SCIENCE CHINA
Information Sciences

• **Supplementary File** •

Intelligent secure near-field communication

Nan ZHANG[1], Jifa ZHANG[1], Chengwen XING[2], Na DENG[1] & Nan ZHAO[1*]

[1] *School of Information and Communication Engineering, Dalian University of Technology, Dalian* 116024*, China;*
[2] *School of Information and Electronics, Beijing Institute of Technology, Beijing* 100081*, China*

Appendix A Near-field communication system

The wavefronts of EM waves in the near-field region are spherical waves, so we focus on the Euclidean distance between the user and the base station (BS) in the three-dimensional space. To simplify the model, we set the positions of BS and user in the same plane [1]. The near-field system is shown in Figure A1. Assume that the transmit antenna array at BS is a uniform linear array (ULA) with an antenna spacing of d. Without loss of generality, we position the origin of the coordinate system at the center of the ULA. Consequently, the coordinates of nth antenna of the ULA is denoted by $\boldsymbol{c}_n = [nd, 0]^T$, where $n \in \{-\frac{N_t-1}{2}, \ldots, \frac{N_t-1}{2}\}$. Taking the kth user as an example, the distance and the angle from it to the origin is denoted as r_k and θ_k, respectively. Thus, the coordinates of this user can be represented as $\boldsymbol{r}_k = [r_k \cos\theta_k, r_k \sin\theta_k]^T$. Subsequently, the distance from the nth antenna element to the user k can be given by

$$\tilde{r}_{k,n}(r_k, \theta_k, n) = \|\boldsymbol{r}_k - \boldsymbol{c}_n\|_2 = \sqrt{r_k^2 + n^2 d^2 - 2 r_k n d \cos\theta_k}. \tag{A1}$$

Furthermore, within the near-field Fresnel region, i.e., $1.2D \leqslant r \leqslant \frac{2D^2}{\lambda}$, where $D = (N_t - 1)d$ denotes the aperture of antenna, the channel gain of all links to the kth user can be computed as the free-space pathloss of the central link, which is given by $\tilde{\beta}_k = \frac{\sqrt{\rho_0}}{r_k}$, where $\rho_0 = \frac{\lambda}{4\pi}$ is the free-space pathloss at the reference distance 1 m [1].

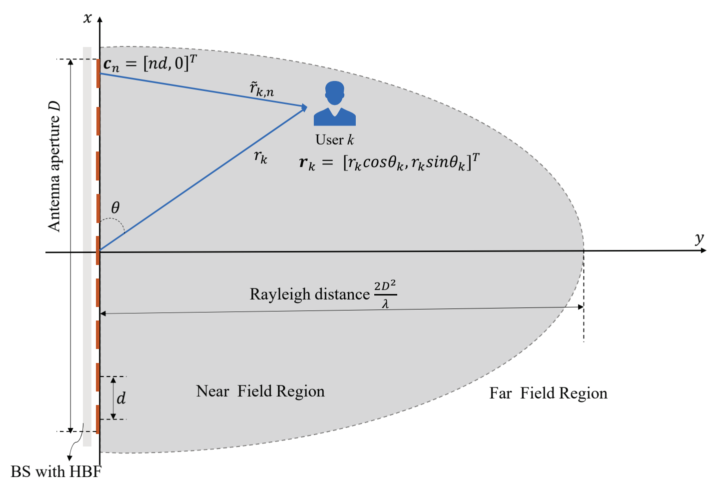

Figure A1 Near-field communication system.

Appendix B DRL for optimization problem

The high dimensionality of the optimal problem makes the traditional optimization techniques intractable. Fortunately, deep reinforcement learning (DRL) technology has stronger robustness to system uncertainty and low dependence on complex mathematical formulas [2], which can significantly reduce the complexity of the algorithm. To solve this problem, we resort to the DRL-based algorithm, which is particularly beneficial to solve time-varying wireless communication systems. In reinforcement learning (RL), the agent interacts with the environment through trial and error to find the optimal policy. In each interaction, the agent observes the current state of environment, and selects the optimal action by policy network, which affects the environment. After receiving the actions of the agent, the environment returns the corresponding instant reward and changes a new state. The agent observes a new state in the next time step, and so on. DRL is the combination of RL and deep learning (DL), uses deep neural networks (DNNs) to learn the mapping relationship between complex state spaces and action spaces [3]. In our system, base station (BS)

* Corresponding author (email: zhaonan@dlut.edu.cn)

stands for the agent, and the secure near-field communication network can be viewed as the environment. The key elements of DRL are defined as follows:

Action: Since the hybrid beamfoming (HBF) matrix and the position of BS are jointly optimized to maximize the sum secrecy rate, the action should include positions and HBF matrix, which in the lth time step can be defined as $a_l = [\boldsymbol{W}_l, \boldsymbol{F}_l, \boldsymbol{p}_l]$, whose dimension is $2N_{\mathrm{t}}N_{\mathrm{rf}} + 2N_{\mathrm{rf}}K + 2$.

