Neurocomputing 402 (2020) 183–194

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Adaptive neuro-fuzzy PID controller based on twin delayed deep deterministic policy gradient algorithm

Qian Shi, Hak-Keung Lam*, Chengbin Xuan, Ming Chen

Department of Engineering, King's College London, Strand,London WC2R 2LS, U.K.

ARTICLE INFO

Article history:
Received 3 January 2020
Revised 13 March 2020
Accepted 21 March 2020
Available online 8 April 2020

Communicated by Prof. Zidong Wang

Keywords:
Twin delayed deep deterministic policy gradient algorithm
Reinforcement learning
Fuzzy PID controller
Cart-pole system

ABSTRACT

This paper presents an adaptive neuro-fuzzy PID controller based on twin delayed deep deterministic policy gradient (TD3) algorithm for nonlinear systems. In this approach, the observation of the environment is embedded with information of a multiple input single output (MISO) fuzzy inference system (FIS) and have a specially defined fuzzy PID controller in neural network (NN) formation acting as the actor in the TD3 algorithm, which achieves automatic tuning of gains of fuzzy PID controller. From the control perspective, the controller combines the merits of both FIS and PID controller and utilizes reinforcement learning algorithm for optimizing parameters. From the reinforcement learning point of view, embedding the prior knowledge into the fuzzy PID controller incorporated in the actor network helps reduce the learning difficulty in the training process. The proposed method was tested on the cart-pole system in simulation environment with comparison of a linear PID controller, which demonstrates the robustness and generalization of the proposed approach.

1. Introduction

Conventional linear proportional-integral-derivative (PID) controllers are widely implemented in industrial environment because of the simplicity of structure and the robust performance in practical applications. However, PID controllers reveal limitations when systems have non-linearities or uncertainties. Besides, the performance of the controllers has high dependency on tuning of parameters [1,2]. Several approaches have been proposed for finding the proper set of parameters of the controller, like Ziegler-Nichols method [3], ant colony optimization algorithm [4,5] and genetic algorithm [6]. Additionally, intelligent controller such as fuzzy inference system (FIS), which embeds human experience into control actions, was proposed for more complex and challenging control situations because of its outstanding capability of approximating nonlinear systems [1,7]. Fuzzy PID controller embedding the structure of PID controller as well as the human expert knowledge from FIS is found to provide distinguished controlling performance in several researches [1,7,8].

With the rapid development of machine learning algorithms in recent years, reinforcement learning (RL) algorithms have been extensively implemented with control theory to generate innovative control strategies [9–14]. One of the widely utilized methods is Q-learning algorithm [15]. In research [9], Q-learning algorithm was directly implemented to generate control signal to control a self-balanced robot. Younesi and Shayeghi [16] adopted Q-learning algorithm to generate additional force for correcting the output of a pre-tuned PID controller, where the weights of the PID controller remain fixed during the whole training process. There are also several researches that applied Q-learning algorithm as a tuning method to find proper set of parameters for controllers, like multiple PID controllers [17–19] and fuzzy controllers [6,20]. Another approach which combines FIS with Q-learning algorithm is known as Fuzzy Q-learning [21] algorithm. It uses both FIS to generate actions and approximate the Q-function. The antecedent of the fuzzy system is the state of the system, the consequent is the all competitive actions with the local Q-values of all state-action pairs. The fuzzy system is to find the action with the best Q-value. The global action and Q-value are obtained by weighted sum of the chosen actions and the corresponding values in each rule. The Q-function is optimized by minimizing Temporal Difference (TD) error [22], the local Q-value for each state-action pair is updated proportionally according to the global gradient descent. However, the limitation of Q-learning still exists. Both the states and actions are based on discrete spaces, while most of the controlling problems are defined on continuous spaces. The curse of dimensionality [23] becomes a challenging issue, where sparse division could lead to missing of important information or sacrificing controlling precision while dense discretization decreases learning efficiency

* Corresponding author.
E-mail addresses: qian.shi@kcl.ac.uk (Q. Shi), hak-keung.lam@kcl.ac.uk (H.-K. Lam), chengbin.xuan@kcl.ac.uk (C. Xuan), ming.1.chen@kcl.ac.uk (M. Chen).

https://doi.org/10.1016/j.neucom.2020.03.063
0925-2312/© 2020 Elsevier B.V. All rights reserved.

and increases the difficulty in convergence, especially for spaces or actions with high dimensionality [16].

Nevertheless, the actor-critic RL algorithms based on continuous input and output domains could address the issue mentioned above. In [24], fuzzy actor-critic learning (FACL) method is proposed. The controller is based on continuous state space and outputs continuous actions. FIS is used as function approximators for both the critic and actor while actor-critic method is implemented to tune the parameters of the conclusions of the multiple inputs single output (MISO) FIS. Thought the output of the actor is continuous, there are actually a set of discrete actions in each rule with different chosen probabilities, the total continuous action is taken as the weighted sum of the all discrete actions in all rules with the highest chosen probability. Therefore, the curse of dimensionality is still unavoidable. Khodaei et al. [25] designed a neuro-fuzzy controller for the closed-loop control of anesthesia, which combines actor-critic method with a single-input-single-output-TSK-type fuzzy system. Instead of using the actor network to generate the values of the gains of the NN, it adopts emotional learning methods, which updates the coefficients of the controller based on the reward signal of critic and the input force. The proposed method reduces the overshoot during the induction and disturbance rejection phase in controlling closed-loop anesthesia. Sedighizadeh and Rezazadeh [26] adopted the actor-critic method by having the actor and critic share the same radial basis function (RBF) network. The simulation results showed this innovative controller performs better than conventional PID controller. Wu et al. [27] applied deterministic policy gradient (DPG) method for the depth control of UAV. Deep deterministic policy gradient (DDPG) [23] algorithm, one of actor-critic algorithms, adopts the experience replay method to break the correlations among experience samples and uses separate target networks as implemented in the deep Q-network (DQN) [28] algorithm to stabilize the learning process, but generates deterministic policy. It is also found high potential in synthesizing traditional controllers. Carlucho et al. [11] applied DDPG algorithm directly to generating control policy based on the data collected by sensors to control low-level autonomous underwater vehicles (AUV) and showed successful application in real experiment. In the research of [29], a single input interval type-2 fuzzy PI (SIT2-FPI) controller is implemented as major controller, which has fixed proportional and integral gains as to stabilize the certain criteria of the system to required value. DDPG algorithm was used to generate regulatory continuous values of proportional and integral gains correspondingly to optimize the controlling performance. The results are stated to have less transient settling time and less overshooting or undershooting when switching between different objective targets. Though the DDPG algorithm shows capability of improving the performance, the DDPG algorithm only acts as an assistant role with a tuned fuzzy controller accomplishing the main control task. Designing the main controller to work properly still requires prior knowledge and additional effort to tune parameters. Besides, having DDPG to learn from scratch instead of being an assistant section would increase the difficulty of convergence. Furthermore, there are only two parameters need to be tuned, increasing the number of parameters required tuning would cause extra computational burden.

As stated above, the DDPG algorithm enhances the performance of traditional controllers while fuzzy-PID controller combining the merits of both PID controller and fuzzy controller is proved to offer superior control performance in several researches. In this paper, we applied an updated version of DDPG algorithm named twin delayed deep deterministic policy (TD3) [30] with an adaptive neuro-fuzzy PID controller to optimize the controlling performance. Instead of having actor network of TD3 algorithm to generate multiple parameters for the fuzzy PID controller, a specially designed fuzzy PID controller in NN formation acts as the actor role in the TD3 algorithm, where the weights of the actor NN become the parameters (PID gains) that need to be tuned and the output directly becomes control force. Therefore, the actor is equivalent to a fuzzy PID controller that tuning of PID gains is achieved automatically by updating the weights of actor network in the learning process. In this way, the TD3 algorithm could efficiently find optimal control strategy by embedding actor NN with prior knowledge of PID controllers and FIS. The cart-pole system is chosen to test the proposed method considering it is a benchmark problem in both reinforcement learning and control domain with characteristic of high non-linearity. A neuro-PID controller is also designed for comparison purpose with details provided in Section 4.

The contributions of the paper can be summarized as follows,

1. An adaptive neuro-fuzzy PID controller, implemented as an actor, based on TD3 algorithm is proposed.
2. A linear PID controller based on TD3 algorithm is provided for comparison purpose.
3. A cart-pole simulation environment is provided as to test the feasibility of the proposed approach.
4. The generalization and robustness tests were conducted to prove the advantages of the proposed method.

The remainder of the paper is organised as follows. Section 2 provides the basic knowledge relevant to this research; Section 3 presents the details of the proposed approach; Section 4 shows the simulation results of the proposed methodology and the conclusions are drawn in Section 5.

2. Preliminaries

This section introduces some basic knowledge related to this paper, which includes fuzzy PID controller, TD3 algorithm and the cart-pole system.

2.1. Fuzzy PID controllers

2.1.1. Linear PID controllers

As shown in Fig. 1, a linear PID controller consists of three sections which are proportional, integral and derivative section. The mathematical expression of a linear PID controller in discrete time domain is

$$u(t_k) = K_P e(t_k) + K_I \sum_{i=0}^{k} e(t_i)\Delta t_k + K_D \frac{e(t_k) - e(t_{k-1})}{\Delta t_k}, \tag{1}$$

where t_k is the kth time step; $u(t_k)$ is the output in kth time step; $e(t_k)$ is the error in kth time step; Δt_k is the interval of sampling time in simulation; $K_P \in \mathcal{R}$, $K_I \in \mathcal{R}$, $K_D \in \mathcal{R}$ are the proportional, integral and derivative gain, respectively. R refers to the set point.

