A Secure Robot Learning Framework for Cyber Attack Scheduling and Countermeasure

Wu, Chengwei; Yao, Weiran; Luo, Wensheng; Pan, Wei; Sun, Guanghui; Xie, Hui; Wu, Ligang

DOI
10.1109/TRO.2023.3275875

Publication date
2023

Document Version
Final published version

Published in
IEEE Transactions on Robotics

A Secure Robot Learning Framework for Cyber Attack Scheduling and Countermeasure

Chengwei Wu, *Member, IEEE*, Weiran Yao, *Member, IEEE*, Wensheng Luo, *Member, IEEE*, Wei Pan, Guanghui Sun, Hui Xie, *Senior Member, IEEE*, and Ligang Wu, *Fellow, IEEE*

***Abstract*—The problem of learning-based control for robots has been extensively studied, whereas the security issue under malicious adversaries has not been paid much attention to. Malicious adversaries can invade intelligent devices and communication networks used in robots, causing incidents, achieving illegal objectives, and even injuring people. This article first investigates the problems of optimal false data injection attack scheduling and countermeasure design for car-like robots in the framework of deep reinforcement learning. Using a state-of-the-art deep reinforcement learning approach, an optimal false data injection attack scheme is proposed to deteriorate the tracking performance of a robot, guaranteeing the tradeoff between the attack efficiency and the limited attack energy. Then, an optimal tracking control strategy is learned to mitigate attacks and recover the tracking performance. More importantly, a theoretical stability guarantee of a robot using the learning-based secure control scheme is achieved. Both simulated and real-world experiments are conducted to show the effectiveness of the proposed schemes.**

***Index Terms*—Deep reinforcement learning, optimal attack scheduling, robot, secure control.**

I. Introduction

ROBOTIC systems are playing a more and more important role in daily life and industry, freeing people from trivial and dangerous work and improving efficiency. Along with the increasing development of computers, communication networks, and intelligent sensing devices, modern robots equipped with these devices lead to higher performance and more functionalities, the increase of which, however, exposes robots to adversaries. When robots exchange information between intelligent sensors and electrical control units over open and shared communication networks, malicious attackers can compromise the integrity of robotic systems by executing cyber attacks, deteriorating the performance of robotic systems [1], [2]. Attacks can be executed in different modules, such as the sensor/actuator, communication network, and physical interface. For instance, in [3] Iranian air forces captured an American Lockheed Martin RQ-170 Sentinel unmanned aerial vehicle (UAV) by spoofing the Global Position System (GPS) data in 2011. In [4] attackers trick the yacht's navigation system by spoofing the GPS, putting the yacht off course. Maggi et al. [5] detailed exactly how an industrial robot is attacked by advanced hackers. Additionally, in the movie The Fate of the Furious (2017), there exist clips in which hundreds of cars run amok in the street because they are taken over by adversaries. Although the stability and safety problems of robotic systems have been paid much attention to and various results have been reported [6], [7], [8], [9], their security problems cannot be solved by using previous results. To protect the performance of robots from deterioration, even destruction, the study of designing robotic systems with high assurance is a hot topic, attracting researchers from control and computer science communities. This paper intends to establish a secure robot learning framework for cyber attack scheduling and countermeasure, analyzing how an adversary with limited attack energies constructs an optimal attack scheme to invade robots and proposing a secure control algorithm against adversaries.

A. Related Literature Review

Malicious attacks on robots can be regarded as adversarial actions with the intent of compromising the availability and integrity of transmitted information. There exist dramatic differences between system faults and attacks, making existing fault-tolerant control schemes fail to work in modern robots against malicious attacks. Faults occur without explicit objects, while attacks are camouflaged and designed elaborately [10], [11], [12]. For example, an unmanned aerial vehicle can crash when faults occur. If cyber attacks are executed, adversaries can take over unmanned aerial vehicles. Generally speaking, attacks can be classified into several types, including the denial of service (DoS) attack, the false data injection attack (deception attack), the replay attack, and the zero-dynamics attack [13]. Although the topic of securing robotic systems in the presence of malicious attacks is in its early stage, several remarkable results have been

Manuscript received 23 December 2022; accepted 11 April 2023. Date of publication 5 June 2023; date of current version 4 October 2023. This work is supported in part by the National Natural Science Foundation of China under Grant 62033005, Grant 62203136, Grant 62022030, Grant 62173107, Grant 62106062, and in Part by the Key R&D Program of Heilongjiang Province under Grant 2022ZX01A18. This paper was recommended for publication by Associate Editor G. Huang and Editor W. Burgard upon evaluation of the reviewers' comments. *(Corresponding author: Ligang Wu.)*

Chengwei Wu, Weiran Yao, Wensheng Luo, Guanghui Sun, and Ligang Wu are with the Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, China (e-mail: chengweiwu@hit.edu.cn; yaoweiran@hit.edu.cn; wensheng.luo@hit.edu.cn; guanghuisun@hit.edu.cn; ligangwu@hit.edu.cn).

Wei Pan is with the Department of Computer Science, The University of Manchester, M13 9PL Manchester, U.K., and also with the Department of Cognitive Robotics, Delft University of Technology, 2628 CD Delft, Netherlands (e-mail: wei.pan@tudelft.nl).

Hui Xie is with the State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150001, China (e-mail: xiehui@hit.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2023.3275875.

Digital Object Identifier 10.1109/TRO.2023.3275875

reported in the literature. In [14], it has demonstrated how to capture and control a UAV through spoofing of GPS data, and the necessary conditions to capture a UAV have been established. Considering stealthy attacks on ground vehicle sensors, the authors of [10] proposed a resilient iterative filtering approach to estimate system states by using redundant sensor measurements. With the aid of software and hardware redundancy, effective attack detection and defense framework have been designed for a quadcopter against different types of attacks in [15], in which adversaries do not have access to the proposed scheme. In [16], the nonlinear dynamics of mobile robots have been utilized to construct an attack detection scheme, which can generate alarms promptly when sensor/actuator attacks occur. Such a scheme has also been validated on two types of robots with various attack scenarios. A switching distributed control approach has been given to secure multirobot systems against actuator DoS attacks and sensor deception attacks in [17]. For more related results, we refer readers to [18], [19], [20], and references therein.

Some other alternative approaches to securing modern robots under cyber attacks can refer to schemes for cyber-physical systems because the typical application of cyber-physical systems is a modern robotic system. Several conclusions and elegant secure algorithms have been proposed in the past two decades based on the control-theoretical approach. To name a few, if a defender successfully attempts to recover the system state under sparse cyber attacks, the number of attacked sensors cannot exceed half of the number of sensors [21], [22]. When DoS attacks are described as a stochastic model (Markov model and Bernoulli model), the critical value of the probability of success of the attack, under which the estimate and stability performances can be guaranteed, has been derived in [23]. According to these conclusions, some improved algorithms have been proposed for cyber-physical systems against malicious adversaries, see for example, a switching observer-based estimate scheme in [24], a linear quadratic secure controller in [25], and a learning-based secure tracking control algorithm in [26]. The detection and secure control schemes have been provided for cyber-physical systems against replay attacks in [27], [28]. As much stealthier attacks, false data injection attacks have been widely investigated, for example, attack detection schemes [29], [30], secure state estimate algorithms [31], [32], resilient controllers [33], [34]. From an attacker's perspective, researchers have investigated how to construct effective attack sequences to deteriorate system performance, such as DoS attack scheduling [35], [36] and false data injection attack scheme [37], [38], based on which system defenders can design more effective countermeasures. Furthermore, combined with the game-theoretical approach, defenders and adversaries are described in a unified framework, and saddle-point equilibrium policies have been derived for both sides [39], [40], [41].

Although progress has been made in securing cyber-physical systems, the aforementioned results rely on exact system knowledge, which may not be obtained easily. Reinforcement learning approach provides an alternative way to design systems policies without using such knowledge. Various reinforcement learning algorithms have been proposed [42], [43], [44], [45], [46], [47], [48], based on which an attitude retention scheme has been reported for a robotic fish with sim-to-real transfer [49], quadrupedal locomotion in challenging terrain [50], learning-based control and filtering approaches with stability guarantee [51], [52], [53], [54] have been reported. Considering the complexity and scale of cyber-physical systems, the deep reinforcement learning approach has been applied to solve security problems of cyber-physical systems, including secure control, attack detection, and game-based defending policy design (see a survey [55] for more details). Similarly, adversaries can also utilize the deep reinforcement learning approach to construct attack schemes [56]. However, how to design an optimal attack scheme and optimal countermeasure in the framework of deep reinforcement learning with stability guarantee is not investigated for modern robotic systems under malicious attacks in literature.