State: The state determines the actions of the agent, so the state should be related to channels information and HBF matrix,. Consequently, the state in the lth time step is given by $s_l = [\boldsymbol{h}_{k,l}, \boldsymbol{h}_{e,l}, \boldsymbol{h}_{k,l}^{\mathrm{H}}\boldsymbol{F}_l\boldsymbol{w}_{k,l}, \boldsymbol{h}_{e,l}^{\mathrm{H}}\boldsymbol{F}_l\boldsymbol{w}_{k,l}, \boldsymbol{W}_l, \boldsymbol{F}_l, \boldsymbol{p}_l]$. Since the input of DNNs should be real-values. We feed the real and imaginary parts of $\boldsymbol{F}_l$ and $\boldsymbol{W}_l$ into them, respectively. The dimension of state is $2N_{\mathrm{t}}K + 2N_{\mathrm{t}} + 4K + 2N_{\mathrm{t}}N_{\mathrm{rf}} + 2N_{\mathrm{rf}}K + 2$.

Reward: The environment returns the instant reward to evaluate the performance of the action selected by the policy network under the current state [4]. Our objective is to maximize sum secrecy rate. Therefore, the secrecy rate can be employed as the reward. Meanwhile, we move the quality of service (QoS) constraint and the positions of BS constraint into the reward function as the penalty term. The objective of DRL-based algorithm is to find a policy network that can maximize the mathematical expectation of cumulative discount reward. Thus, the optimization problem corresponding to the DRL can be expressed as [5]

$$\max_{\vartheta} \mathcal{J}(\vartheta) = \max_{\boldsymbol{\mu}_\vartheta} \mathbb{E}_\pi \left[\sum_{l=1}^{L} \gamma^{l-1} r(\boldsymbol{s}_l, \boldsymbol{a}_l) \right] \tag{B1}$$

where $\boldsymbol{\mu}_\vartheta$ denotes the deterministic policy network with the parameter ϑ, $\gamma \in [0,1]$ represents the discount factor for reward, and the $\mathbb{E}(\cdot)$ expresses the mathematical expectation, $\mathcal{J}(\vartheta)$ represents the objective function, which expresses the mathematical expectation of cumulative discount reward under the policy network $\boldsymbol{\mu}_\vartheta$.

In our work, we choose the DDPG-based algorithm as a benchmark. But the DDPG algorithm still has problems such as overestimation. In order to solve the problems, the TD3 made several improvements. First, the TD3 algorithm adds noise to the target policy network during training to prevent the policy network from exploiting Q-value errors, obtain more robust and stable performance. Second, the TD3 algorithm updates the value networks every time step but updates the policy network and the three target networks less frequently, which prevents overfitting to the current Q-function estimates. What's more, TD3 algorithm uses the minimum of the two Q-functions, known as double Q-learning, to reduce the overestimation bias further. These improvements collectively enhance the stability and performance of the TD3 algorithm compared to its predecessor, DDPG, which makes the TD3 algorithm perform better than the DDPG algorithm in general cases. Consequently, in this letter, a TD3-based algorithm is proposed to solve the dynamic continuous changing problem, where the agent gradually learns a deterministic policy by the trial-and-error interaction to select the optimal action.

The TD3-based algorithm consists of six DNNs, which are two value networks, one policy network, and their corresponding target networks. The value networks and policy network have the same structure as their corresponding target network. All DNNs have four layers, and all layers are fully connected. As shown in Figure B1, the value network consists of an input layer and three hidden layers, which use *ReLU* as the activation function,and take the Q-value as an output. The policy network includes an input layer, two hidden layers and an output layer, with the input and hidden layers using *ReLU* as the activation function and the output layer using *tanh* as the activation function.