2.1.2. Fuzzy inference system

A multiple inputs-single output (MISO) FIS consists of a set of fuzzy rules in the following format:

Rule i: IF $x_1(t)$ is M_1^i AND ... AND $x_N(t)$ is M_N^i THEN $y(\boldsymbol{x}(t)) = y_i(t)$, where $x_j(t)$, $j = 1, 2, \ldots, N$, is linguistic variable; M_j^i is the fuzzy term for the jth linguistic variable $x_j(t)$ in the ith rule; $N \in \mathcal{N}$, which is the number of the linguistic variables; $y(\boldsymbol{x}(t))$ is the output of the fuzzy inference system; $y_i(t)$ is the output of ith rule.

The final output of FIS is

$$y(\boldsymbol{x}(t)) = \sum_{i=1}^{p} w_i(\boldsymbol{x}(t)) y_i(t), \tag{2}$$

$\boldsymbol{x}(t) = [x_1(t), x_2(t), \cdots, x_N(t)]$; $p \in \mathcal{N}$ is the total number of fuzzy rules; $w_i(\boldsymbol{x}(t))$ is the normalized weight of ith rule,

$$w_i(\boldsymbol{x}(t)) = \frac{\prod_{n=1}^{N} \mu_{M_n^i}(\boldsymbol{x}(t))}{\sum_{m=1}^{p} \left(\prod_{n=1}^{N} \mu_{M_n^m}(\boldsymbol{x}(t))\right)} \forall i, \tag{3}$$

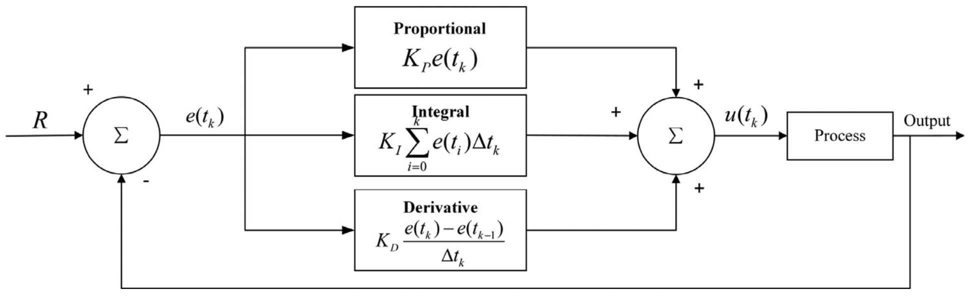

Proportional
$K_P e(t_k)$
Integral
$K_I \sum_{i=0}^{k} e(t_i)\Delta t_k$
Derivative
$K_D \frac{e(t_k)-e(t_{k-1})}{\Delta t_k}$
R
$e(t_k)$
$u(t_k)$
Process
Output
+
-

Fig. 1. Structure of PID controller.

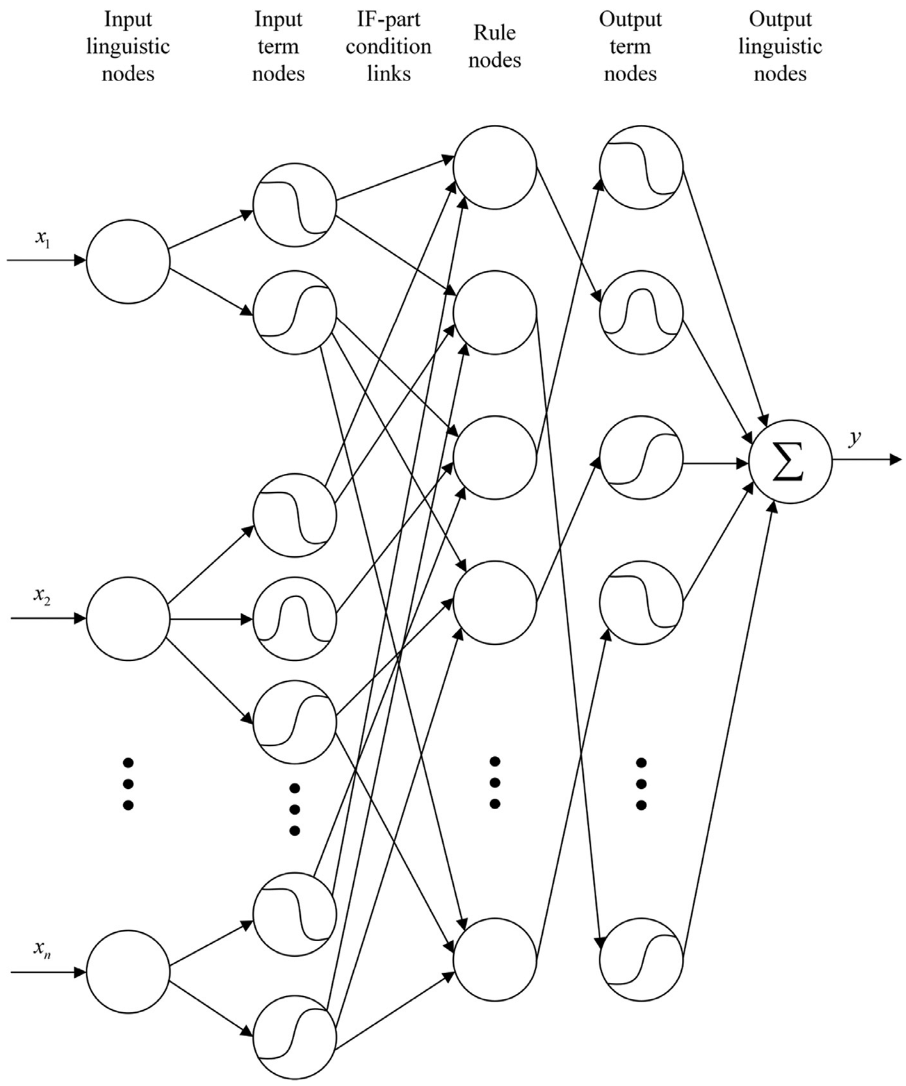


Fig. 2. Fuzzy inference system in NN formation, i.e., neuro-fuzzy network.

where $w_i(\boldsymbol{x}(t)) \geq 0$, $\forall i$ and $\sum_{i=1}^{p} w_i(\boldsymbol{x}(t)) = 1$; $\mu_{M_n^i}(\boldsymbol{x}(t)), n = 1, 2, \cdots, N$ is the grade of membership for the fuzzy term M_n^i.

A FIS in neural network formation (neuro-fuzzy network) is shown in Fig. 2.

2.1.3. Fuzzy PID controller

Fig. 3 presents the structure of fuzzy PID controller, where the process indicates the system controlled by the fuzzy PID controller. The fuzzy PID controller in discrete time domain can be expressed in the following format:

Rule i: IF $e(t_k)$ is M_1^i AND $de(t_k)$ is M_2^i AND $se(t_k)$ is M_3^i THEN $u(\boldsymbol{x}(t_k)) = u_i(t_k)$, where $de(t_k)$ is the change of error in the kth time step, $de(t_k) = e(t_k) - e(t_{k-1})$; $s_e(t_k)$ is the accumulated error until kth time step, $se(t_k) = \sum_{m=0}^{k} e(t_m)$; $u_i(t_k)$ is the output of ith PID controller. The final output of the fuzzy PID controller can be expressed as

$$u(\boldsymbol{x}(t_k)) = \sum_{i=1}^{p} w_i(\boldsymbol{x}(t_k)) u_i(t_k), \tag{4}$$

where $p \in \mathcal{N}$ is the total number of fuzzy rules.

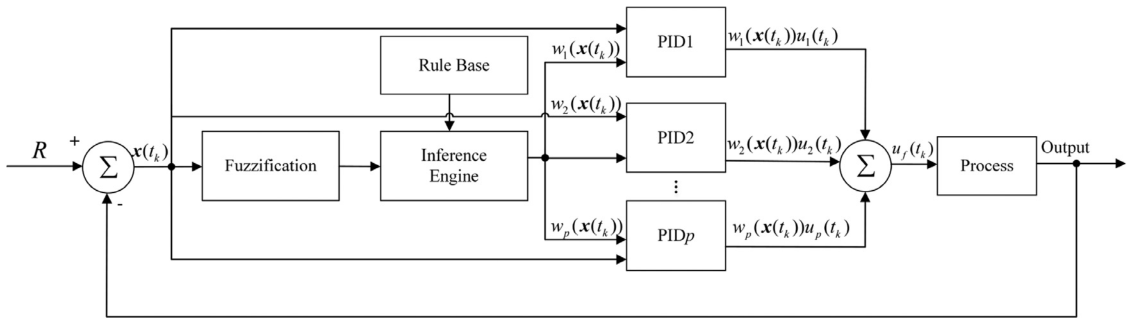


Fig. 3. Structure of fuzzy PID controller.

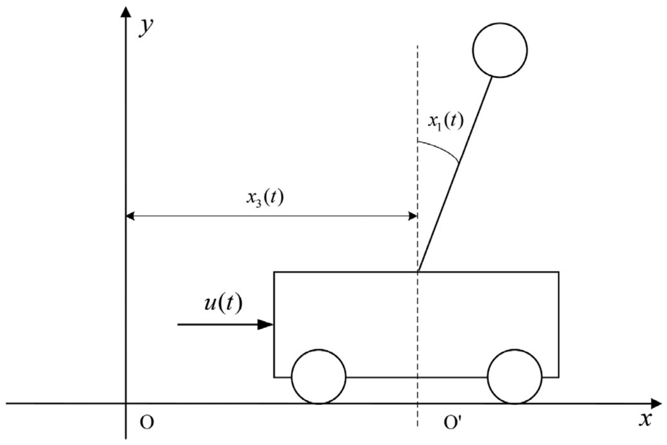


Fig. 4. Cart-pole system.