B. Contributions of This Article

In the control community, none of the existing methods adequately addresses optimal attack and countermeasure design problems in the presence of strong nonlinearities. The method in [52] focused on typical control problems without attack, while the method in [53] focused on state estimation problems and cannot be applied to the secure control problem. We draw inspiration from [52], [53] to deal with (strong) nonlinearity. However, under attacks, the secure control problem for nonlinear dynamical systems cannot be trivially solved by directly applying the methods in [52] and [53] or their combinations. In this article, we integrate the control method, e.g., controller design based on a nominal model, and the reinforcement learning-based method to investigate the optimal false data injection attack and countermeasure design problems for modern robotic systems. We first provide a nominal controller for a robot without attacks. Such a controller is capable of stabilizing the system. Then, we consider that the adversaries intend to perturb the control signals by injecting false data. A learning frame is settled for a malicious adversary to construct optimal false data attacks, following which a defender is also described in a learning frame. Algorithms are provided to learn optimal false data injection attacks and optimal countermeasures. More importantly, the Lyapunov theory provides strict mathematical proof to show that robotic systems under the learning-based secure scheme are stable. The differences between our paper and existing related results are described in Table I. The main contributions of this article can be summarized as follows.

1) This article is, for the first time, focused on the optimal false data injection attack problem for modern robots described by nonlinear systems. Standing from the adversary's perspective, a learning-based attack algorithm is provided to deteriorate the tracking performance with minimal attack energies. The advantage of such an algorithm is that the robot's kinematic model is not linearized. When attacks occur, the nonlinear model can describe the real dynamics. In contrast, a linear model may fail [38], [57], which further implies that the attack scheme relying on a linear model cannot reveal the attack effect.

TABLE I
COMPARISONS WITH THE EXISTING METHODS

	secure defense	control	estimation	exponential performance	optimal attack	direct defense	physical constraint	real-world experiment
[33]	✓	✓	✗	✗	✗	✗	✗	✗
[52]	✗	✓	✗	✗	✗	✗	✓	✗
[53]	✗	✗	✓	✓	✗	✗	✓	✗
[54]	✗	✗	✓	✗	✗	✗	✓	✓
[57]	✓	✗	✗	✗	✓	✗	✗	✗
This article	✓	✓	✗	✓	✓	✓	✓	✓

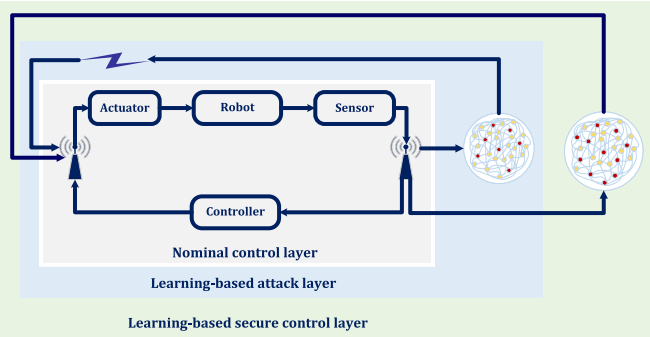


Fig. 1. Secure framework in this article.

2) A learning-based secure countermeasure is proposed to militate the optimal false data injection attacks. More importantly, the stability of robotic systems under the learned policy is strictly proved by combining the Lyapunov theory. The proof shows that deep reinforcement learning is not merely an empirical approach and that the stability guarantee can be achieved.
3) Compared with our previous work [52], [53], [54] designing a learning-based filter and controller, the security problem is solved in this article. Comparisons also show that our previous work cannot work when attacks are implemented, further showing that it is important to design a secure control countermeasure to mitigate attacks.

The rest of this article is organized as follows. Section II provides the preliminaries and formulates the problem to be solved. Section III presents the learning-based false data injection attack algorithm. Section IV gives the secure robot learning framework for mitigating attacks. Section V provides the simulation and real-world experiment results. Finally, this article is concluded in Section VI, and Section VII provides some discussion.

Notation: The notations used throughout the article are defined as follows. $\mathbb{R}^n$ denotes the Euclidean space of n. The superscript "$\top$" denotes the matrix transpose. $\mathbb{E}\{x\}$ means the expectation of the stochastic variable x. $\|x\|$ denotes 2-norm of the vector.

II. PRELIMINARIES AND PROBLEM FORMULATION

This article mainly focuses on designing a secure learning framework for a robot under malicious false-data injection attacks. The blueprint of such a framework is given in Fig. 1, where three layers are included. First, a nominal controller, which can be chosen from existing results, is provided for the robot in the basic layer. Second, an adversary monitors the sensor data transmitted by the communication network and learns optimal false data attacks to inject into the control signals in the attack layer. Third, a learning-based secure control algorithm is designed to mitigate attacks in the secure control layer.

For a robot in Fig. 1, for example, a manipulator, mobile robot, or quadrotor, its dynamics can be described by the following equation:

$$\dot{x} = f(x) + g(x)u$$

where $x \in \mathbb{R}^{n_x}$ is the system state, $u \in \mathbb{R}^{n_u}$ is the control input, and $f(x)$ and $g(x)$ are nonlinear smooth functions.

Either using the control theory or using the reinforcement learning approach, one can design a controller[1] for the robotic system. To address the secure control problem of robots under unknown malicious attacks, we combine control theory and the reinforcement learning approach to design a secure control framework, based on which the secure algorithms eliminate the exact knowledge of the system model and make the robot satisfy the performance described in Definition 1.

Definition 1: A system is exponentially stable if there exists a scalar $0 < \lambda < 1$ and a scalar $c > 0$ such that the following inequality holds:

$$\|x(t)\| \leq c\|x(0)\|\lambda^t.$$

In this article, we take a car-like mobile robot (CMR) as an example to detail the design process and effectiveness of the proposed secure framework. The robot in Fig. 1 is a CMR. Specifically, a nominal controller is provided for the CMR to follow the desired trajectory. A malicious attacker modifies control signals by injecting false data attacks. The system defender provides a solution to the security problem of a CMR under cyber threats. In the following context, details are given and discussed.

A. Kinematic Model of a CMR

Fig. 2 depicts a CMR in our lab and its physical model. The motion of such a CMR can be described by a vector $[x, y, \theta]^\top$, where the pair (x, y) represents the position of the robot, and θ is the heading angle of the CMR. For the CMR, its kinematic model is described as follows [58]:

$$\dot{x} = v\cos\theta,\ \dot{y} = v\sin\theta,\ \dot{\theta} = \frac{v}{L}\tan\phi \tag{1}$$

where v is the velocity of the robot to be designed, L is the wheelbase, and ϕ is the steering angle of the front wheels to be designed.

[1]We use both the controller and policy alternatively in this article.

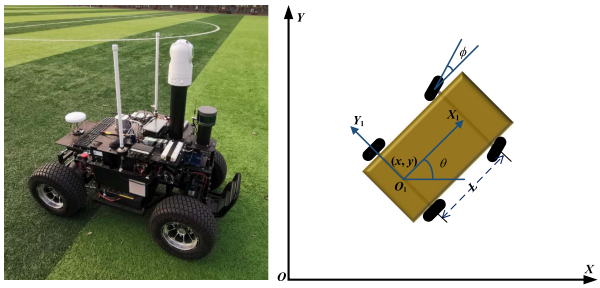


Fig. 2. Car-like robot in our lab and its physical model.

For a CMR, it is reasonable to assume that the velocity v satisfies $v_{\min} \le v < v_{\max}$ with $v_{\min}$ and $v_{\max}$ being the minimum as well as maximum velocities, respectively. The steering angle of the front wheels ϕ satisfies $\phi \in [-\frac{\pi}{6}, \frac{\pi}{6}]$. These constraints will be considered when the secure control algorithms designed in this article are implemented.

Remark 1: In [58], it has shown that the accuracy of the kinematic model in (1) is acceptable. Additionally, Wang and Low [59] provided a general and unifying kinematic model of mobile robots with wheel skidding and slipping. Although these physical constraints are not addressed in this article, the algorithms to be designed in the following content can be directly adopted to learn the corresponding false data injection attacks and secure control signals, where the error model (4) is replaced with that of [59].

The following virtual robot generates the desired trajectories:

$$\dot{x}_r = v_r \cos\theta_r,\ \dot{y}_r = v_r \sin\theta_r,\ \dot{\theta}_r = \frac{v_r}{L}\tan\phi_r \tag{2}$$

where v_r is the velocity of the reference robot and ϕ_r is the steering angle of the front wheels.

Based on (1) and (2), the error dynamics under the global coordinate (XOY) is described as

$$\begin{aligned}\dot{e}_x &= v\cos\theta - v_r\cos\theta_r\\ \dot{e}_y &= v\sin\theta - v_r\sin\theta_r\\ \dot{e}_\theta &= \frac{v}{L}\tan\phi - \frac{v_r}{L}\tan\phi_r\end{aligned} \tag{3}$$

where $e_x = x - x_r$, $e_y = y - y_r$, and $e_\theta = \theta - \theta_r$.

According to the geometric relationship, the local coordinate frame of the error system can be described as

$$\begin{bmatrix} x_e\\ y_e\\ \theta_e\end{bmatrix} = \begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix} e_x\\ e_y\\ e_\theta\end{bmatrix}$$

based on which we can derive

$$\begin{aligned}\dot{x}_e &= v - v_r\cos\theta_e + y_e\frac{v}{L}\tan\phi\\ \dot{y}_e &= v_r\sin\theta_r - x_e\frac{v}{L}\tan\phi\\ \dot{\theta}_e &= \frac{v}{L}\tan\phi - \frac{v_r}{L}\tan\phi_r.\end{aligned} \tag{4}$$

As shown in Fig. 1, our framework to be designed includes three layers. Accordingly, the control commands v and ϕ transmitted to the actuator consist of three parts, that is, a nominal controller $[v_b\ \phi_b]$, a malicious attack $[v_a\ \phi_a]$, and a secure controller $[v_d\ \phi_d]$. For the model in (1), there exist many excellent control schemes, see for example [60], [61], [62] and the references therein. Here, the scheme in [60], which will be regarded as a nominal controller in this article due to its robustness and ease of implementation, is given below.