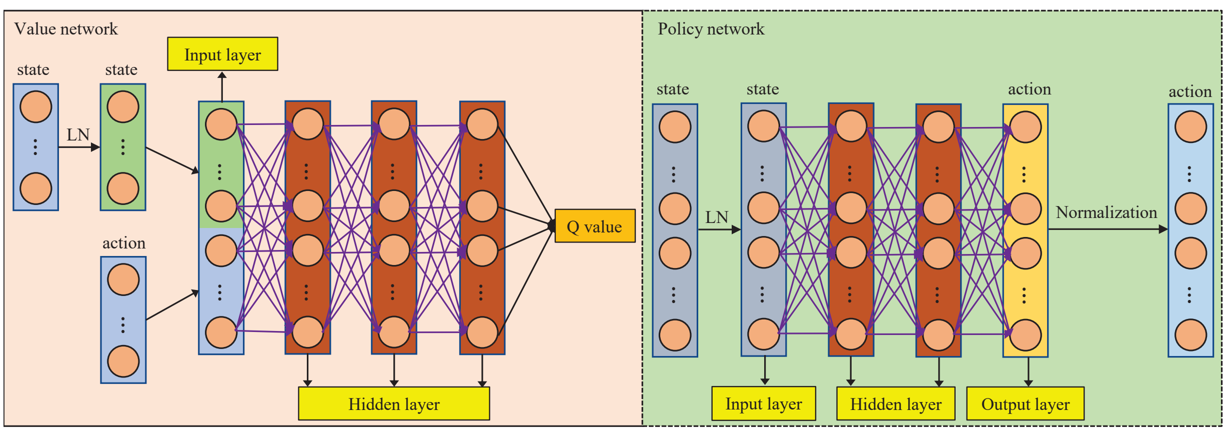

Figure B1 The structure of DNNs.

We describe the policy network along with target network and the value networks along with target networks respectively as follows [6].

Policy network: The policy network of TD3 is a deterministic policy network, $\boldsymbol{\mu}_\vartheta$ with the learnable parameter ϑ. The policy network takes the s_l as input, and outputs the action a_l. In the TD3-based algorithm, a target policy network is also employed to alleviate the overestimation problem. Similarly, the parameter of the target policy network is $\boldsymbol{\mu}'_\vartheta$ with the parameter ϑ'.

Value network: The value network takes both the state and action as input and output Q-value to evaluate the performance of taking action a_l under state s_l. The TD3-based algorithm employs two value networks, $q_{\omega_1}$and q_{ω_2}, with parameters $\omega_1$and ω_2, respectively, along with their corresponding target networks $q_{\omega'_1}$ and $q_{\omega'_2}$, with parameters ω'_1 and ω'_2 to alleviate the overestimation problem.

To improve the performance of algorithm, the TD3-based algorithm adds clipped expiration noise $\xi \sim \mathcal{CN}(0, \sigma^2, -c, c)$ to the action computed by target policy network ϑ', which is denoted by $\hat{a}'_{l+1}$. Furthermore, TD3-based algorithm updates the value networks every time step but updates the policy network and the three target networks every m time steps, where m is a tunable hyperparameter [7].

As shown in Figure B2, at the initial stage, the parameters of the policy network and value networks ϑ, $\omega_1$and ω_2 are randomly generated, and the parameters of the target networks are initialized as $\vartheta = \vartheta'$, $\omega_1 = \omega'_1$ and $\omega_2 = \omega'_2$. (s_l, a_l, r_l, s_{l+1}) is put into the replay buffer $\mathcal{E}$ whose size is $E_{\rm B}$ as an experience replay tuple during the policy and value networks working. Then the tuple is randomly sampled a mini-batch with size $N_{\rm E}$ from the buffer to update the parameters until the number of tuples in buffer E is bigger than $N_{\rm E}$.

The target policy network predicts the action and adds the clipped noise to it as

$$\hat{a}'_{l+1} = \boldsymbol{\mu}_{\vartheta'_{\rm now}}(s_{l+1}) + \xi, \tag{B2}$$

where $\vartheta'_{\rm now}$ denotes the current parameter of target policy network, and $\xi \sim \mathcal{CN}(0, \sigma^2, -c, c)$. Then, the TD target is computed by

$$\hat{y}_l = r_l + \gamma q_{\omega'}(s_{l+1}, \hat{a}'_{l+1}), \tag{B3}$$

where $q_{\omega'}(s_{l+1}, \hat{a}'_{l+1}) = \min\{q_{\omega'_{1,\rm now}}(s_{l+1}, \hat{a}'_{l+1}), q_{\omega'_{2,\rm now}}(s_{l+1}, \hat{a}'_{l+1})\}$ represents the minimum prediction result of the two target value networks, and $\omega'_{1,\rm now}$ and $\omega'_{2,\rm now}$ refer to the current parameters of target value networks, respectively. The two value networks also make predictions as below

$$\hat{q}_{1,l} = q_{\omega_{1,\rm now}}(s_l, a_l), \hat{q}_{2,l} = q_{\omega_{2,\rm now}}(s_l, a_l), \tag{B4}$$

where $\omega_{1,\rm now}$ and $\omega_{2,\rm now}$ represent the current parameters of value networks, respectively. Thus, the TD error can be expressed by

$$\delta_{1,l} = \hat{q}_{1,l} - \hat{y}_l, \delta_{2,l} = \hat{q}_{2,l} - \hat{y}_l. \tag{B5}$$