2.2. Cart-pole system

As shown in Fig. 4, the cart-pole system consists of a cart connected with a pole through a hinge. The cart can only move in x direction while the pole rotates in $x-y$ plane. A horizontal force $u(t)$ is applied to the cart in the same direction as the movement of the cart to balance the pole. The stable equilibrium is when pole stays up right and the cart goes back to origin. Taking the state of the cart-pole system as $\boldsymbol{x_{cp}}(t) = [x_1(t), x_2(t), x_3(t), x_4(t)]$, the dynamic model of the cart-pole system is given as follows [31],

$$\dot{x}_1(t) = x_2(t), \tag{5}$$

$$\dot{x}_2(t) = \frac{\begin{pmatrix} -F_1(M+m)x_2(t) - m^2l^2x_2(t)^2\sin x_1(t) \\ \cos x_1(t) + F_0mlx_4(t)\cos x_1(t) + (M+m)mgl \\ \sin x_1(t) - ml\cos x_1(t)u(t) \end{pmatrix}}{(M+m)(J+ml^2) - m^2l^2(\cos x_1(t))^2}, \tag{6}$$

$$\dot{x}_3(t) = x_4(t), \tag{7}$$

$$\dot{x}_4(t) = \frac{\begin{pmatrix} F_1mlx_2(t)\cos x_1(t) + (J+ml^2)mlx_2(t)^2 \\ \sin x_1(t) - F_0((J+ml^2)x_4(t) - m^2gl^2 \\ \sin x_1(t)\cos x_1(t) + (J+ml^2)u(t) \end{pmatrix}}{(M+m)(J+ml^2) - m^2l^2(\cos x_1(t))^2}, \tag{8}$$

where $x_1(t)$ is the displacement of angle (rad); $x_2(t)$ is the angular velocity (rad/sec); $x_3(t)$ is the position of the cart (m); $x_4(t)$ is the linear velocity of the cart (m/s); $u(t)$ is the force applied to the cart (N); M is the mass of the cart (kg); m is the mass of the pendulum (kg); l is the half length of the pendulum (m); J is the moment of inertia of the pendulum, $J = \frac{1}{3}ml^2$ kgm^2; F_0 is the friction factor of the cart (N/m/s); F_1 is the friction factor of the pendulum (N/rad/s); g is the gravity acceleration, $g = 9.8\text{m/s}^2$.

2.3. TD3 algorithm

TD3 algorithm is an off-line RL algorithm based on DDPG proposed in 2015 [23]. This approach adopted a similar method implemented in Double-DQN [30] to reduce the overestimation in function approximation, delaying update frequency in actor network and adding noises to target actor network to release the sensitivity and instability in DDPG. The procedure of TD3 is shown in Algorithm 1 .

Algorithm 1: TD3 Algorithm.

```
1  Initialize critic networks $Q_1(s,a|\theta_1)$ and $Q_2(s,a|\theta_2)$, actor
   network $\pi(s|\phi)$;
2  Initialize target critic networks $Q_1'$, $Q_2'$ with $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$
   and target actor network $\pi'$ with $\phi' \leftarrow \phi$;
3  Initialize replay buffer $\mathcal{R}$;
4  For every episode:
5  Initialize state $s$
6  repeat
7      Select action $a \sim \pi(s|\phi) + \mathcal{N}(0,\sigma)$;
8      Observe next state $s'$ and reward $r$
9      Store transition tuple $(s,a,r,s')$ in $\mathcal{R}$;
10     Sample mini-batch of $K$ transitions;
11     Get $a_i' \sim \pi'(s_i'|\phi') + \epsilon$,where $\epsilon \sim \text{clip}(\mathcal{N}(0,\sigma), -c, c)$;
12     Set $y_i = r_i + \gamma\min\{Q_{1i}'(s_i', a_i'|\theta_1'), Q_{2i}'(s_i', a_i'|\theta_2')\}$;
13     Update critic network $Q_m$ by minimizing loss
14     $L_m = \frac{1}{K}\sum_{i=1}^{K}(y_i - Q_m(s_i, a_i|\theta_m))^2$, $m = 1, 2$;
15     Every $d$ steps:
16         Update actor network $\pi$ following gradient
17         $\nabla_\phi = \frac{1}{K}\sum_{i=1}^{K}\nabla_a Q_1(s,a|\theta_1)|_{s=s_i, a=\pi(s_i|\phi)}\nabla_\phi\pi(s_i|\phi)$;
18         Update target critic networks and target actor network:
19         $\theta_m' \leftarrow \tau\theta_m + (1-\tau)\theta_m'$, $m = 1, 2$
20         $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$;
21     $s \leftarrow s'$
22 until $s$ reaches terminal state $s_T$;
```

3. Adaptive neuro-fuzzy PID controller based on TD3 algorithm

In this section, the details of the adaptive neuro-fuzzy PID network based on TD3 algorithm are provided. As shown in Fig. 5, the neuro-fuzzy PID network is consisted of two sections which are neuro-fuzzy network and neuro-PID network. The yellow block on the left demonstrates the neuro-fuzzy network while the orange block on the right shows the neuro-PID network, the details of these two networks are depicted in Fig. 6. The state of the environment $\boldsymbol{s}(t_k)$ is input to neuro-fuzzy network and generates fuzzy state $\boldsymbol{s_f}(t_k)$ which embeds the information of FIS in current state. The neuro-PID network working as the actor network in TD3 takes

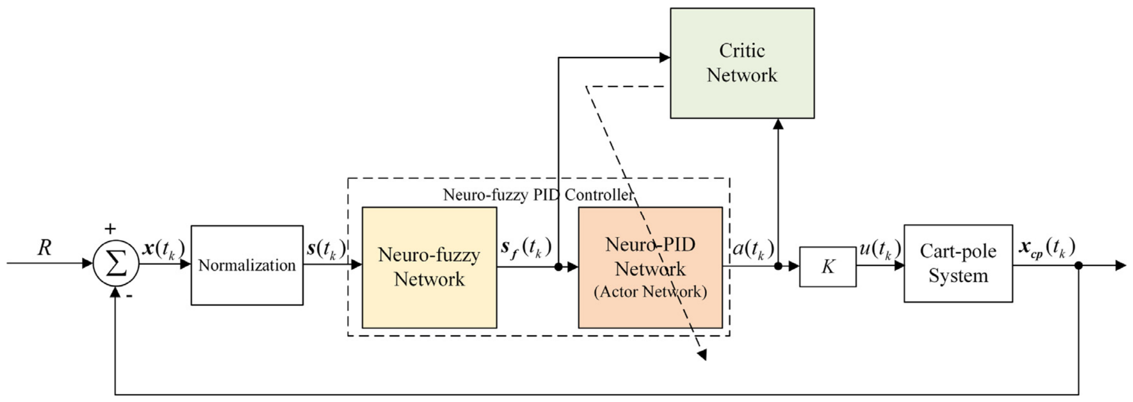


Fig. 5. Structure of control system.

the fuzzy state $\boldsymbol{s_f}(t_k)$ as input and generates action $a(t_k)$. The action value $a(t_k)$ is multiplied with scale factor K to generate control signal $u(t_k)$ for balancing the system. The state and action are taken by critic network, which approximates the value function of state-action pairs and affects the action generated by actor network (neuro-PID network in Fig. 5) while the parameters of neuro-fuzzy network keep fixed during the training process. The details of the structure of neuro-fuzzy PID controller designed for cart-pole system are introduced in Section 3.1 while the training procedure is described in Section 3.2.

3.1. Neuro-fuzzy PID network

3.1.1. Neuro-fuzzy network

The neuro-fuzzy network is the transformation of the FIS, each layer has a corresponding meaning in the FIS. Therefore, the structure of this network is determined by the predefined fuzzy system. In the input layer, there are six variables that are chosen to present the state of the cart-pole system, which can be expressed as $\boldsymbol{s}(t_k) = \boldsymbol{x}(t_k)/\boldsymbol{X}$, where $\boldsymbol{x}(t_k) = [e_\theta(t_k), de_\theta(t_k), se_\theta(t_k), e_x(t_k), de_x(t_k), se_x(t_k)]$. $e_\theta(t_k) = R_\theta - x_1(t_k)$ is the error of the angle, which is the difference between the pole and the reference point, $R_\theta = 0$ in this case; $de_\theta(t_k) = e_\theta(t_k) - e_\theta(t_{k-1})$ is the change of angle error in the kth time step; $se_\theta(t_k) = \sum_{i=0}^{k} e_\theta(t_i)$ is the accumulated angle error until time step t_k. Similarly, $e_x(t_k) = R_x - x_3(t_k)$ is the position error, which is the difference between the cart and the reference point, $R_x = 0$ in this case; $de_x(t_k) = e_x(t_k) - e_x(t_{k-1})$ is the change of position error in the k^{th} time step; $se_x(t_k) = \sum_{i=0}^{k} e_x(t_i)$ is the accumulated position error until time step t_k; $\boldsymbol{X} = [X_1, X_2, \cdots, X_6]$ is a constant normalization vector which guarantees all state variables are within $[-1, 1]$. In the rest of the paper, the variable of time t will be omitted for the simplicity of expression. As depicted in Fig. 6, the inputs of neuro-fuzzy network are separated as two sets of variables, which are variables related to the angle of the pole $\boldsymbol{s_a} = [s_1, s_2, s_3]$ and ones related to the position of the cart $\boldsymbol{s_p} = [s_4, s_5, s_6]$. In the input linguistic layer, the input term nodes are partially connected to the input linguistic nodes, which indicates the fuzzy terms of each input linguistic variable in the FIS. The number of the input term nodes is determined by the number of fuzzy terms each linguistic variable has. s_1 has two fuzzy terms SMALL and LARGE; s_2 has three fuzzy terms SMALL, MEDIUM and LARGE; s_3 has two fuzzy terms SMALL and LARGE. The antecedent membership functions of $\boldsymbol{s}_a$ are shown in Fig. 7, where Z-shaped and S-shaped membership functions are used for the fuzzy terms of state s_1; Z-shaped, S-shaped and triangular membership functions are chosen for fuzzy terms of state s_2; Z-shaped and S-shaped membership functions are chosen for the fuzzy terms of state s_3. Similarly, the fuzzification of variables related to the cart are as follows, s_4 has three fuzzy terms SMALL, MEDIUM and LARGE; s_5 has two fuzzy terms SMALL and LARGE; s_6 has three fuzzy terms SMALL, MEDIUM and LARGE. The membership functions for $\boldsymbol{s_p}$ are shown in Fig. 8, where triangular and S-shaped membership functions are used for the fuzzy terms of state s_4; Z-shaped and S-shaped membership functions are chosen for fuzzy terms of state s_5; Z-shaped, triangular and S-shaped membership functions are chosen for the fuzzy terms of state s_6.