Lemma 1: [60] If the reference inputs v_r, ϕ_r and the derivatives $\dot{v}_r$, $\dot{\phi}_r$ are bounded, the closed-loop tracking error system (4) can be globally stabilized by the following control scheme:

$$v_b = -k_1 x_e + v_r\cos\theta_e,\ \phi_b = \arctan\frac{\omega L}{v_b} \tag{5}$$

where

$$\begin{aligned}\omega =& -k_2\bar{\theta}_e + \omega_r - k_0 v_r y_e \bar{f} + \dot{\alpha},\ \omega_r = \frac{v_r}{L}\tan\phi_r\\ \bar{f} =& \frac{\sin\theta_e\cos\alpha + (\cos\theta_e - 1)\sin\alpha}{\bar{\theta}_e}\\ \alpha =& \alpha(t, x_e, y_e) = \rho(t)\bar{h}(t, x_e, y_e)\\ \dot{\rho} =& -(|v_r| + |\omega_r|)\rho,\ \rho(0) = 1\\ \dot{\alpha} =& -(|v_r| + |\omega_r|)\alpha + \rho\frac{\partial\bar{h}}{\partial t}\\ &+\rho\left(\frac{\partial\bar{h}}{\partial x_e}(v_b - v_r\cos\theta_e) + \frac{\partial\bar{h}}{\partial y_e}v_r\sin\theta_e\right)\\ \bar{\theta}_e =& \theta_e - \alpha\end{aligned}$$

and k_0, k_1, and k_2 are positive scalars to be designed. $\bar{h}$ is a C^2-class function, whose properties and alternative expressions can refer to [60].

The robot runs under network conditions. The Euler discretization method is utilized to obtain the discrete dynamics of the CMR. Then, (1) can be discretized as

$$\begin{aligned}x(k+1) &= x(k) + v(k)\cos\theta(k)\Delta t\\ y(k+1) &= y(k) + v(k)\sin\theta(k)\Delta t\\ \theta(k+1) &= \theta(k)\frac{v(k)}{L}\tan\phi(k)\Delta t\end{aligned} \tag{6}$$

where Δt is the discretization period.

The discretized form of the reference model is described as

$$\begin{aligned}x_r(k+1) &= x_r(k) + v\cos\theta_r(k)\Delta t\\ y_r(k+1) &= y_r(k) + v\sin\theta_r\Delta t\\ \theta_r(k+1) &= \theta_r(k) + \frac{v_r}{L}\tan\phi_r(k)\Delta t.\end{aligned} \tag{7}$$

The discretized form of the error system (4) is as follows:

$$\begin{aligned}x_e(k+1) =& x_e(k) + (v(k) - v_r(k)\cos\theta_e(k)\\ &+y_e(k)\frac{v}{L}\tan\phi(k)\Big)\Delta t\\ y_e(k+1) =& y_e(k) + (v_r(k)\sin\theta_r(k)\end{aligned}$$

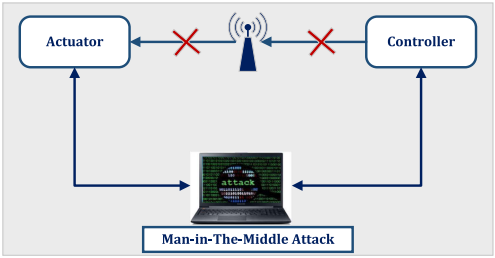


Fig. 3. Illustration of the man-in-the-middle attack.

$$-x_e(k)\frac{v(k)}{L}\tan\phi(k)\Big)\Delta t$$
$$\theta_e(k+1)=\theta_e(k)+\Big(\frac{v(k)}{L}\tan\phi(k)$$
$$-\frac{v_r(k)}{L}\tan\phi_r(k)\Big)\Delta t. \tag{8}$$

B. Attack Model

As shown in Fig. 1, a malicious adversary executes false data injection attacks to compromise control commands when control commands are transmitted to the actuator via the communication network. Under such attacks, the control commands are described as follows:

$$v(k)=v_a(k)+v_b(k)$$
$$\phi(k)=\phi_a(k)+\phi_b(k)$$

where $v_a(k)$ and $\phi_a(k)$ are the false data attacks, which will be designed in Section III. $v_b(k)$ and $\phi_b(k)$ are nominal control signals designed by using Lemma 1.

For adversaries, we assume that the following knowledge is known to them.

1) Adversaries inject malicious attacks without violating the physical constraints of the robot, by which adversaries not only can save the limited attack energies but guarantee the effectiveness of attacks to some degree.
2) Adversaries can access the communication network by using some attack methods, such as the man-in-the-middle attack, a blueprint given in Fig. 3. Additionally, adversaries can intercept the communication network between the sensor and the controller and eavesdrop on sensor data.

C. Motivation Example

This section only provides a numerical example to show the necessity of protecting a CMR's performance from deterioration. The experimental results are given in Section VI. The control scheme in Lemma 1 is used to demonstrate the tracking performance of the CMR with and without attacks, respectively. Define $k_0=6$, $k_1=1$, $k_2=3$, and $\bar{h}=1.2\tanh(x_e^2+y_e^2)\sin k$. The initial state of a CMR is set as $x=1$, $y=0.5$, and $\theta=0$. The reference trajectory is given as a U-shaped trajectory. When the CMR is not attacked, Fig. 4(a) shows the tracking trajectory without any attacks. In the simulation, we assume that the control commands are modified randomly by adversaries. As

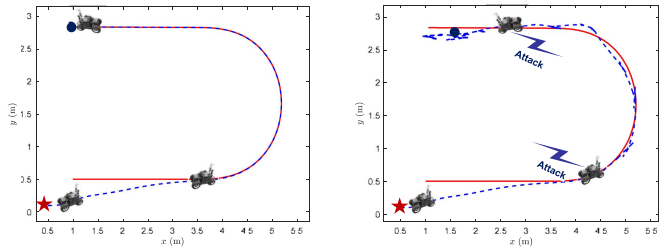


Fig. 4. Tracking performance with and without attacks. (a) Tracking performance without attacks. (b) Tracking trajectory under random false data injection attacks.

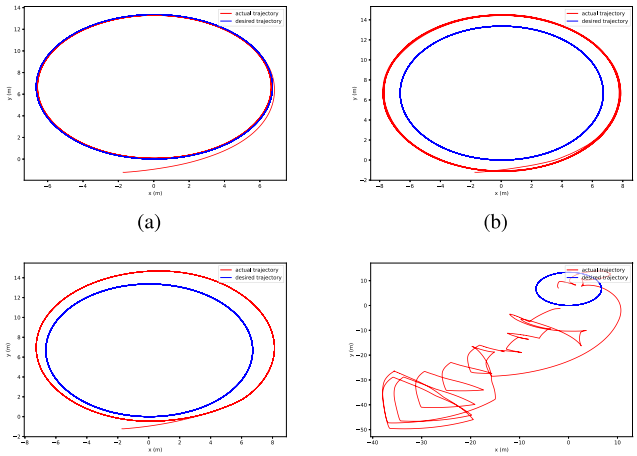


Fig. 5. Tracking performance under the algorithm in [52]. (a) Tracking performance without attacks. (b) Tracking performance under random attacks. (c) Tracking performance under state-dependent attacks. (d) Tracking performance under optimal attacks.

a comparison, false data injection attacks are generated randomly[2] are implemented when the CMR runs along the desired path. Fig. 4(b) shows the tracking trajectory under attacks. From these simulation results, we can conclude that a CMR is vulnerable to malicious adversaries. The CMR cannot run following the given U-shaped trajectory with a satisfying tracking performance. Accordingly, it is of great importance and necessity to provide countermeasures to prevent a robot's performance from deteriorating. Furthermore, the performance of the attack depends both on the amount of information available and on the way adversaries utilize it. As demonstrated in Fig. 5, false data injection attacks can deteriorate the tracking performance under a learning-based control algorithm. By comparing Fig. 5(b)–(d), it is obvious that optimal attacks can achieve the best attack performance. These motivate us to investigate the optimal false data injection attack and defense countermeasure design.

D. Problem Formulation

As can be seen from the above example and literature review in Section I, cyber threats can deteriorate and even destroy the

[2]Here, we only use random attacks to show the malicious effects on a CMR. Malicious adversaries can design an attack strategy based on their goals.

performance of robotic systems. This article mainly focuses on solving two problems. One is how to construct optimal false data injection attacks to disturb robotic systems. The other is how to design a secure control algorithm for robotic systems against false data injection attacks. These two problems to be solved are described mathematically as follows.