To overcome the overestimation, the loss functions of the value networks can be expressed as

$$L(\omega_1) = \frac{1}{N_{\rm E}}\sum_{l=1}^{N_{\rm E}}(\delta_{1,l})^2, L(\omega_2) = \frac{1}{N_{\rm E}}\sum_{l=1}^{N_{\rm E}}(\delta_{2,l})^2, \tag{B6}$$

where $N_{\rm E}$ denotes the size of mini-batch.

The parameters of value network can be updated by

$$\omega_{1,\rm new} = \omega_{1,\rm now} - \alpha\nabla_{\omega_1}L(\omega_1), \omega_{2,\rm new} = \omega_{2,\rm now} - \alpha\nabla_{\omega_2}L(\omega_2), \tag{B7}$$

where α denotes the learning rate, $\nabla_{\omega_1}L(\omega_1)$ and $\nabla_{\omega_2}L(\omega_2)$ represent the gradients of $L(\omega_1)$ and $L(\omega_2)$ with respect to the parameters ω_1 and ω_2, respectively, and $\omega_{1,\rm new}$ and $\omega_{2,\rm new}$ refer to the new parameters of value network after updating, respectively.

Next, update the parameters of the policy network and three target networks every m time steps.The parameter of policy network can be updated as

$$\vartheta_{\rm new} = \vartheta_{\rm now} + \frac{\beta}{N_{\rm E}}\sum_{l=1}^{N_{\rm E}}\nabla_\vartheta\boldsymbol{\mu}_{\vartheta_{\rm now}}(s_l)\nabla_{a_l}q_{\omega_{1,\rm now}}(s_l, \hat{a}_l), \tag{B8}$$

where $\vartheta_{\rm now}$ and $\vartheta_{\rm new}$ represent the current and new parameters of policy network, β refers to the learning rate of policy network, $\nabla_\vartheta\boldsymbol{\mu}_{\vartheta_{\rm now}}$ denotes the gradient of $\boldsymbol{\mu}_{\vartheta_{\rm now}}$ with respect to the parameter ϑ. Similarly, $\nabla_{a_l}q_{\omega_{1,\rm now}}$ expresses the gradient of $q_{\omega_{1,\rm now}}$ with respect to a_l, $\hat{a}_l$ refers to the prediction result of the policy network in the given state s_l.

The parameters of three target networks can be updated [8] by

$$\vartheta'_{\rm new} = \tau\vartheta_{\rm new} + (1-\tau)\vartheta'_{\rm now}, \tag{B9}$$

$$\omega'_{1,\rm new} = \tau\omega_{1,\rm new} + (1-\tau)\omega'_{1,\rm now}, \tag{B10}$$

$$\omega'_{2,\rm new} = \tau\omega_{2,\rm new} + (1-\tau)\omega'_{2,\rm now}. \tag{B11}$$