In the following layer, the number of the rule nodes indicates the number of fuzzy rules in the FIS while the connections between rule nodes and input term nodes illustrate the condition links in the IF part. The number of fuzzy rules related to $\boldsymbol{s_a}$ is $N_a = 2 \times 3 \times 2 = 12$ rules and the number of rules related to $\boldsymbol{s_p}$ is $N_p = 3 \times 2 \times 3 = 18$ rules. Therefore, there are $N_a + N_p = 12 + 18 = 30$ nodes in this layer. Each rule node generates the normalized weight of the corresponding rule. Weights related to pole are in the form

$$w_{1i}(\boldsymbol{s_a}) = \frac{\prod_{k=1}^{3} \mu_{M_k^i}(\boldsymbol{s_a})}{\sum_{n=1}^{N_a} \left(\prod_{k=1}^{3} \mu_{M_k^n}(\boldsymbol{s_a})\right)}, \tag{9}$$

where $i = 1, 2, \cdots, N_a$, N_a is the number of fuzzy rules related to $\boldsymbol{s_a}$, $N_a = 12$ in this case. The normalized weights for each rule related to cart are in the form

$$w_{2j}(\boldsymbol{s_p}) = \frac{\prod_{k=1}^{3} \mu_{M_k^j}(\boldsymbol{s_p})}{\sum_{n=N_a+1}^{N_a+N_p} \left(\prod_{k=1}^{3} \mu_{M_k^n}(\boldsymbol{s_p})\right)}, \tag{10}$$

where $j = N_a + 1, N_a + 2, \cdots, N_a + N_p$, N_p is the number of fuzzy rules related to $\boldsymbol{s_p}$, $N_p = 18$. The fuzzy state is in the form $\boldsymbol{s_f} = [\boldsymbol{s_{fa}}, \boldsymbol{s_{fp}}]$, where $\boldsymbol{s_{fa}} = [w_{11}s_1, w_{11}s_2, w_{11}s_3, \cdots, w_{1N_a}s_1, w_{1N_a}s_2, w_{1N_a}s_3]$ and $\boldsymbol{s_{fp}} = [w_{2(N_a+1)}s_4, w_{2(N_a+1)}s_5, w_{2(N_a+1)}s_6, \ldots, w_{2(N_a+N_p)}s_4, w_{2(N_a+N_p)}s_5, w_{2(N_a+N_p)}s_6]$.

The output of each rule node is multiplied with the corresponding state, which generates the fuzzy output. The number of nodes in the output layer is $3(N_a + N_p) = 3 \times 30 = 90$ in this case.

3.1.2. Neuro-PID network

Based on the output of neuro-fuzzy network, $\boldsymbol{s_f}$ embeds both the information of states of the environment and the truth values of fuzzy rules in FIS. The actor network (neuro-PID network in Fig. 6) is designed in the form that combines multiple linear PID controllers. Referring to Fig. 6, the outputs of the first hidden layer are expressed as

$$g_i^1 = \begin{cases} \sum_{j=1}^{3} w_{ij}^1 w_{1i}(\boldsymbol{s_a}) s_j, \, i = 1, 2, \cdots, N_a \\ \sum_{j=1}^{3} w_{ij}^1 w_{2i}(\boldsymbol{s_p}) s_{j+3}, \, i = N_a+1, N_a+2, \ldots, N_a+N_p \end{cases} \tag{11}$$

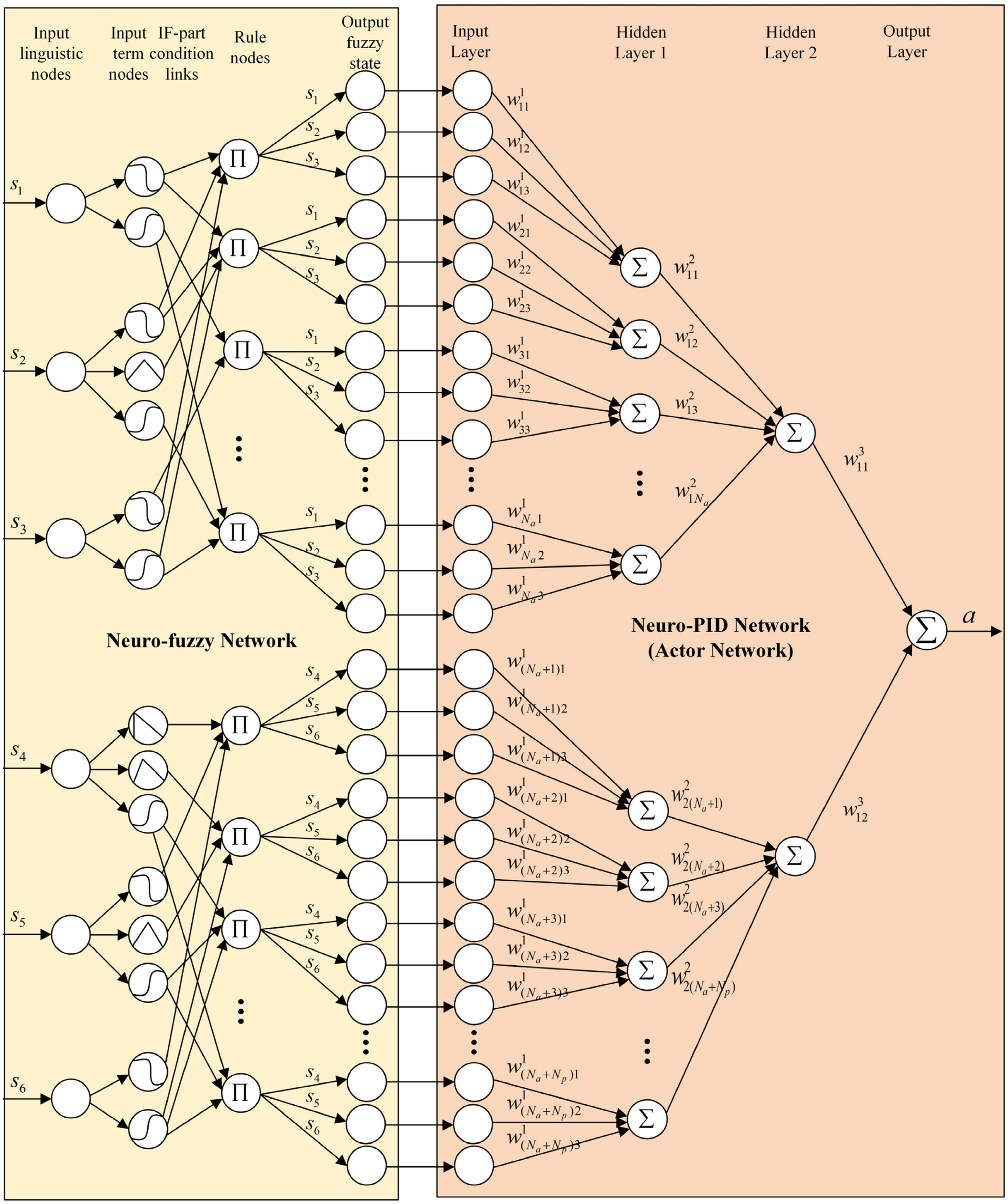


Fig. 6. Structure of neuro-fuzzy PID network.

The outputs of the second hidden layer are

$$g_i^2 = \begin{cases} \sum_{j=1}^{N_a} w_{1j}^2 g_j^1, i = 1 \\ \sum_{l=N_a+1}^{N_a+N_p} w_{2l}^2 g_l^1, i = 2. \end{cases} \tag{12}$$

The final output of the NN is

$$a = \text{clip}\left(\sum_{i=1}^{2} w_{1i}^3 g_i^2, -0.8, 0.8\right), \tag{13}$$

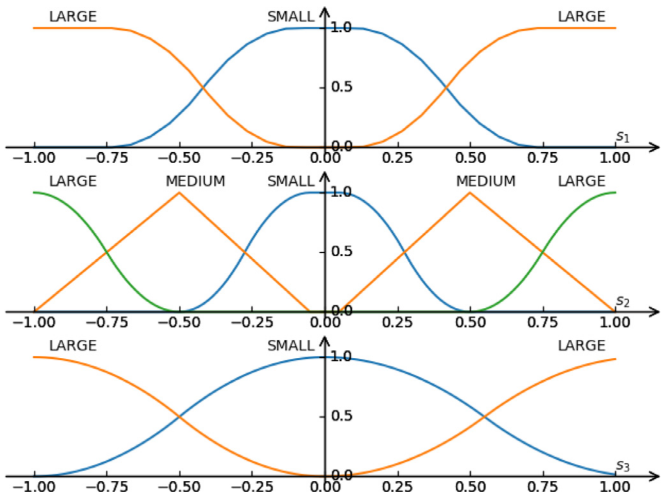


Fig. 7. Membership functions of $\boldsymbol{s}_a$.

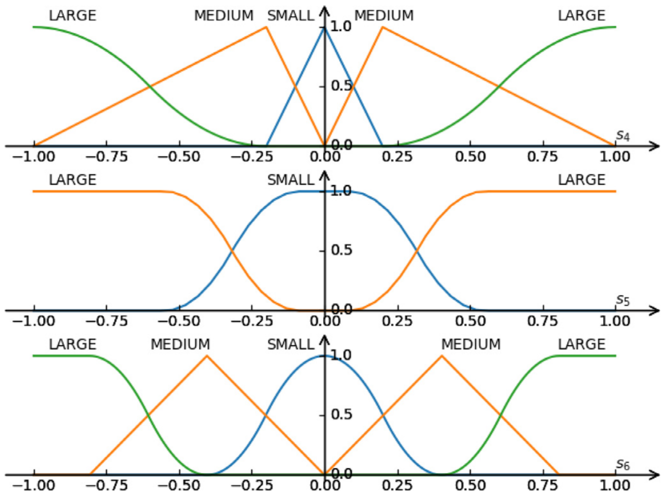


Fig. 8. Membership functions of $\boldsymbol{s}_p$.

where clip(x, a, b) defines as

$$\text{clip}(x, a, b) = \begin{cases} a, \text{ if } x \le a \\ x, \text{ if } a < x < b \\ b, \text{ if } x \ge b, \end{cases} \tag{14}$$

the value of 0.8 is chosen based on the experiment test results as to stabilize the learning process.

