doi.org/10.1016/J.COMPIND.2020.103373.

Jin, Xin, Li, Changsheng, Wang, Yu, Li, Xiaogang, Xiang, Yongguang, Gu, Tian, 2020. Investigation and optimization of load distribution for tandem cold steel strip rolling process. Metals 10 (5), 677. https://doi.org/10.3390/MET10050677.

Lillicrap, Timothy P., Hunt, Jonathan J., Pritzel, Alexander, Heess, Nicolas, Erez, Tom, Tassa, Yuval, Silver, David, Wierstra, Daan, 2016. Continuous control with deep reinforcement learning. In: Proceedings of the International Conference on Learning Representations (ICLR).

Liu, Chao, Ding, Jinliang, Sun, Jiyuan, 2021. Reinforcement learning based decision making of operational indices in process industry under changing environment. IEEE Trans. Ind. Inform. 17 (4), 2727–2736. https://doi.org/10.1109/TII.2020.3005207.

Liu, Yan Jun, Tang, Li, Tong, Shaocheng, Philip Chen, C.L., Li, Dong Juan, 2015. Reinforcement learning design-based adaptive tracking control with less learning parameters for nonlinear discrete-time MIMO systems. IEEE Trans. Neural Netw. Learn. Syst. 26 (1), 165–176. https://doi.org/10.1109/TNNLS.2014.2360724.

Liu, Guangbiao, Zhou, Jianzhong, Jia, Benjun, He, Feifei, Yang, Yuqi, Sun, Na, 2019. Advance short-term wind energy quality assessment based on instantaneous standard deviation and variogram of wind speed by a hybrid method. Applied Energy 238 (March), 643–667. https://doi.org/10.1016/J.APENERGY.2019.01.105.

Mathieu, N., Potier-Ferry, M., Zahrouni, H., 2017. Reduction of flatness defects in thin metal sheets by a pure tension leveler. Int. J. Mech. Sci. 122 (March), 267–276. https://doi.org/10.1016/J.IJMECSCI.2017.01.036.

Moriyama, Takao, De Magistris, Giovanni, Tatsubori, Michiaki, Pham, Tu. Hoa, Munawar, Asim, Tachibana, Ryuki, 2018. Reinforcement learning testbed for power-consumption optimization. Commun. Comput. Inform. Sci. 946 (October), 45–59. https://doi.org/10.1007/978-981-13-2853-4_4.

Nawfel, Jena L., Englehart, Kevin B., Scheme, Erik J., 2021. A multi-variate approach to predicting myoelectric control usability. IEEE Trans. Neural Syst. Rehab. Eng. 29, 1312–1327. https://doi.org/10.1109/TNSRE.2021.3094324.

Nian, Rui, Liu, Jinfeng, Huang, Biao, 2020. A review on reinforcement learning: introduction and applications in industrial process control. Comput. Chem. Eng. 139 https://doi.org/10.1016/j.compchemeng.2020.106886.

Ning, Zhaolong, Zhang, Kaiyuan, Wang, Xiaojie, Obaidat, Mohammad S., Guo, Lei, Hu, Xiping, Hu, Bin, Guo, Yi, Sadoun, Balqies, Kwok, Ricky Y.K., 2021. Joint computing and caching in 5G-envisioned internet of vehicles: a deep reinforcement learning-based traffic control system. IEEE Trans. Intell. Transp. Syst. 22 (8), 5201–5212. https://doi.org/10.1109/TITS.2020.2970276.

Paakkari, Jussi, 1998. On-Line Flatness Measurement of Large Steel Plates Using Moiré Topography. University of Oulu.

Qiu, Peihua, 2013. Introduction to Statistical Process Control. CRC Press.

Schulman, John, Moritz, Philipp, Levine, Sergey, Jordan, Michael I., Abbeel, Pieter, 2016. High-dimensional continuous control using generalized advantage estimation. In: Proceedings of the International Conference on Learning Representations (ICLR), pp. 1–14.

Schulman, John, Levine, Sergey, Abbeel, Pieter, Jordan, Michael, Moritz, Philipp, 2015. Trust region policy optimization. In: Proceedings of the International Conference on Machine Learning (ICML).

Schulman, John, Wolski, Filip, Dhariwal, Prafulla, Radford, Alec, Klimov, Oleg, 2017. Proximal policy optimization algorithms. ArXiv 1–12.

Shin, Joohyun, Badgwell, Thomas A., Liu, Kuang Hung, Lee, Jay H., 2019. Reinforcement learning – overview of recent progress and implications for process control. Comput. Chem. Eng. 127 (August), 282–294. https://doi.org/10.1016/J.COMPCHEMENG.2019.05.029.

Spielberg, S.P.K., Gopaluni, R.B., Loewen, P.D., 2017. Deep reinforcement learning approaches for process control. In: Proceedings of the 2017 6th International Symposium on Advanced Control of Industrial Processes, AdCONIP 2017. Institute of Electrical and Electronics Engineers Inc, pp. 201–6. ⟨https://doi.org/10.1109/ADCONIP.2017.7983780⟩.

Ståhl, Niclas, Mathiason, Gunnar, Alcacoas, Dellainey, 2021. Using reinforcement learning for generating polynomial models to explain complex data. SN Comput. Sci. 2 (2), 1–11. https://doi.org/10.1007/S42979-021-00488-W.

Sun, Jie, Deng, Jifei, Peng, Wen, Zhang, Dianhua, 2021. Strip crown prediction in hot rolling process using random forest. Int. J. Precis. Eng. Manuf. (0123456789) https://doi.org/10.1007/s12541-020-00454-1.

Sutton, R.S., Barto, A.G., 2018a. Reinforcement Learning: An Introduction. MIT press.

Vanvuchelen, Nathalie, Gijsbrechts, Joren, Boute, Robert, 2020. Use of proximal policy optimization for the joint replenishment problem. Comput. Ind. 119 (August) https://doi.org/10.1016/J.COMPIND.2020.103239.

Wang, Pengfei, Jin, Shuren, Li, Xu, Huang, Huagui, Wang, Haifeng, Zhang, Dianhua, Li, Wentian, Yao, Yulin, 2022a. Optimization and prediction model of flatness actuator efficiency in cold rolling process based on process data. Steel Res. Int. 93 (1) https://doi.org/10.1002/SRIN.202100314.

Wang, Xiaocen, Lin, Min, Tong, Junkai, Liang, Lin, Li, Jian, Zeng, Zhoumo, Liu, Yang, 2022b. Guided wave imaging based on fully connected neural network for quantitative corrosion assessment. In: Annual Review of Progress in Quantitative Nondestructive Evaluation. American Society of Mechanical Engineers Digital Collection. https://doi.org/10.1115/QNDE2021-75020.

Zhang, Zhao, Liu, Shuxin, Liu, Manhua, 2021b. A multi-task fully deep convolutional neural network for contactless fingerprint minutiae extraction. Pattern Recognit. 120 (December) https://doi.org/10.1016/J.PATCOG.2021.108189.

Zhang, Heng, Peng, Qingjin, Zhang, Jian, Gu, Peihua, 2021a. Planning for automatic product assembly using reinforcement learning. Comput. Ind. 130 (September) https://doi.org/10.1016/J.COMPIND.2021.103471.