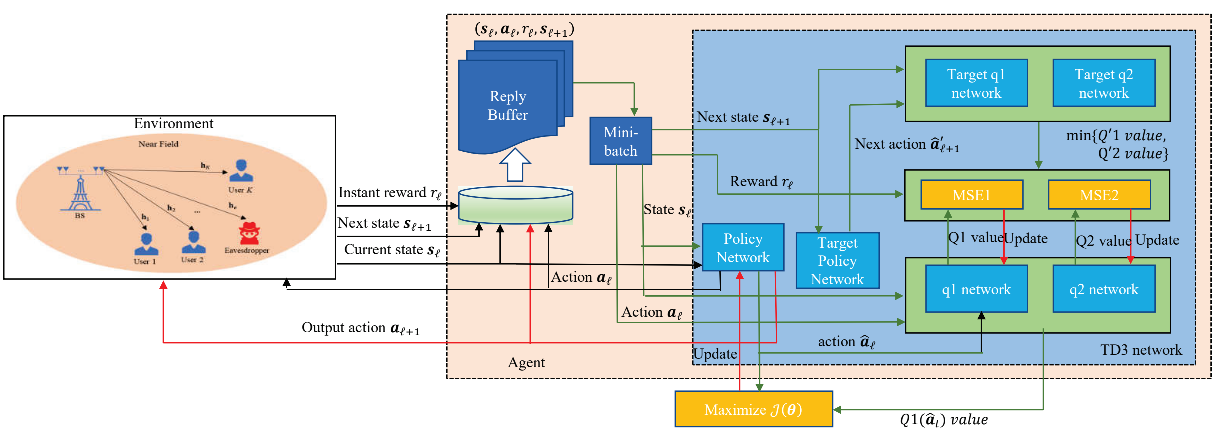

Figure B2 The agent and environment.

The training process of the TD3-based algorithm is shown in Algorithm B1. Before the algorithm starts, the environment parameters are set as shown in Table B1. At the beginning of Algorithm B1, the position of BS and hybrid beamforming matrix are randomly generated, and then the channels $h_{k,l}, h_{e,l}$ are generated. Algorithm B1 will execute over T episodes, and each episode consists of L time steps. At the beginning of each episode, the NFC environment is reset, and the digital beamforming matrix is restricted during the reset environment to satisfy the transmit power constraint. In each episode, the interactions between the agent and the environment generate the experience tuples which are stored in the experience replay buffer. The DNNs' training starts after accumulating a certain amount of tuples. Then the optimal action, which is denoted as the action corresponding to the maximal cumulative discount reward can be obtained through the updates of DNNs. To improve the performance of the TD3-based algorithm, the clipped noise is added to the actions predicted by the target policy network to smooth the process and improve the exploration ability as shown in Algorithm B1. What's more, TD3-based algorithm is robust to the different initial points. To verify this, we plot the average reward convergence curves of different initial points by changing the random seeds, and the result is shown in Figure B3. It can be seen that the proposed TD3-based algorithm can converge to a satisfied result under different initial points, which is also one of the advantages of DRL-based algorithm over traditional optimization techniques.

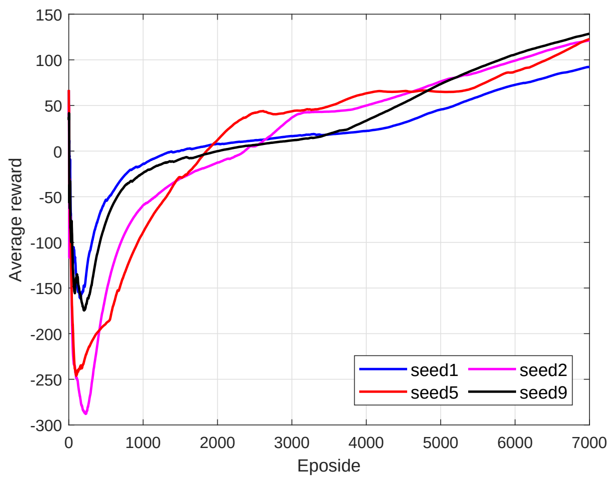

Figure B3 The average reward convergence curves of different random seeds.

Table B1 Environment parameters in Algorithm B1

Parameter	Description	Value
N_{t}	number of antennas in the BS	105
N_{rf}	number of RF chains	10
K	number of users	4
L	number of time steps	50
D	antenna aperture	0.5 m
κ	Rician factor	10
λ	waveform length	0.01 m
$P_{\max}$	maximal transmit power	40dBm
$AWGN$	power of additive white Gaussian noise	1×10^{-6} W
$\mathfrak{r}_k$	rate threshold	0.1 bps/Hz

Furthermore, the update frequency of the value network is faster than that of the policy network and the three target networks as show in lines 16-18 of Algorithm B1, which denotes updating the value network once per time step, while updating the policy network and the three target networks every m time steps, with the aim to train more reliable value networks so that we can obtain more stable results. The hyper-parameters need to be continuously adjusted based on the performance of the convergence curve. The values of hyper-parameters proposed in Algorithm B1 are shown in Table B2. We choose the DDPG-based algorithm as a benchmark, and the values of hyper-parameters of the benchmark are shown in Table B3

References

1 Wang Z, Mu X, Liu Y. Near-Field Integrated Sensing and Communications. IEEE Communications Letters, 2023, 27(8): 2048-2052.