The force that applied to the cart-pole system is

$$u = Ka, \tag{15}$$

where $K = 1500$. The force then can be expressed as

$$\begin{aligned} u &= K\left(w_{11}^3 g_1^2 + w_{12}^3 g_2^2\right) \\ &= K\left(w_{11}^3 \sum_{j=1}^{N_a} w_{1j}^2 g_j^1 + w_{12}^3 \sum_{l=N_a+1}^{N_a+N_p} w_{2l}^2 g_l^1\right) \\ &= K\left(w_{11}^3 \sum_{j=1}^{N_a} w_{1j}^2 \sum_{k=1}^{3} w_{jk}^1 w_{1j}(\boldsymbol{s_a}) s_k \right. \\ &\quad \left. + w_{12}^3 \sum_{l=N_a+1}^{N_a+N_p} w_{2l}^2 \sum_{q=1}^{3} w_{lq}^1 w_{2l}(\boldsymbol{s_p}) s_{q+3}\right) \\ &= \sum_{j=1}^{N_a} w_{1j}(\boldsymbol{s_a})(K w_{11}^3 w_{1j}^2 w_{j1}^1 s_1 + K w_{11}^3 w_{1j}^2 w_{j2}^1 s_2 \\ &\quad + K w_{11}^3 w_{1j}^2 w_{j3}^1 s_3) + \sum_{l=N_a+1}^{N_a+N_p} w_{2l}(\boldsymbol{s_p})(K w_{12}^3 w_{2l}^2 w_{l1}^1 s_4 \\ &\quad + K w_{12}^3 w_{2l}^2 w_{l2}^1 s_5 + K w_{12}^3 w_{2l}^2 w_{l3}^1 s_5)) \\ &= \sum_{j=1}^{N_a} w_{1j}(\boldsymbol{s_a})(K_{Pj} s_1 + K_{Dj} s_2 + K_{Ij} s_3) \\ &\quad + \sum_{l=N_a+1}^{N_a+N_p} w_{2l}(\boldsymbol{s_p})(K_{Pl} s_4 + K_{Dl} s_5 + K_{Il} s_5)) \end{aligned} \tag{16}$$

Table 1
Rule-base table of s_2 and s_3 if s_1 is SMALL.

s_3		
s_2	SMALL	LARGE
SMALL	u_{a1}	u_{a2}
MEDIUM	u_{a9}	u_{a10}
LARGE	u_{a3}	u_{a4}

Table 2
Rule-base table of s_2 and s_3 if s_1 LARGE.

s_3		
s_2	SMALL	LARGE
SMALL	u_{a5}	u_{a6}
MEDIUM	u_{a11}	u_{a12}
LARGE	u_{a7}	u_{a8}

Table 3
Rule-base table of s_4 and s_6 if s_5 is SMALL.

s_6			
s_4	SMALL	MEDIUM	LARGE
SMALL	u_{p1}	u_{p13}	u_{p2}
MEDIUM	u_{p9}	u_{p17}	u_{p10}
LARGE	u_{p5}	u_{p15}	u_{p6}

Table 4
Rule-base table of s_4 and s_6 if s_5 is LARGE.

s_6			
s_4	SMALL	MEDIUM	LARGE
SMALL	u_{p3}	u_{p14}	u_{p4}
MEDIUM	u_{p12}	u_{p18}	u_{p12}
LARGE	u_{p7}	u_{p16}	u_{p8}

where $K_{Pj} = K w_{11}^3 w_{1j}^2 w_{j1}^1$, $K_{Dj} = K w_{11}^3 w_{1j}^2 w_{j2}^1$, $K_{Ij} = K w_{11}^3 w_{1j}^2 w_{j3}^1$, $K_{Pl} = K w_{12}^3 w_{2l}^2 w_{l1}^1$, $K_{Dl} = K w_{12}^3 w_{2l}^2 w_{l2}^1$, $K_{Il} = K w_{12}^3 w_{2l}^2 w_{l3}^1$. Therefore, Eq. (16) can be presented as

$$u = \sum_{j=1}^{N_a} w_{1j}(\boldsymbol{s_a}) u_{aj} + \sum_{w=1}^{N_p} w_{2(w+N_a)}(\boldsymbol{s_p}) u_{pw}, \tag{17}$$

where

$$u_{aj} = K_{Pj} s_1 + K_{Dj} s_2 + K_{Ij} s_3, \tag{18}$$

$$u_{pw} = K_{P(w+N_a)} s_4 + K_{D(w+N_a)} s_5 + K_{I(w+N_a)} s_6. \tag{19}$$

Accordingly, the rule base of $\boldsymbol{s_a}$ is shown in Tables 1 and 2 while the rule base of $\boldsymbol{s_p}$ is shown in Tables 3 and 4. The consequent membership function of each rule is chosen as singleton membership function. The outputs shown in the rule bases from Tables 1 to 4 corresponding to the variables in Eqs. (18) and (19), which are $u_{aj}, j = 1, 2, \ldots, N_a$ and $u_{pw}, w = 1, 2, \ldots, N_p$, respectively. These variables stand for the outputs u_i in the concept of fuzzy PID controller expressed in Eq. (4), which can not be found as any exact variables shown in Fig. 6.

Therefore, the neuro-fuzzy network and neuro-PID network combine together to formulate adaptive neuro-fuzzy PID controller. To be noticed, the weights of neuro-fuzzy network are fixed ones while the weights of the neuro-PID network are the gains of fuzzy PID controller, which are updated by TD3 algorithm during the learning process.

Remark 1. The reason to keep the weights of neuro-fuzzy network fixed while updating the weights of neuro-PID network is for the consideration of updating parameters in the actor network. As presented in TD3 algorithm, updating actor network follows the gradient $\nabla = \frac{1}{K}\sum_{i=1}^{K}\nabla_a Q_1 \nabla_{s_f} a$. If the weights in both neuro-fuzzy network are not fixed, the gradient for updating actor network will be $\nabla' = \frac{1}{K}\sum_{i=1}^{K}\nabla_a Q_1 \nabla_{s_f} a \nabla_s s_f$. However, as shown in Fig. 6, it could be noticed that in neuro-fuzzy network, the gradient $\nabla_s s_f$ related to neuro-fuzzy network is difficult to calculate considering the fuzzy membership functions in the input term nodes and the product operation between rule nodes and output fuzzy state. Therefore, only the weights of neuro-PID networks are updated while the weights in neuro-fuzzy network remain fixed.

3.2. Training procedure

The training procedure of the proposed controller is shown in Algorithm 2. The reward function in the training process is designed as follows.

Algorithm 2: TD3 Algorithm with Neuro-fuzzy PID as Actor.

```
1  Initialize critic networks $Q_1(s,a|\theta_1)$ and $Q_2(s,a|\theta_2)$,
   neuro-fuzzy PID actor network $\pi_f(s|\phi_f)$;
2  Initialize target critic networks $Q_1'$, $Q_2'$ with $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$
   and target actor network $\pi_f'$ with $\phi_f' \leftarrow \phi_f$;
3  Initialize replay buffer $\mathcal{R}$;
4  For every episode:
5  Initialize state $s$
6  repeat
7      Observe fuzzy state $s_f$;
8      Select action $a \sim \pi_f(s_f|\phi_f) + \mathcal{N}(0,\sigma)$;
9      Observe next state $s'$ and reward $r$
10     Store transition tuple $(s,a,r,s')$ in $\mathcal{R}$;
11     Sample mini-batch of $K$ transitions;
12     Get fuzzy states of mini-batch $s_{fi}$ and $s_{fi}'$ $(i = 1,2,\ldots,K)$;
13     Get $a_i' \sim \pi_f'(s_{fi}'|\phi') + \epsilon$,where $\epsilon \sim \text{clip}(\mathcal{N}(0,\sigma), -c, c)$
14     Set $y_i = r_i + \gamma \min\{Q_{1i}'(s_{fi}', a_i'|\theta_1'), Q_{2i}'(s_{fi}', a_i'|\theta_2')\}$
15     Update critic network $Q_m$ by minimizing loss
16     $L_m = \frac{1}{K}\sum_{i=1}^{K}(y_i - Q_m(s_{fi}, a_i|\theta_m))^2$, $m = 1,2$;
17     Every $d$ steps:
18         Update actor network $\pi_f$ following gradient
19         $\nabla_{\phi_f} = \frac{1}{K}\sum_{i=1}^{K}\nabla_a Q_1(s,a|\theta_1)|_{s=s_{fi}, a=\pi_f(s_{fi})}\nabla_{\phi_f}\pi_f(s_{fi}|\phi_f)$
20         Update target networks:
21         $\theta_m' \leftarrow \tau\theta_m + (1-\tau)\theta_m'$
22         $\phi_f' \leftarrow \tau\phi_f + (1-\tau)\phi_f'$
23     $s \leftarrow s'$
24 until s reaches terminal state $s_T$;
```

$$r = \begin{cases} 0.4r_1, \text{ if } |s_4| \geq 0.3 \\ 0.4 + 0.4r_2, \text{ if } 0.05 \leq |s_4| < 0.3 \\ 0.8 + 0.2r_3, \text{ if } |s_4| < 0.05, \end{cases} \tag{20}$$

where

$$r_1 = 0.4 \times (1 - \frac{|s_4| - 0.3}{0.7}) + 0.5\ r_{s_5} + 0.1 \times (1 - |s_1|), \tag{21}$$

r_{s_5} is a reward regarding to the error and the change of error of the position, which is expressed as

$$r_{s_5} = \begin{cases} 1, \text{ if } s_4 s_5 < 0 \\ \frac{0.5 - |s_5|}{0.5}, \text{ otherwise}; \end{cases} \tag{22}$$

$$r_2 = 0.4 \times \left(1 - \frac{|s_4|}{0.2}\right) + 0.2 \times \left(1 - \frac{|s_5|}{0.3}\right) + 0.1 \times \left(1 - \frac{|s_2|}{0.3}\right) + 0.2 \times \left(1 - \frac{\sum_{i=0}^{t}|s_4(i)|}{100}\right) + 0.1 \times \left(1 - \frac{\sum_{i=0}^{t}|s_1(i)|}{200}\right) \tag{23}$$

$$r_3 = 0.5 \times \left(1 - \frac{|s_4|}{0.05}\right) + 0.3 \times \left(1 - \frac{\sum_{i=0}^{t}|s_4(i)|}{50}\right) + 0.2 \times \left(1 - \frac{\sum_{i=0}^{t}|s_1(i)|}{100}\right). \tag{24}$$