Problem 1: An attacker intends to deteriorate the tracking performance by injecting false data attacks with minimum attack cost. Then, the optimal attack design problem can be formulated as follows:

$$\begin{aligned} u_a(k) = &\arg\max_{u_a(k)} J_{\text{adv}} \\ \text{subject to} \quad &\bar{x}(k+1) = \bar{x}(k) + v(k)\cos\bar{\theta}(k)\Delta t \\ &\bar{y}(k+1) = \bar{y}(k) + v(k)\sin\bar{\theta}(k)\Delta t \\ &\bar{\theta}(k+1) = \bar{\theta}(k) + \frac{v(k)}{L}\tan\phi(k)\Delta t \\ &v_{\min} \le v \le v_{\max},\ \phi \in \left[-\frac{\pi}{6}, \frac{\pi}{6}\right] \end{aligned}$$

where $\bar{x}(k)$, $\bar{y}(k)$, as well as $\bar{\theta}(k)$ are the states of a CMR under attacks, and

$$\begin{aligned} u_a(k) &= \begin{bmatrix} v_a(k) & \phi_a(k) \end{bmatrix}^\top \\ v(k) &= v_a(k) + v_b(k),\ \phi(k) = \phi_a(k) + \phi_b(k) \\ J_{adv} &= \lim_{N\to\infty}\frac{1}{N}\sum_{k=1}^{N}\left\{\bar{X}^\top(k)Q_a\bar{X}(k) - u_a^\top(k)R_a u_a(k)\right\} \\ \bar{X}(k) &= \begin{bmatrix} \bar{x}(k) - x_r(k) & \bar{y}(k) - y_r(k) & \bar{\theta}(k) - \theta_r(k) \end{bmatrix}^\top \end{aligned}$$

and $Q_a \ge 0$ and $R_a > 0$ are weighting matrices.

Problem 2: Under attacks, the defender's objective is to find an optimal secure controller to mitigate attacks with a minimum control cost. Following this objective, the optimal secure control problem is formulated as follows:

$$\begin{aligned} u_d(k) = &\arg\min_{u_d(k)} J_{\text{sec}} \\ \text{subject to} \quad &\tilde{x}(k+1) = \tilde{x}(k) + v(k)\cos\tilde{\theta}(k)\Delta t \\ &\tilde{y}(k+1) = \tilde{y}(k) + v(k)\sin\tilde{\theta}(k)\Delta t \\ &\tilde{\theta}(k+1) = \tilde{\theta}(k)\frac{v(k)}{L}\tan\phi(k)\Delta t \\ &v_{\min} \le v \le v_{\max},\ \phi \in \left[-\frac{\pi}{6}, \frac{\pi}{6}\right] \end{aligned}$$

where $\tilde{x}(k)$, $\tilde{y}(k)$, as well as $\tilde{\theta}(k)$ are the states of a CMR under the secure control, and

$$\begin{aligned} u_d(k) &= \begin{bmatrix} v_d(k) & \phi_d(k) \end{bmatrix}^\top \\ v(k) &= v_a(k) + v_b(k) + v_d(k) \\ \phi(k) &= \phi_a(k) + \phi_b(k) + \phi_d(k) \\ J_{sec} &= \lim_{N\to\infty}\frac{1}{N}\sum_{k=1}^{N}\left\{\tilde{X}^\top(k)Q_d\tilde{X}(k) + u_d^\top(k)R_d u_d(k)\right\} \\ \tilde{X}(k) &= \begin{bmatrix} \tilde{x}(k) - x_r(k) & \tilde{y}(k) - y_r(k) & \tilde{\theta}(k) - \theta_r(k) \end{bmatrix}^\top \end{aligned}$$

and $Q_d \ge 0$ and $R_d > 0$ are weighting matrices.

III. Optimal False Data Injection Scheduling

In this section, a learning framework is proposed to design the optimal false data injection attacks to deteriorate the performance of the CMR. Several effective reinforcement learning algorithms have been proposed and applied widely, for example, deterministic policy gradient algorithm [45], trust region policy optimization algorithm [46], proximal policy optimization algorithm [47], and soft actor-critic reinforcement learning algorithm [48], each of which can be used here to solve *Problem 1*, and learn the optimal false data injection attacks. Here, we adopt the soft actor-critic reinforcement learning algorithm to solve the optimal attack problem.

A. Markov Decision Process of a CMR Under Attacks

A reinforcement learning setup includes the agent and the environment, which interact with each other to improve the agent's ability. The environment is described by a Markov decision process defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ means the action space, $\mathcal{P}$ denotes the transition probability distribution, $\mathcal{R}$ represents the attack cost,[3] and $\gamma \in [0, 1)$ is the discount factor. Then, combining the attacked CMR and reference dynamics in the discrete form, the Markov decision process can be described as

$$\bar{x}_1(k+1) \sim \mathcal{P}\left(\bar{x}_1(k+1)|\bar{x}_1(k), u_a(k)\right) \tag{9}$$

where $\bar{x}_1(k) \in \mathcal{S}$ with $\bar{x}_1(k) = [\bar{x}(k)\ \bar{y}(k)\ \bar{\theta}(k)\ \ x_r(k)\ y_r(k)\ \theta_r(k)]^\top$, $\mathcal{P}(\bar{x}_1(k+1)|\bar{x}_1(k), u_a(k))$ represents the transition probability from $\bar{x}_1(k)$ to $\bar{x}_1(k+1)$ under the attack $u_a(k)$.

B. Learning-Based Optimal False Data Injection Attack Algorithm

This section presents the definitions of the attack cost and action-value function, following which the attack policy to be learned is derived. Then, the gradients of the policies are given. An algorithm is presented to learn the false data injection attacks.

1) Notations in the Learning-Based Attack Algorithm: According to Problem ****1, the attack cost $\mathcal{R}(k)$ is defined as

$$\mathcal{R}(k) = \bar{X}^\top(k)Q_a\bar{X}(k) - u_a^\top(k)R_a u_a(k).$$

When the reinforcement learning algorithm is trained, the action-value function (also known as the Q function) $Q_{\pi_a}(\bar{x}_1(k), u_a(k))$ is maximized to find a solution to *Problem 1*, where π_a denotes the attack policy to be learned. The expression of $Q_{\pi_a}(\bar{x}_1(k), u_a(k))$ is computed as

$$Q_{\pi_a}\left(\bar{x}_1(k), u_a(k)\right) = \gamma\mathbb{E}_{\bar{x}_1(k+1)}\left[V_{\pi_a}(\bar{x}_1(k+1))\right] + \mathcal{R}(k)$$

where $\mathbb{E}_{\bar{x}_1(k+1)}[\cdot]$ is the expectation of $\sum_{\bar{x}_1(k+1)}\mathcal{P}_{k+1|k}[\cdot]$ over the distribution of $\bar{x}_1(k+1)$, $\mathcal{P}_{k+1|k} = \mathcal{P}(\bar{x}_1(k+$

[3] In reinforcement learning literature, it is called reward.

$1)|\bar{x}_1(k), u_a(k))$, $\pi_a(u_a(k)|\bar{x}_1(k))$ is the probability of choosing the attack $u_a(k)$ at state $\bar{x}_1(k)$ from the attack policy π_a, and

$$V_{\pi_a}(\bar{x}_1(k)) = \sum_{k}^{\infty} \sum_{u_a(k)} \pi_a(u_a(k)|\bar{x}_1(k)) \sum_{\bar{x}_1(k+1)} \mathcal{P}_{k+1|k} \times (\mathcal{R}(k) + \gamma V_{\pi_a}(\bar{x}_1(k+1))).$$

If we can find a solution to the following optimal problem, *Problem 1* can be solved

$$\pi_a^* = \arg\max_{\pi_a} Q_{\pi_a}(\bar{x}_1(k), u_a(k)) \tag{10}$$

where π_a^* is the optimal attack policy, and $u_a(k)$ samples from this optimal attack policy.

Next, the soft actor-critic reinforcement learning approach is introduced to learn the optimal attack policy π_a^*. In the soft actor-critic algorithm, an entropy item $\mathcal{H}_a(\pi(u_a(k+1)|\bar{x}_1(k+1)))$ is introduced to adequately explore the action space. Then, the optimal problem described in (10) is rewritten as

$$\pi_a^* = \arg\max_{\pi_a} (Q_{\pi_a}(\bar{x}_1(k), u_a(k)) + \alpha_a \mathcal{H}_a(\pi_a(u_a(k+1)|\bar{x}_1(k+1)))) \tag{11}$$

where α_a is a parameter used to regulate the importance of $\mathcal{H}_a(\pi(u_a(k+1)|\bar{x}_1(k+1)))$, and

$$\begin{aligned}&\mathcal{H}_a(\pi_a(u_a(k+1)|\bar{x}_1(k+1)))\\ &= -\sum_{u_a(k)} \pi_a(u_a(k)|\bar{x}_1(k)) \ln(\pi_a(u_a(k)|\bar{x}_1(k)))\\ &= -\mathbb{E}_{\pi_a}[\ln(\pi_a(u_a(k)|\bar{x}_1(k)))].\end{aligned}$$

According to the above description, the reinforcement learning algorithm is to solve the following problem:

$$\pi_a^* = \arg\max_{\pi_a \in \Pi_a} (\mathcal{R}(k) + \gamma \mathbb{E}_{\bar{x}_1(k+1)}[V_{\pi_a}(\bar{x}_1(k+1)) - \alpha_a \mathbb{E}_{\pi_a}[\ln(\pi_a(u_a(k)|\bar{x}_1(k)))]])$$

where Π_a is the policy set.