2 Peng Z, Zhang Z, Kong L, et al. Deep Reinforcement Learning for RIS-Aided Multiuser Full-Duplex Secure Communications With Hardware Impairments. IEEE Internet of Things Journal, 2022, 9(21): 21121-21135.

3 Zhong C, Cui M, Zhang G, et al. Deep Reinforcement Learning-Based Optimization for IRS-Assisted Cognitive Radio Systems. IEEE Transactions on Communications, 2022, 70(6): 3849-3864.

4 Saleem R, Ni W, Ikram M, Jamalipour A. Deep-Reinforcement-Learning-Driven Secrecy Design for Intelligent-Reflecting-Surface-Based 6G-IoT Networks. IEEE Internet of Things Journal, 2023, 10(10): 8812-8824.

5 Yang Z, Liu Y, Chen Y, et al. Deep Reinforcement Learning for RIS-Aided Non-Orthogonal Multiple Access Downlink Networks. In: GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Taipei, Taiwan, 2020. 1-6.

Algorithm B1 TD3-based algorithm for optimization problem

Initialization: Generate the channels $\boldsymbol{h}_{k,l}, \boldsymbol{h}_{e,l}, k = 1, \ldots, K$; Randomly generate the parameters of policy network and value networks $\vartheta, \omega_1, \omega_2$. Set $\vartheta' \leftarrow \vartheta, \omega_1' \leftarrow \omega_1, \omega_2' \leftarrow \omega_2$, which refer to the parameters of target network. Empty the replay buffer $\mathcal{E}$, whose size is E_{B}.
Input: The position of BS, users and eavesdropper, number of transmit antenna. N_{t}, number of users K, antenna aperture D, waveform length λ, Rician factor κ and number of snapshots L.
Output: The optimal action $a_l = [\boldsymbol{W}_l, \boldsymbol{F}_l, \boldsymbol{p}_l]$
for episode $t = 1, 2, \ldots, T$ **do**
    for time step $l = 1, 2, \ldots, L$ **do**
        The agent selects action a_l based on state s_l via policy network.
        The environment generates the next state s_{l+1} and instant reward r_l based on a_l via value network.
        Store the experience replay tuples (a_l, s_l, r_l, s_{l+1}) in replay buffer $\mathcal{E}$.
        if The number of tuples $E \geqslant N_{\mathrm{E}}$ **then**
            Randomly sample a mini-batch with size N_{E} from the buffer.
            Obtain $\hat{a}_{l+1}'$ by B2.
            Compute the TD targets by B3.
            The value networks make predictions via B4.
            Taking B5 as TD error, get the loss functions of both value networks by B6.
            Update the parameters of value networks via B7
            if $l \mod m = 0$ **then**
                Update the parameter of policy network ϑ by B8
                Update the parameters of target networks via B9, B10 and B11
            end if
        end if
    end for
end for

Table B2 Hyper-parameters in Algorithm B1

Parameter	Description	Value
γ	discount factor	0.96
α	learning rate	4×10^{-5}
τ	learning rate in soft/hard updates of the target networks	1×10^{-4}
$decay$	decay rate	2×10^{-5}
N_{E}	size of mini-batch	32
E_{B}	buffer size	20000
m	update frequency	2
T	maximum number of episodes	7000
env	OpenAI Gym environment name	NFC
$seed$	seed number for PyTorch and NumPy	0
ρ	penalty coefficient	0.9

Table B3 Hyper-parameters in DDPG-based algorithm

Parameter	Description	Value
γ	discount factor	0.95
α	learning rate	4×10^{-5}
τ	learning rate in soft/hard updates of the target networks	1×10^{-4}
$decay$	decay rate	3×10^{-5}
N_{E}	size of mini-batch	64
E_{B}	buffer size	20000
m	update frequency	2
T	maximum number of episodes	7000
env	OpenAI Gym environment name	NFC
$seed$	seed number for PyTorch and NumPy	0
ρ	penalty coefficient	0.9

6 Li P, Wang Y, Gao Z. Path Planning of Mobile Robot Based on Improved TD3 Algorithm. 2022 IEEE International Conference on Mechatronics and Automation (ICMA), 2022, 715-720.
7 Egbomwan O E, Liu S, Chaoui H. Twin Delayed Deep Deterministic Policy Gradient (TD3) Based Virtual Inertia Control for Inverter-Interfacing DGs in Microgrids. IEEE Systems Journal, 2023, 17(2): 2122-2132.
8 Shen X, Ma K, Yang M, et al. Variable impedance control method for robot contact force based on TD3 algorithm. 2022 China Automation Congress (CAC), 2022, 2327-2332.