The reward has three levels which are based on the absolute normalized error of the cart position $|s_4|$. When $|s_4| \geq 0.3$, the cart is considered far away from the origin which is undesirable. The reward is decided by r_1 shown in Eq. (21), r_1 has three terms with the maximum reward restricted to 0.4. The first term is the absolute error of the cart position, the closer the cart is to the origin, the higher reward can be achieved. The second term is the relationship between the error and the change of error for the cart position. When $|s_4|$ is large, the primary goal is set to push the cart back to origin, in which case $s_4 s_5 < 0$, where r_{s_5} is set as 1. Otherwise, it indicates that the error is changing in the undesired direction, where the larger the absolute change of error of the cart position is, the smaller reward r_{s_5} has. The third term is the normalized absolute error of the pole, the closer the pole to the up straight position, the higher reward it has. When $0.05 \leq |s_4| < 0.3$, it is considered the cart-pole system is approaching the balanced position, which receives 0.4 as a base value while having an extra reward r_2 shown in Eq. (23), in this level, reward r changes between 0.4 and 0.8. Reward r_2 has five terms. The first term indicates the cart position, where the smaller absolute error has higher reward. The second term indicates the change error of the cart position, considering the cart-pole system is approaching the balanced point, it is expected the change of error to be small. Similarly, the third term expects the change error of the pole to be small value as well. The fourth and fifth term are related to the accumulated absolute error of the cart and pole position, respectively, which aims to reduce the fluctuations of the cart and the pole. When $|s_4| < 0.05$, it is considered that the system is getting to a relative stable situation, where reward has 0.8 base value with an extra value r_3. In this level, the reward changes between 0.8 and 1.0. r_3 has three terms, which have similar meanings described in r_2. It expects the cart continuing going back to origin while minimizing the fluctuations of the cart and the pole during the whole process.

4. Simulation results

4.1. Control objectives

Different from most of the tests done for the reinforcement learning algorithms on cart-pole system, more requirements are added from the controlling perspective in this research, which can be reflected by the transient response curves of both cart and pole. The transient response curves are expected to have less oscillations, overshoot, undershoot and steady state error. These requirements can be achieved by adopting neuro-fuzzy PID controller as

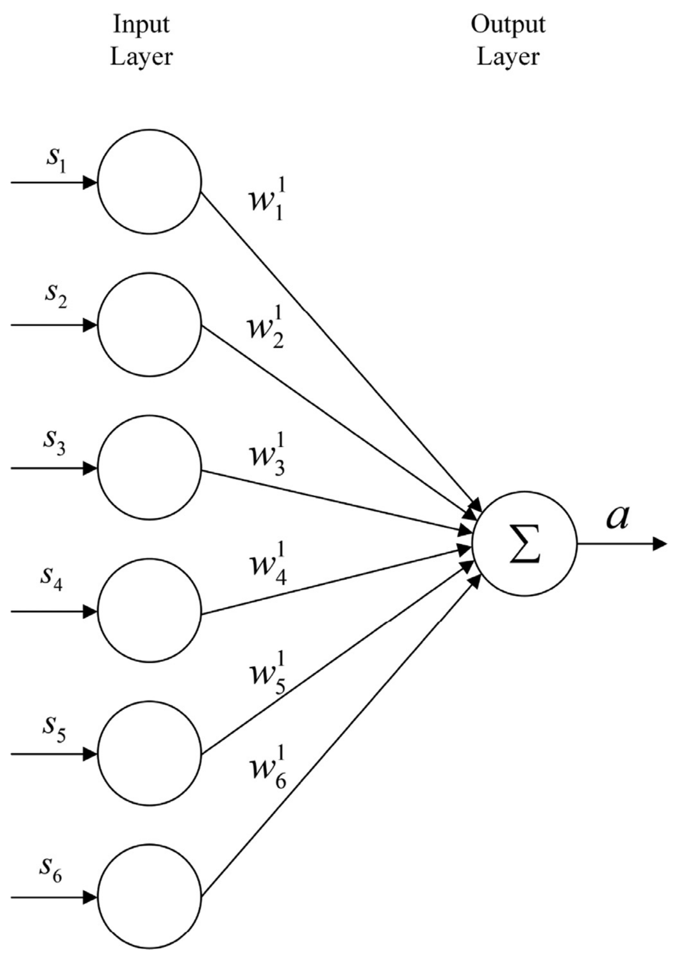


Fig. 9. Structure of linear PID controller in the actor network.

actor NN and designing more comprehensive reward scheme. Besides, as to test the generalization of the proposed controller, a random initial position test from a large operating scope ([−60deg, +60deg]) will be implemented. Furthermore, the trained controller will be tested with different mass of the pole or the cart as well as a low-frequency external disturbance in the beginning period of controlling process as to demonstrate the robustness of the proposed method.

4.2. Linear PID controller

As for comparison, a linear PID controller tuned with TD3 algorithm is designed. Similar to the proposed approach, the actor network is replaced with a neuro-PID controller of which structure presented in Fig. 9. The output of the neural network is

$$a = \sum_{i=1}^{6} s_i w_i^1. \tag{25}$$

The force applied to the cart-pole system is

$$u = Ka, \tag{26}$$

K is the scale factor, $K = 1500$.

Remark 2. The linear PID controller can be regarded as a special case of neuro-fuzzy PID controller where the number of fuzzy rule is reduced to 1.

Table 5
Parameters Setting in TD3.

Variable	Value
max episodes	3000
steps per episode	1000
d	3
K	64
buffer size	10^6
τ	0.001
γ	0.99

Table 6
Parameters Setting for the Cart-pole System.

Variable	Value
M	1.3282 kg
m	0.22 kg
l	0.304 m
F_0	22.915 N
F_1	0.007056 N

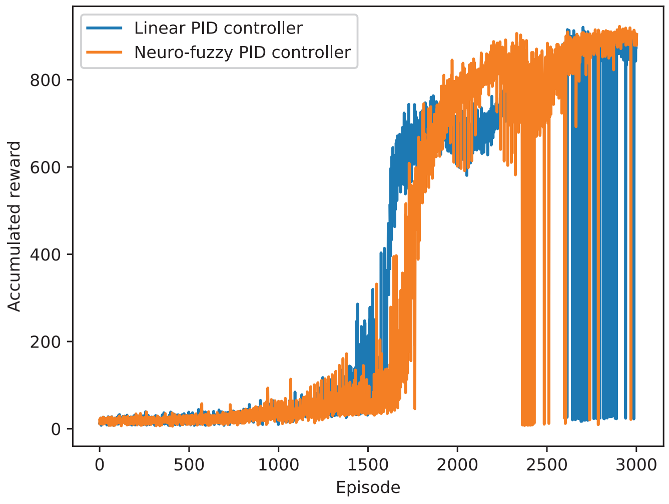


Fig. 10. Comparison of learning curves.

4.3. Parameter settings

As for the purpose of the comparison, both neuro-fuzzy PID controller and linear PID controller are trained with the same set of parameters during the training process except for the learning rate of critic and actor network, which are chosen differently to achieve the best learning performance of the controller itself. The settings of parameters are shown in Table 5. For the neuro-fuzzy PID controller, the learning rates of the actor and critic network are 2.2×10^{-6} and 6.5×10^{-5}, respectively, while for the linear PID controller, the learning rates of the actor and critic network are 8×10^{-6} and 5×10^{-5}, respectively. The critic networks for both neuro-fuzzy PID controller and linear PID controller have 150 neurons in the first hidden layer, 100 neurons in the second hidden layer and one neuron in the output layer. Besides, the parameters of cart-pole system are given in Table 6.

4.4. Training results

The learning curves of neuro-fuzzy PID controller and linear PID controller are shown in Fig. 10. The learning curves present the accumulated reward in each episode instead of the averaged values over episodes.

The transient response curves of neuro-fuzzy PID controller and linear PID controller are shown in Fig. 11 and the forces are shown

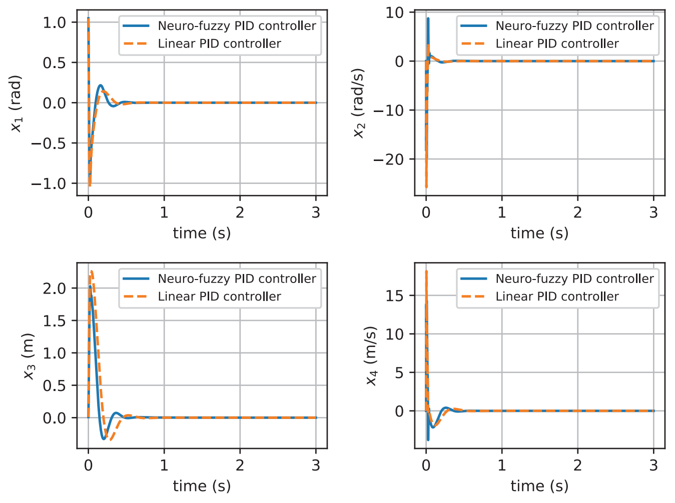


Fig. 11. Comparison of transient response curves.