In the soft actor-critic algorithm, the policy evaluation and policy improvement steps repeatedly execute to learn the optimal attack policy π_a^*. In the policy evaluation step, a Bellman backup operator $\mathcal{T}^{\pi_a}$ is utilized to repeatedly compute the soft action-value function $Q_{\pi_a}(\bar{x}_1(k), u_a(k))$, the computation of which is as follows:

$$\mathcal{T}^{\pi_a} Q_{\pi_a}(\bar{x}_1(k), u_a(k)) = \mathcal{R}(k) + \gamma \mathbb{E}_{\bar{x}_1(k+1)}[V_{\pi_a}(\bar{x}_1(k+1))]$$

where

$$V_{\pi_a}(\bar{x}_1(k)) = \mathbb{E}_{\pi_a}[Q_{\pi_a}(\bar{x}_1(k), u_a(k)) - \ln(\pi_a(u_a(k)|\bar{x}_1(k)))].$$

In the policy improvement step, the policy is updated by

$$\pi_a^{\text{new}} = \arg\min_{\pi_a' \in \Pi} \mathscr{D}_{\text{KL}}\left(\pi_a'(\cdot|\bar{x}_1(k)) \,\middle\|\, \frac{e^{\frac{1}{\alpha} Q^{\pi_a^{\text{old}}}(\bar{x}_1(k), \cdot)}}{Z^{\pi_a^{\text{old}}}}\right) \tag{12}$$

where π_a^{old} is the policy from the last update, $Q^{\pi_a^{old}}$ is the Q-value of π_a^{old}, $\mathscr{D}_{\text{KL}}$ means the Kullback–Leibler divergence, and $Z^{\pi_a^{\text{old}}}$ denotes a normalization factor. Then, (12) can be rewritten as

$$\pi_a^* = \arg\min_{\pi_a \in \Pi_a} \mathbb{E}_{\pi_a}[\alpha \ln(\pi_a(u_a(k)|\bar{x}_1(k))) - Q_{\pi_a}(\bar{x}_1(k), u_a(k))]. \tag{13}$$

To solve the optimization problem formulated in (13), deep neural networks are utilized to approximate, respectively, the action-value function $Q_{\pi_a,\theta_a}(\bar{x}_1(k), u_a(k))$, and the policy $\pi_{a,\varphi_a}(u_a(k)|\bar{x}_1(k))$, where θ_a and φ_a are employed to parameterized neural networks. In the following, the updating rules (i.e., the gradients) and a learning algorithm are given.

2) Updating Rules and Implementation of the Attack Algorithm: The parameter θ_a is trained by minimizing the following Bellman residual:

$$J_{Q_{\pi_a}}(\theta_a) = \mathbb{E}_{\bar{x}_1(k), u_a(k) \sim \mathcal{M}_a}\left[\frac{1}{2}\left(Q_{\pi_{a,\theta_a}}(\bar{x}_1(k), u_a(k)) - \mathcal{R}(k) - \gamma \mathbb{E}_{\bar{x}_1(k+1)}[V_{\pi_{a,\bar{\theta}_a}}(\bar{x}_1(k+1))]\right)^2\right]$$

the gradient estimate of which is as follows:

$$\begin{aligned}\nabla_{\theta_a} J_{Q_{\pi_a}}(\theta_a) = \nabla_{\theta_a} J_{Q_{\pi_a}}(\theta_a) &\left(Q_{\pi_{a,\theta_a}}(\bar{x}_1(k), u_a(k)) - \mathcal{R}(k) - \gamma Q_{\pi_{a,\bar{\theta}_a}}(\bar{x}_1(k+1), u_a(k+1))\right.\\ &\left.+\gamma \alpha_a \ln(\pi_{a,\varphi_a}(u_a(k+1))|\bar{x}_1(k+1))\right)\end{aligned}$$

where $\mathcal{M}_a$ denotes the dataset generated by the system.

The parameter φ_a is trained by minimizing the following equation with the reparameterization trick [48]:

$$J_{\pi_a}(\varphi_a) = \mathbb{E}_{\pi_a,\varphi_a}[\alpha_a \ln(\pi_{a,\varphi_a}(f_{\varphi_a}((\epsilon(k); \bar{x}_1(k))|\bar{x}_1(k)))) - Q_{\pi_a,\theta_a}(\bar{x}_1(k), f_{\varphi_a}((\epsilon(k); \bar{x}_1(k))))]$$

the gradient estimate of which is

$$\begin{aligned}\nabla_{\varphi_a} J_{\pi_a}(\varphi_a) = \nabla_{\varphi_a} &\alpha_a \ln(\pi_{a,\varphi_a}(u_a(k)|\bar{x}_1(k)))\\ &+ \left(\nabla_{u_a(k)} \alpha_a \ln(\pi_{a,\varphi_a}(u_a(k)|\bar{x}_1(k))) - \nabla_{u_a(k)} Q_{\pi_a}(\bar{x}_1(k), u_a(k))\right)\\ &\times \nabla_{\varphi_a} f_{\varphi_a}(\epsilon(k); \bar{x}_1(k))\end{aligned}$$

where $u_a(k) = f_{\varphi_a}(\epsilon(k); \bar{x}_1(k))$ with $\epsilon(k)$ being a noise vector.

The parameter α_a is trained by minimizing the following:

$$J(\alpha_a) = \mathbb{E}_{\pi_a}\{-\alpha_a \ln \pi_a(u_a(k)|\bar{x}_1(k)) - \alpha_a \bar{\mathcal{H}}_a\}$$

where $\bar{\mathcal{H}}_a$ is a target entropy.

According to the gradients derived above, Algorithm 1 is given to training deep neural networks to learn the optimal false data injection attacks. The convergence analysis of Algorithm 1 can refer to [48], and it is omitted here for want for space.

Remark 2: Here, only the optimal false data injection attack design problem is addressed, while the attack detection scheme is not designed. To save the secure control cost, a χ^2 attack detector can be designed by combining the learning-based filter in our previous work [53], [54].

Algorithm 1: Learning-Based False Data Injection Attack Algorithm.

Initialize parameters θ_a, φ_a, α_a, and τ
Apply the controller in Lemma 1 to collect replay memory $\mathcal{M}_a$
Set the target parameter $\bar{\theta}_a$ as $\bar{\theta}_a \leftarrow \theta_a$
while Training **do**
  for each data collection step **do**
    Inject $u_a(k)$ into the control commands of the CMR with $u_a(k)$ sampling from the policy $\pi_{a,\varphi}(\bar{x}_1(k)|u_a(k))$
    Update the memory $\mathcal{M}_a \leftarrow \mathcal{M}_a \cup \bar{x}_1(k)$
  end for
  for each gradient step **do**
    $\theta_a \leftarrow \theta_a - \iota_{Q_a} \nabla_{\theta_a} J_{Q_{\pi_a}}(\theta_a)$,
    $\varphi_a \leftarrow \phi_a - \iota_{\pi_a} \nabla_{\varphi_a} J_{\pi_a}(\varphi_a)$
    $\alpha_a \leftarrow \alpha_a - \iota_{\alpha_a} \nabla_{\alpha_a} J(\alpha_a)$
    $\bar{\theta}_a \leftarrow \tau\theta_a + (1-\tau)\bar{\theta}_a$,
  end for
end while
Output optimal parameters θ_a^*, φ_a^*

IV. Learning-Based Countermeasure

From an adversary's perspective, a CMR's tracking performance can deteriorate by injecting malicious attack commands generated using Algorithm 1. In this section, we provide a solution to *Problem 2*, that is, a learning-based secure control algorithm with stability guarantee is proposed, under which the malicious attacks can be mitigated. Here, the soft actor-critic learning algorithm is modified by introducing a Lyapunov function constraint, which is used to guarantee that the tracking error of the CMR under the secure controller preserves the exponential stability. In what follows, related notation and the secure algorithm are detailed.

A. Markov Decision Process of a CMR With a Secure Controller

Similar to the descriptions in Section III, the Markov decision process of a CMR with a secure controller consists of five elements, that is, the state space $\bar{\mathcal{S}}$, the action space $\bar{\mathcal{A}}$, the control cost $\bar{\mathcal{C}}$, the transition probability $\bar{\mathcal{P}}$, and the discounted factor $\bar{\gamma}$. The involution of such a process is as follows:

$$\tilde{X}(k+1) \sim \bar{\mathcal{P}}\left(\tilde{X}(k+1)|\tilde{X}(k), u_d(k)\right) \tag{14}$$

where $\tilde{X}(k) \in \bar{\mathcal{S}}$, and $u_d(k) \in \bar{\mathcal{A}}$.

Remark 3: Differently from the Markov decision process for the attack case in (9), $\tilde{X}(k)$ is regarded as the state in (14). Readers may be curious about this setup. Generally, $\bar{x}_1(k)$ can be also defined as the state in (14). However, satisfying results cannot be obtained when we train the deep neural networks to learn the secure controller. Instead, if $\tilde{X}(k)$ is defined as the state, a secure controller can be successfully trained. We argue that the objective of the learning-based secure countermeasure is to recover the tracking performance (i.e., $\tilde{X}(k) \to 0$). When $\tilde{X}(k)$ instead of $\bar{x}_1(k)$ is defined as the state, deep neural networks can use more explicit input, removing unnecessary inferencing.

B. Learning-Based Secure Control Algorithm

This section describes how to design a secure control algorithm to mitigate false data injection attacks learned by using Algorithm 1. Based on the defender's objective, some notations of the learning-based secure control algorithm are given. A Lyapunov function is introduced to preserve stability, and the Lyapunov function constraint is added during the training process. Next, we detail how to design such a secure control algorithm.

1) Notations in the Learning-Based Secure Control Algorithm: The defender's objective is to use the minimum control cost to recover the tracking performance. Therefore, the cost $\bar{\mathcal{C}}(k)$ is defined as

$$\bar{\mathcal{C}}(k) = \tilde{X}^\top(k) Q_d \tilde{X}(k) + u_d^\top(k) R_d u_d(k)$$

where $Q_d \geq 0$ and $R_d > 0$ are weighting matrices.

Based on the definition of control cost $\bar{\mathcal{C}}(k)$, the defender's objective is to minimize the action-value function $Q_{\pi_d}(\tilde{X}(k), u_d(k))$ with π_d being the secure policy to be learned. The definitions of $Q_{\pi_d}(\tilde{X}(k), u_d(k))$ and $V_{\pi_d}(\tilde{X}(k))$ are omitted due to lack of space due to similarity to the previous definitions.

If we can find a solution to the following optimization problem, *Problem 2* can be solved

$$\pi_d^* = \arg\min_{\pi_d} Q_{\pi_d}\left(\tilde{X}(k), u_d(k)\right) \tag{15}$$

where π_d^* is the optimal secure policy and $u_d(k)$ samples from such an optimal secure policy.

To guarantee that the entropy of action can be maximized, the value-action function $Q_{\pi_d}(\tilde{X}(k), u_d(k))$ adding the entropy item $\mathcal{H}_d$ is rewritten as [4]

$$Q_{\pi_d}\left(\tilde{X}(k), u_d(k)\right) = \bar{\mathcal{C}}(k) + \bar{\gamma}\mathbb{E}_{\tilde{X}(k+1)}\Big[V_{\pi_d}(\tilde{X}(k+1)) - \alpha_d \mathcal{H}_d(\pi_d(u_d(k+1)|\tilde{X}(k+1)))\Big]$$

where α_d means a temperature parameter, which is used to adjust the relative importance of the entropy item. The entropy of policy is defined as $\mathcal{H}_d(\pi_d(u(k+1)|\tilde{X}(k+1)))$, the expression of which is similar to $\mathcal{H}_a(\pi(u_a(k+1)|\bar{x}_1(k+1)))$.

Then, the learning-based secure control algorithm is to solve the following optimization problem, which is equivalent to *Problem 2*:

$$\pi_d^* = \arg\min_{\pi_d \in \Pi_d}\Big(\bar{\mathcal{C}}(k) + \bar{\gamma}\mathbb{E}_{\tilde{X}(k+1)}\Big[V_{\pi_d}(\tilde{X}(k+1)) - \alpha_d \mathcal{H}_d\left(\pi_d\left(u(k+1)|\tilde{X}(k+1)\right)\right)\Big]\Big) \tag{16}$$

where Π_d is the policy set.

[4] In the soft actor-critic reinforcement learning algorithm, the objective of which is to maximize both the reward and the entropy of action, the entropy item $\alpha_d \mathcal{H}_d(\pi_d(u_d(k+1)|\tilde{X}(k+1)))$ is introduced. Instead, $-\alpha_d \mathcal{H}_d(\pi_d(u_d(k+1)|\tilde{X}(k+1)))$ is used here to minimize the secure control cost $\bar{\mathcal{C}}(k)$ but maximize the entropy of action.

Since the learning-based secure control algorithm to be designed is also in the actor-critic framework, both the policy evaluation and policy improvement steps need to be executed repeatedly to learn the optimal attack policy π_d^*. Similarly, the Bellman backup operation is utilized to repeatedly compute the soft action-value function $Q_{\pi_d}(\tilde{X}(k), u_d(k))$ including the entropy item in the policy evaluation step. The Bellman backup operation can refer to that in Section III.

By minimizing the control cost while maximizing the entropy of action, π_d^* can be solved by

$$\pi_d^* = \arg\min_{\pi_d \in \Pi_d} \mathbb{E}_{\pi_d}\left[\alpha_d \ln\left(\pi_d\left(u_d(k)|\tilde{X}(k)\right)\right) + Q\left(\tilde{X}(k), u_d(k)\right)\right]. \tag{17}$$

Here, suppose we directly construct deep neural networks to approximate the action-value function and the policy and use the updating rules similar to those given in Section III. In that case, the secure controller can also be learned. In such a scenario, the secure countermeasure is an empirical algorithm whose stability cannot be guaranteed. A main improvement of the secure control algorithm to be proposed in this article is that the stability of the tracking error system can be preserved by using the secure controller. Therefore, the Lyapunov function often used in the control community is introduced. As discussed in our previous results [52], [53], [54], a direct choice of the Lyapunov function is the action-value function $Q_{\pi_d}(\tilde{X}(k), u_d(k))$. If we can ensure that $Q_{\pi_d}(\tilde{X}(k), u_d(k))$ preserves the properties of the Lyapunov function in the learning process, that is, $Q_{\pi_d}(\tilde{X}(k+1), u_d(k+1)) - Q_{\pi_d}(\tilde{X}(k), u_d(k)) < -\beta\bar{\mathcal{C}}(k)$ with β being a scalar in the learning process, it is possible to analyze the stability of the tracking error system.

Thus, the policy is improved by solving the following constrained optimization problem:

$$\pi_d^* = \arg\min_{\pi_d \in \Pi_d} \mathbb{E}_{\pi_d}\left[\alpha_d \ln\left(\pi_d\left(u_d(k)|\tilde{X}(k)\right)\right) + Q\left(\tilde{X}(k), u_d(k)\right)\right]$$

subject to

$$Q_{\pi_d}(\tilde{X}(k+1), u_d(k+1)) - Q_{\pi_d}(\tilde{X}(k), u_d(k)) < -\beta\bar{\mathcal{C}}(k). \tag{18}$$

To design a learning-based secure control algorithm to obtain a solution to (18), the deep neural networks with parameters θ_d and φ_d are constructed to approximate $Q_{\pi_d,\theta_d}(\tilde{X}(k), u_d(k))$, and the policy $\pi_{d,\varphi_d}(u_d(k)|\tilde{X}(k))$, respectively. Next, the updating rules for parameters θ_d and φ_d and the learning-based secure control algorithm are detailed.

2) Updating Rules and Implementation of the Secure Algorithm: Here, θ_d is also trained by minimizing the Bellman residual defined as follows:

$$J_{Q_{\pi_d}}(\theta_d) = \mathbb{E}_{\tilde{X}(k), u_d(k) \sim \mathcal{M}_d}\left[\frac{1}{2}\left(Q_{\pi_{d,\theta_d}}\left(\tilde{X}(k), u_d(k)\right) - \bar{\mathcal{C}}(k) - \bar{\gamma}\mathbb{E}_{\tilde{X}(k+1)}\left[V_{\pi_{d,\bar{\theta}_d}}(\tilde{X}(k+1))\right]\right)^2\right]$$

the gradient estimate of which is as follows:

$$\nabla_{\theta_d} J_{Q_{\pi_d}}(\theta_d) = \nabla_{\theta_d} Q_{\pi_{d,\theta_d}}\Big(Q_{\pi_{d,\theta_d}}(\tilde{X}(k), u_d(k)) - \bar{\mathcal{C}}(k) - \gamma Q_{\pi_d,\theta_d}(\tilde{X}(k+1), u_d(k+1)) - \gamma\alpha_d \ln(\pi_{d,\varphi_d}(u_d(k+1))|\tilde{X}(k+1))\Big)$$

where $\mathcal{M}_d$ denotes the dataset generated by the system.

Remark 4: The function $Q_{\pi_d,\theta_d}(\tilde{X}(k), u_d(k))$ is set as the Lyapunov function in the training process. However, such a function is parameterized by deep neural networks. The positive definite property of a Lyapunov function cannot always be satisfied. A trick can be used when $Q_{\pi_d,\theta_d}(\tilde{X}(k), u_d(k))$ is computed in the training process, for example, use its absolute value.