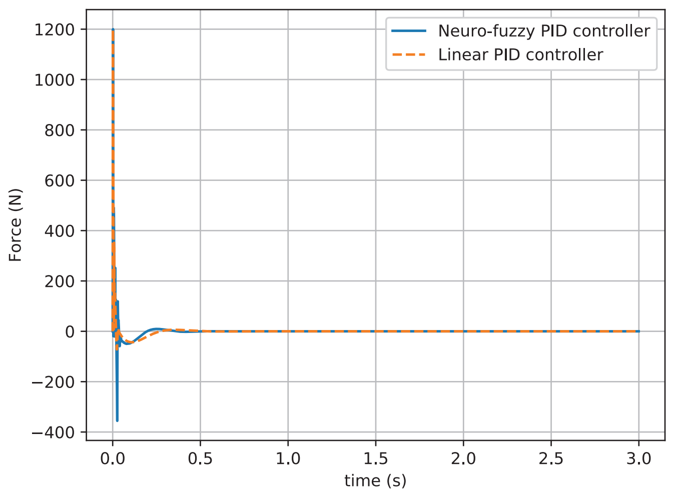


Fig. 12. Comparison of forces.

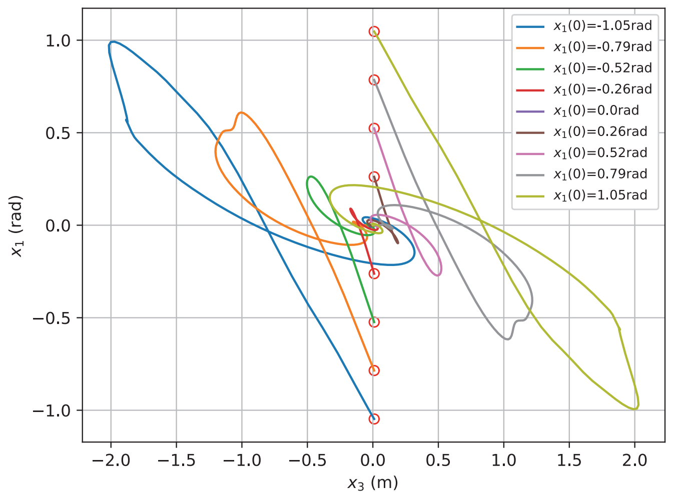


Fig. 13. Phase portrait x_3 and x_1. The curves in different colours represent the phase flows. The red circles indicate the initial states of the trajectories. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

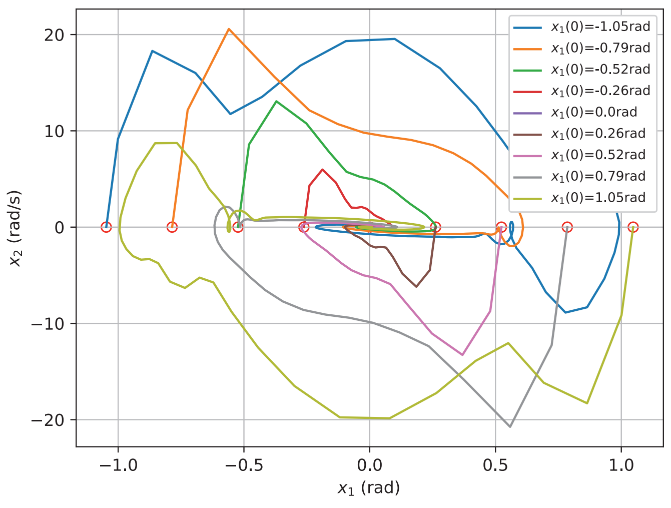


Fig. 14. Phase portrait of x_1 and x_2 from different initial positions. The curves in different colours represent the phase flows. The red circles indicate the initial states of the trajectories. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

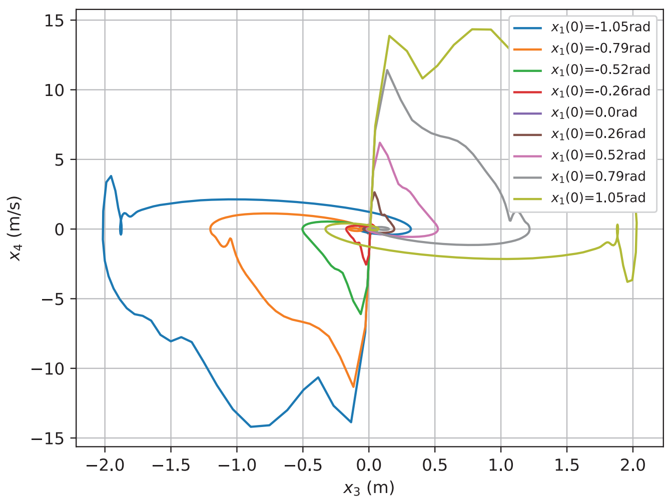


Fig. 15. Phase portrait of x_3 and x_4 from different initial positions. The curves in different colours represent the phase flows. The red circles indicate the initial states of the trajectories. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

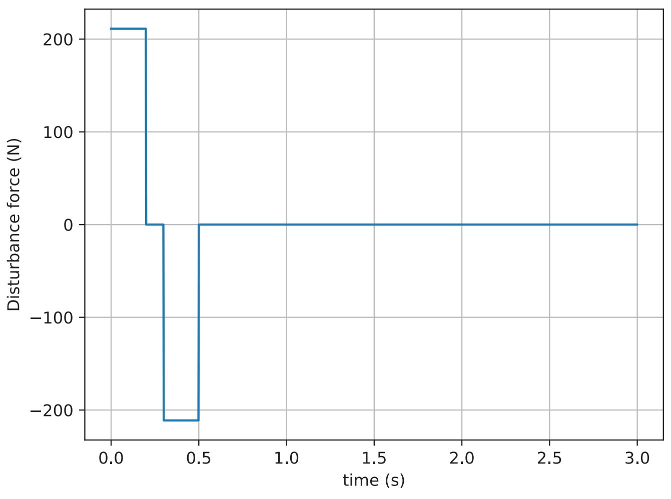


Fig. 16. Disturbance force.

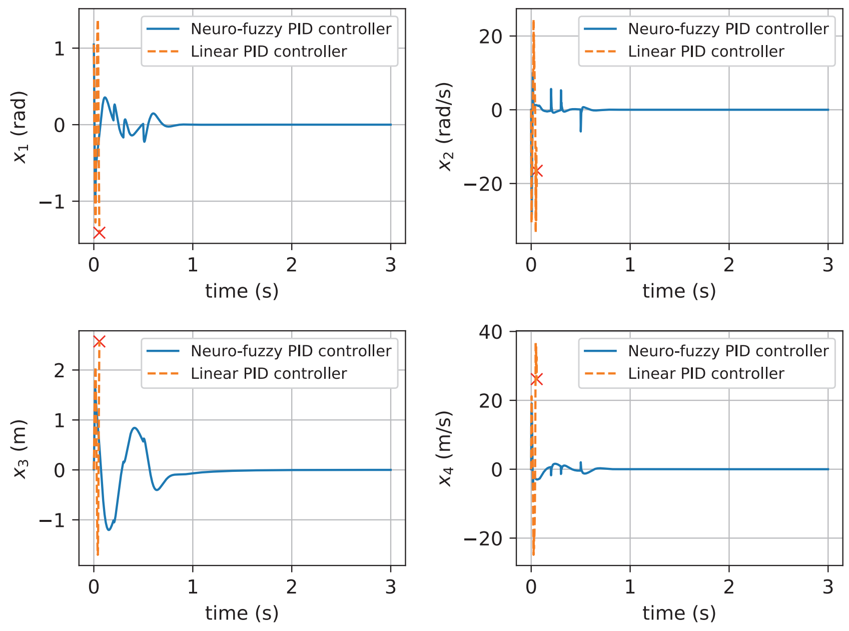


Fig. 17. Comparison of transient response curves with disturbance.

in Fig. 12. It can be observed that neuro-fuzzy PID controller has shorter settling time with less overshoot and undershoot.

4.4.1. Random initial positions test

The training process is done from a fixed state where θ=60°, $x = 0.01$m with the velocity of cart and pole are zero. In the random initial positions test, the angle of the pole is selected from the range of $[-60^\circ, 60^\circ]$ with a sample step of 15° while the initial position remains unchanged as 0.01 m, the velocity for both pole and cart are zero. The phase portraits of neuro-fuzzy PID controller are shown from Figs. 13 to 15.

4.4.2. Robustness test

Two different experiments are designed as to test the robustness of the proposed controller. In the first test, both controllers are tested by changing the mass of the pole or the cart with the fixed initial condition, where θ=60°, $x = 0.01$m with the velocity of cart and pole are zero. The mass of the pole changes with ± 90% and the mass of the cart changes with ± 50%. The neuro-fuzzy PID controller is able to successfully balance the system in all situations while neuro-linear PID controller fails when the mass of the pole is reduced greater than 88%. In the second test, a low-frequency external force is added to control force u as to imitate the disturbance in real environment, like bumping into obstacles or the disturbance of the wind. The disturbance force is shown in Fig. 16, where the disturbance is mainly added in the beginning period of controlling process as to increase the difficulty of balancing the cart-pole system. When comparing the response curves of both controllers shown in Fig. 17, it can be seen that the neuro-fuzzy PID controller is able to balance the system back to origin while linear PID controller fails in the beginning of the control process (the break point is marked with red cross in Fig. 17). According to results of robustness tests, the proposed neuro-fuzzy PID controller shows obvious priority of robustness both in tolerating the change of system parameters and defending external disturbance.

5. Conclusion

In this research, an adaptive neuro-fuzzy PID controller based on TD3 algorithm is proposed. The specially designed fuzzy PID controller acts as the actor role in the TD3 algorithm to automatically tune the parameters for multiple PID controllers while embedding fuzzy system information at the same time. The proposed controller is tested on the benchmark cart-pole problem with a neuro-linear PID controller provided as comparison with both the ability of generalization and robustness tested. According to the simulation results, the generalization and the robustness of tolerating the change of system parameters and external disturbance of the proposed controller is proved.

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.

CRediT authorship contribution statement

Qian Shi: Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Hak-Keung Lam:** Supervision, Conceptualization, Methodology, Writing - review & editing. **Chengbin Xuan:** Writing - review & editing. **Ming Chen:** Writing - review & editing.

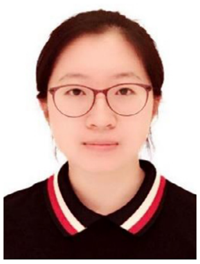
Acknowledgment

The work described in this paper was partly supported by King's College London and Chinese Scholarship Council.

References

[1] A.R.N. Ravari, H.D. Taghirad, A novel hybrid fuzzy-PID controller for tracking control of robot manipulators, in: 2008 IEEE International Conference on Robotics and Biomimetics, IEEE, 2009, pp. 1625–1630.
[2] M.N. Howell, M.C. Best, On-line PID tuning for engine idle-speed control using continuous action reinforcement learning automata, Control Eng. Pract. 8 (2) (2000) 147–154.
[3] K.J. Åström, T. Hägglund, PID Controllers: Theory, Design, and Tuning, 2, Instrument society of America Research Triangle Park, NC, 1995.
[4] H.B. Duan, D.B. Wang, X.F. Yu, Novel approach to nonlinear PID parameter optimization using ant colony optimization algorithm, J. Bionic Eng. 3 (2) (2006) 73–78.

[5] H.A. Varol, Z. Bingul, A new PID tuning technique using ant algorithm, in: Proceedings of the 2004 American Control Conference, 3, 2004, pp. 2154–2159 vol.3, doi:10.23919/ACC.2004.1383780.
[6] H. Boubertakh, M. Tadjine, P.Y. Glorennec, S. Labiod, Tuning fuzzy PD and PI controllers using reinforcement learning, ISA Trans. 49 (4) (2010) 543–551.
[7] V.T. Aghaei, A. Onat, I. Eksin, M. Guzelkaya, Fuzzy PID controller design using Q-learning algorithm with a manipulated reward function, in: 2015 European Control Conference (ECC), IEEE, 2015, pp. 2502–2507.
[8] X. Wen, J. Wang, Fuzzy PID controller based on improved neural network for satellite attitude, in: 2015 Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC), IEEE, 2015, pp. 1206–1211.
[9] M. Rahman, S.M.H. Rashid, M.M. Hossain, Implementation of Q learning and deep Q network for controlling a self balancing robot model, Robot. Biomim. (2018) 4–9, doi:10.1186/s40638-018-0091-9.
[10] M. Chen, H.K. Lam, Q. Shi, B. Xiao, Reinforcement learning-based control of nonlinear systems using lyapunov stability concept and fuzzy reward scheme, IEEE Transactions on Circuits and Systems II: Express Briefs (2019), doi:10.1109/TCSII.2019.2947682. 1–1
[11] I. Carlucho, M.D. Paula, S. Wang, Y. Petillot, G.G. Acosta, Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning, Robot. Auton. Syst. 107 (2018) 71–86, doi:10.1016/j.robot.2018.05.016.
[12] C. Liu, H. Zhang, G. Xiao, S. Sun, Integral reinforcement learning based decentralized optimal tracking control of unknown nonlinear large-scale interconnected systems with constrained-input, Neurocomputing 323 (2019) 1–11.
[13] H. Xiong, T. Ma, L. Zhang, X. Diao, Comparison of end-to-end and hybrid deep reinforcement learning strategies for controlling cable-driven parallel robots, Neurocomputing 377 (2020) 73–84.
[14] F. Li, Q. Jiang, S. Zhang, M. Wei, R. Song, Robot skill acquisition in assembly process using deep reinforcement learning, Neurocomputing 345 (2019) 92–102.
[15] C.J. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (3–4) (1992) 279–292.
[16] A. Younesi, H.C.A. Shayeghi, Q-learning based supervisory PID controller for damping frequency oscillations in a hybrid mini / micro-grid 15(1) (2019) 126–141.
[17] A.e. Hakim, H. Hindersah, E. Rijanto, Application of reinforcement learning on self-tuning PID controller for soccer robot multi-agent system, in: 2013 Joint International Conference on Rural Information Communication Technology and Electric-Vehicle Technology (rICT ICeV-T), 2013, pp. 1–6, doi:10.1109/rICT-ICeVT.2013.6741546.
[18] Q. Shi, H.K. Lam, B. Xiao, S.H. Tsai, Adaptive PID controller based on Q-learning algorithm, CAAI Trans. Intell. Technol. 3 (4) (2018) 235–244.
[19] I. Carlucho, M. De Paula, S.A. Villar, G.G. Acosta, Incremental Q-learning strategy for adaptive PID control of mobile robots, Expert Syst. Appl. 80 (2017) 183–199.
[20] M. Esmaeili, H. Shayeghi, H. Mohammad Nejad, A. Younesi, Reinforcement learning based PID controller design for LFC in a microgrid, COMPEL 36 (4) (2017) 1287–1297.
[21] P.Y. Glorennec, L. Jouffe, Fuzzy Q-learning, in: Proceedings of 6th International Fuzzy Systems Conference, 2, IEEE, 1997, pp. 659–662.
[22] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, 1, MIT press, Cambridge, 1998.
[23] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv:1509.02971 (2015).
[24] L. Jouffe, Fuzzy inference system learning by reinforcement methods, IEEE Trans. Syst. Man Cybern.Part C 28 (3) (1998) 338–355.
[25] M.J. Khodaei, M.H.B. Inaloo, A. Mehrvarz, N. Jalili, An adaptive neuro-fuzzy strategy in closed-loop control of anesthesia, arXiv:1910.06325 (2019).
[26] M. Sedighizadeh, A. Rezazadeh, Adaptive PID controller based on reinforcement learning for wind turbine control, in: Proceedings of World Academy of Science, Engineering and Technology, 27, 2008, pp. 257–262.
[27] H. Wu, S. Song, K. You, C. Wu, Depth control of model-free AUVS via reinforcement learning, IEEE Trans. Syst. Man Cybern. 49 (12) (2018) 2499–2510.
[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, arXiv:1312.5602 (2013).
[29] M. Gheisarnejad, J. Boudjadar, M.H. Khooban, A new adaptive type-II fuzzy-based deep reinforcement learning control: fuel cell air-feed sensors control, IEEE Sens. J. 19 (20) (2019) 9081–9089.
[30] S. Fujimoto, H. van Hoof, D. Meger, Addressing function approximation error in actor-critic methods, arXiv:1802.09477 (2018).
[31] H.K. Lam, H.F.H. Leung, Fuzzy controller with stability and performance rules for nonlinear systems, Fuzzy Sets Syst. 158 (2) (2007) 147–163.



Qian Shi received the B.Eng. degree from the Department of Mechatronic Manufacturing, Harbin Institute of Technology, China, in 2016 and the MSc (Dist.) degree from Faculty of Engineering and Design, University of Bath, United Kingdom, in 2017. She is currently a Ph.D. student in Department of Engineering, King's College London, United Kingdom. Her research interests include reinforcement learning, artificial intelligence and intelligent control.

Hak-Keung Lam received the B.Eng. (Hons.) and Ph.D. degrees from the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong, in 1995 and 2000, respectively. During the period of 2000 and 2005, he worked with the Department of Electronic and Information Engineering at The Hong Kong Polytechnic University as Post-Doctoral Fellow and Research Fellow respectively. He joined as a Lecturer at King's College London in 2005 and is currently a Reader. His current research interests include intelligent control, computational intelligence and machine learning. He has served as a program committee member, international advisory board member, invited session chair and publication chair for various international conferences and a reviewer for various books, international journals and international conferences. He is an associate editor for IEEE Transactions on Fuzzy Systems, IEEE Transactions on Circuits and Systems II: Express Briefs, IET Control Theory and Applications, International Journal of Fuzzy Systems, Neurocomputing and Nonlinear Dynamics; and guest editor for a number of international journals. He is on the editorial board of Journal of Intelligent Learning Systems and Applications, Journal of Applied Mathematics, Mathematical Problems in Engineering, Modelling and Simulation in Engineering, Annual Review of Chaos Theory, Bifurcations and Dynamical System, The Open Cybernetics and Systemics Journal, Cogent Engineering and International Journal of Sensors, Wireless Communications and Control. He is a coeditor of two edited volumes: Control of Chaotic Nonlinear Circuits (World Scientific, 2009) and Computational Intelligence and Its Applications (World Scientific, 2012), and author/coauthor of three monographs: Stability Analysis of Fuzzy-Model-Based Control Systems (Springer, 2011), Polynomial Fuzzy Model Based Control Systems (Springer, 2016) and Analysis and Synthesis for Interval Type-2 Fuzzy-Model-Based Systems (Springer, 2016). He is an IEEE Fellow.

Chengbin Xuan received the B.S. degree in aircraft designing and engineering and the M.S. degree in mechanical engineering from Xi'an Jiaotong University, China, in 2016 and 2019. He is currently a candidate for the Ph.D. degree in the Department of Engineering at the King's College London. His research interests include fuzzy control, reinforcement learning and vehicle automation.

Ming Chen received B.S. degree in Mechanical Engineering from Huaihai Institute of Technology (Jiangsu Ocean University now), Lianyungang, China, in 2015, and M.S. degree in Electronics and Information Engineering from Chonbuk National University, Jeonju, Korea, in 2018. He is currently pursuing the Ph.D. degree in Robotics at King's College London, London, U.K. His research interests are in the areas of model predictive control, reinforcement learning and fuzzy-model-based control systems.