Combining the reparameterization trick and a Lagrangian multiplier, φ_d is trained by minimizing the following equation:

$$J_{\pi_d}(\varphi_d) = \mathbb{E}_{\pi_d,\varphi_d}\Big[\alpha_d \ln\left(\pi_{d,\varphi_d}(f_{\varphi_d}((\epsilon(k); \tilde{X}(k))|\tilde{X}(k)))\right) + Q_{\pi_d,\theta_d}\left(\tilde{X}(k), f_{\varphi_a}((\epsilon(k); \tilde{X}(k)))\right)\Big] + \lambda\left(Q_{\pi_d,\theta_d}(\tilde{X}(k+1), u_d(k+1)) - Q_{\pi_d}(\tilde{X}(k), u_d(k)) + \beta\bar{\mathcal{C}}(k)\right)\Big]$$

the gradient estimate of which is

$$\nabla_{\varphi_d} J_{\pi_d}(\varphi_d) = \nabla_{\varphi_d}\alpha_d \ln(\pi_{d,\varphi_d}(u_d(k)|\tilde{X}(k))) + \Big(\nabla_{u_d(k)}\alpha_d \ln(\pi_{d,\varphi_d}(u_d(k)|\tilde{X}(k))) - \nabla_{u_d(k)} Q_{\pi_d}(\tilde{X}(k), u_d(k))\Big) \times \nabla_{\varphi_d} f_{\varphi_d}(\epsilon(k); \tilde{X}(k)) + \lambda\nabla_{u_{d+1}} Q_{\pi_d}\left(\tilde{X}(k+1), u_d(k+1)\right) \times \nabla_{\varphi_d} f_{\varphi_d}(\epsilon(k); \tilde{X}(k+1))$$

where $u_d(k) = f_{\varphi_d}(\epsilon(k); \tilde{X}(k))$.

In the training process, α_d and λ are updated by maximizing the following equaiton:

$$J(\alpha_d) = \mathbb{E}_{\pi_d}\left[\alpha_d \ln \pi_d(u_d(k)|\tilde{X}(k)) + \alpha_d\bar{\mathcal{H}}_d\right]$$

$$J(\lambda) = \lambda\mathbb{E}\left[Q_{\pi_d,\theta_d}(\tilde{X}(k+1), f_{\theta_d}(\tilde{X}(k+1); \epsilon(k))) - Q_{\pi_d}(\tilde{X}(k), u_d(k)) + \beta\bar{\mathcal{C}}(k)\right]$$

where $\bar{\mathcal{H}}_d$ is the target entropy.

Using the above updating rules, Algorithm 2 is proposed to train the parameters of deep neural networks. Once deep neural networks have been successfully trained, secure control commands sampling from policy π_d can be applied to recover the CMR tracking performance.

Algorithm 2: Learning-Based Secure Control Algorithm.

1: Randomly initialize parameters θ_d, φ_d, α_d, and λ
2: Set the target parameter $\bar{\theta}_d$ as $\bar{\theta}_d \leftarrow \theta_d$
3: Apply the controller in Lemma 1 and the attack $u_a(k)$ learned by Algorithm 1 to collect replay memory $\mathcal{M}_d$
4: **while** Training **do**
5: **for** each data collection step **do**
6: Sample $u_d(k)$ from the policy $\pi_{d,\theta_d}(\tilde{X}(k)|u_d(k))$
7: Apply $u_d(k)$ to generate data $\tilde{X}(k)$
8: Update the replay memory $\mathcal{M}_d \leftarrow \mathcal{M}_d \cup \tilde{X}(k)$
9: **end for**
10: **for** each gradient step **do**
11: $\theta_d \leftarrow \theta_d - \iota_{Q_d} \nabla_{\theta_d} J_{Q_{\pi_d}}(\theta_d)$,
12: $\varphi_d \leftarrow \varphi_d - \iota_{\pi_d} \nabla_{\varphi_d} J_{\pi_d}(\varphi_d)$
13: $\alpha_d \leftarrow \alpha_d - \iota_\alpha \nabla_{\alpha_d} J_{\alpha_d}(\alpha_d)$
14: $\lambda \leftarrow \lambda - \iota_\lambda \nabla_\lambda J_\lambda(\lambda)$
15: $\bar{\theta}_d \leftarrow \tau\theta_d + (1-\tau)\bar{\theta}_d$,
16: **end for**
17: **end while**
18: Output optimal parameters θ_d^*, ϕ_d^*

C. Stability Analysis

As declared in the above section, the stability of the tracking error system under the learned secure control scheme can be ensured. Here, the strict proof is provided. Before proceeding, an assumption is given to complete the Proof of Theorem 1.

Assumption 1: An Markov chain induced by a policy π_d is ergodic with a unique distribution probability $q_{\pi_d}(\tilde{X}(k))$ with $q_{\pi_d}(\tilde{X}(k)) = \lim_{k\to\infty} \bar{\mathcal{P}}(\tilde{X}(k) \mid \rho, \pi_d, k)$.

Theorem 1: Assumption 1 holds. Provided that there are scalars $\tilde{\alpha}_1 > 0$, $\tilde{\alpha}_2 > 0$, $\beta \geq 0$ such that the Lyapunov function $Q(k)$[5] learned by using Algorithm 2 satisfies the following conditions:

$$\tilde{\alpha}_1 \bar{\mathcal{C}}(k) \leq Q(k) \leq \tilde{\alpha}_2 \bar{\mathcal{C}}(k) \tag{19}$$

$$\begin{aligned} &\mathbb{E}_{\tilde{X}(k)\sim\mu_{\pi_d}} \left[\mathbb{E}_{\tilde{X}(k+1)\sim\bar{\mathcal{P}}_{\pi_d}} [Q(k+1)] - Q(k) \right] \\ \leq\ & -\beta \mathbb{E}_{\tilde{X}(k)\sim\mu_{\pi_d}} \left[\bar{\mathcal{C}}(k) \right]. \end{aligned} \tag{20}$$

Then the tracking error of a CMR is guaranteed to be exponentially stable in mean square, i.e.,

$$\left\| \mathbb{E}_{\tilde{X}(k)\sim\mu_{\pi_d}} \left[\tilde{X}(k) \right] \right\| \leq \sigma^k \frac{\tilde{\alpha}_2}{\tilde{\alpha}_1} \left\| \mathbb{E}_{\tilde{X}(0)\sim\mu_{\pi_d}} \left[\tilde{X}(0) \right] \right\|$$

where

$$\mu_{\pi_d}(\tilde{X}(k)) \triangleq \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N} \bar{\mathcal{P}} \left(\tilde{X}(k) \mid \rho, \pi_d, k \right)$$

is the state distribution, and $\sigma \in (0,1)$.

Proof: Based on Assumption 1, the sampling distribution $\mu_{\pi_d}(\tilde{X}(k))$ exists. When $k \to \infty$, $q_{\pi_d}(\tilde{X}(k)) = \bar{\mathcal{P}}(\tilde{X}(k) \mid \rho, \pi_d, k)$. Using the Abelian theorem, the sequence $\{\frac{1}{N}\sum_{k=0}^{N} \bar{\mathcal{P}}(\tilde{X}(k) \mid \rho, \pi_d, k), N \in \mathbb{Z}_+\}$ also converges, and $\mu_{\pi_d}(\tilde{X}(k)) = q_{\pi_d}(\tilde{X}(k))$. According to the above discussion, (20) is rewritten as

$$\begin{aligned} &\int_{\mathcal{S}} \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N} \bar{\mathcal{P}}(\tilde{X}(k)|\rho, \pi_d, k) \\ &\quad \times \left(\mathbb{E}_{\bar{\mathcal{P}}_{\pi_d}(\tilde{X}(k+1)|\tilde{X}(k))} [Q(k+1)] - Q(k) \right) \mathrm{d}\tilde{X}(k) \\ &\quad \leq -\beta \mathbb{E}_{\tilde{X}(k)\sim q_{\pi_d}} \left[\bar{\mathcal{C}}(k) \right]. \end{aligned} \tag{21}$$

In Algorithm 2, we can learn a bounded Lyapunov function $Q(k)$. Then, $\bar{\mathcal{P}}(\tilde{X}(k)|\rho, \pi_d, k)Q(k)$ is bounded for $\forall \tilde{X}(k) \in \mathcal{S}$. Besides, the sequence $\{\frac{1}{N}\sum_{k=0}^{N} \bar{\mathcal{P}}(\tilde{X}(k)|\rho, \pi_d, k)Q(k)\}$ converges to the function $q_{\pi_d}(\tilde{X}(k))Q(k)$ in a pointwise way.

Based on Lebesgue's dominated convergence theorem, if a sequence $f_n(\tilde{X}(k))$ converges pointwise to a function f and is dominated by some integrable function $g(\tilde{X}(k))$ in the sense that

$$\left| f_n(\tilde{X}(k)) \right| \leq g(\tilde{X}(k)) \quad \forall \tilde{X}(k) \in \mathcal{S} \quad \forall n.$$

Then, the following equation can be obtained:

$$\lim_{n\to\infty} \int_{\mathcal{S}} f_n(\tilde{X}(k)) \mathrm{d}\tilde{X}(k) = \int_{\mathcal{S}} \lim_{n\to\infty} f_n(\tilde{X}(k)) \mathrm{d}\tilde{X}(k).$$

Using the above equation, (21) can be described as

$$\begin{aligned} &\int_{\mathcal{S}} \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N} \bar{\mathcal{P}}(\tilde{X}(k) \mid \rho, \pi_d, k) \\ &\quad \times \left(\int_{\mathcal{S}} \bar{\mathcal{P}}_{\pi_d} \left(\tilde{X}(k+1)|\tilde{X}(k) \right) \right. \\ &\quad \left. \times Q(k+1) \mathrm{d}\tilde{X}(k+1) - Q(k) \right) \mathrm{d}\tilde{X}(k) \\ &= \lim_{N\to\infty} \frac{1}{N} \left(\sum_{k=1}^{N+1} \mathbb{E}_{P(\tilde{X}(k)|\rho,\pi_d,k)} [Q(k)] \right. \\ &\quad \left. - \sum_{k=0}^{N} \mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(k)|\rho,\pi_d,k)} [Q(k)] \right) \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N} \left(\mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(k+1)|\rho,\pi_d,k+1)} [Q(k+1)] \right. \\ &\quad \left. - \mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(k)|\rho,\pi_d,k)} [Q(k)] \right). \end{aligned}$$

There always exists a scalar σ such that the following equation holds:

$$\left(\frac{1}{\sigma} - 1 \right) \tilde{\alpha}_2 - \frac{\beta}{\sigma} = 0.$$

Combining the above equation and the condition in (19) yields the following result:

$$\begin{aligned} &\frac{1}{\sigma^{\iota+1}} \mathbb{E}_{\bar{\mathcal{P}}(\tilde{x}_{\iota+1}|\rho,\pi_d,\iota+1)} [Q(\iota+1)] - \frac{1}{\sigma^{\iota}} \mathbb{E}_{\bar{\mathcal{P}}(\tilde{x}_{\iota}|\rho,\pi_d,\iota)} [Q(\iota)] \\ &= \frac{1}{\sigma^{\iota+1}} \left(\mathbb{E}_{\bar{\mathcal{P}}(\tilde{x}(\iota+1)|\rho,\pi_d,\iota+1)} [Q(\iota+1)] \right. \end{aligned}$$

[5]For simplicity, $Q(k) \triangleq Q_{\pi_d}(\tilde{X}(k), u_d(k))$.

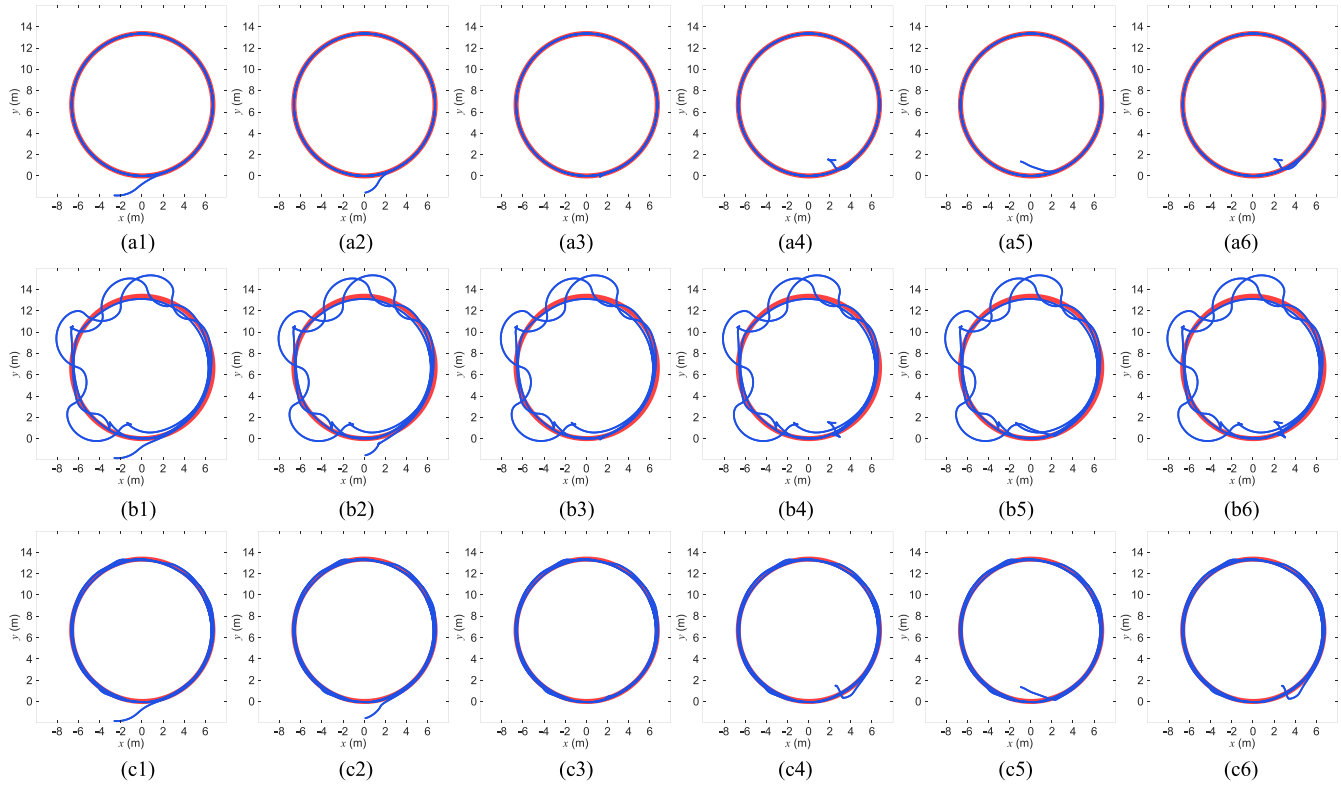


Fig. 6. Simulation results of the circular trajectory. Top: Tracking control of a CMR using the nominal controller in Lemma 1. Middle: Tracking control of a CMR under false data injection attacks designed by Algorithm 1. Bottom: Tracking control of a CMR under the secure controller learned using Algorithm 2.

$$
\begin{aligned}
&-\mathbb{E}_{\bar{\mathcal{P}}(\tilde{x}(\iota)|\rho,\pi_d,\iota)}[Q(\iota)]\big) \\
&+\frac{1}{\sigma^{\iota}}\left(\frac{1}{\sigma}-1\right)\mathbb{E}_{\bar{\mathcal{P}}(\iota|\rho,\pi_d,\iota)}[Q(\iota)] \\
\leq\ &\frac{1}{\sigma^{\iota}}\left(-\frac{\beta}{\sigma}+\left(\frac{1}{\sigma}-1\right)\tilde{\alpha}_2\right)\bar{\mathcal{C}}(k)
\end{aligned}
$$

which implies

$$
\frac{\mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(\iota+1)|\rho,\pi_d,\iota+1)}[Q(\iota+1)]}{\sigma^{\iota+1}}-\frac{\mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(\iota)|\rho,\pi_d,\iota)}[Q(\iota)]}{\sigma^{\iota}}\leq 0.
$$

To sum the above inequality from $\iota=0,1,\ldots,k-1$ yields

$$
\frac{1}{\sigma^{k}}\mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(k)|\rho,\pi_d,k)}[Q(k)]-\mathbb{E}_{\bar{\mathcal{P}}(\tilde{X}(0)|\rho,\pi_d,0)}[Q(0)]\leq 0
$$

which implies

$$
\mathbb{E}_{\tilde{X}(k)\sim\mu_{\pi_d}}\left[\bar{\mathcal{C}}(k)\right]\leq\sigma^{k}\frac{\tilde{\alpha}_2}{\tilde{\alpha}_1}\mathbb{E}_{\tilde{X}(0)\sim\mu_{\pi_d}}\left[\bar{\mathcal{C}}(0)\right].
$$

Furthermore, we can obtain the following inequality:

$$
\mathbb{E}_{\tilde{X}(k)\sim\mu_{\pi_d}}\left[\tilde{X}(k)\right]\leq\sigma^{k}\frac{\tilde{\alpha}_2}{\tilde{\alpha}_1}\mathbb{E}_{\tilde{X}(0)\sim\mu_{\pi_d}}\left[\tilde{X}(0)\right].
$$

Therefore, the tracking error using the learned policy exponentially converges. ■

V. Simulation and Experiments

Both simulation and real-world experiment results are provided to demonstrate the effectiveness of the proposed schemes in this section. The CMR is driven to follow two typical trajectories (i.e., circular and eight-type trajectories), respectively. To keep the parameters and constraints of the CMR in the simulation the same as those in the experiment, set the wheelbase $L=0.88$ m, the steering angle $-\frac{\pi}{6}\leq\phi\leq\frac{\pi}{6}$, and the velocity -1.5 m/s $\leq v\leq 1.5$ m/s. Both the simulation and experimental results are provided in three scenarios presented (i.e., without attacks, under attacks, and under the defense scheme).

A. Simulation

We first provide the simulation results to validate the proposed optimal false data injection attack scheme and the secure countermeasure. In the simulation, six different initial poses of the CMR are given randomly. The sampling period is set as 0.1 s. The desired circular trajectory is generated by the commands $v_r(k)=0.7$ m/s and $\phi_r(k)=\frac{\pi}{30}$. The desired eight-type trajectory is defined as $x_r(k)=12\sin(\frac{2k\pi}{150})$, $y_r(k)=12\sin(\frac{k\pi}{150})$.

Case 1: Tracking control of a CMR using the nominal controller in Lemma 1

For parameters of the nominal controller in Lemma 1, define $k_0=6$, $k_1=1$, $k_2=3$, and $\bar{h}=1.2\tanh(x_e^2+y_e^2)\sin k$. The first row in Figs. 6 and 7 shows the tracking performance of the nominal controller. As can be seen from these simulation figures, the CMR can track the desired circular and eight-type trajectories quickly. Without false data injection attacks, such a controller works well.

Case 2: Tracking control of a CMR under false data injection attacks designed by Algorithm 1

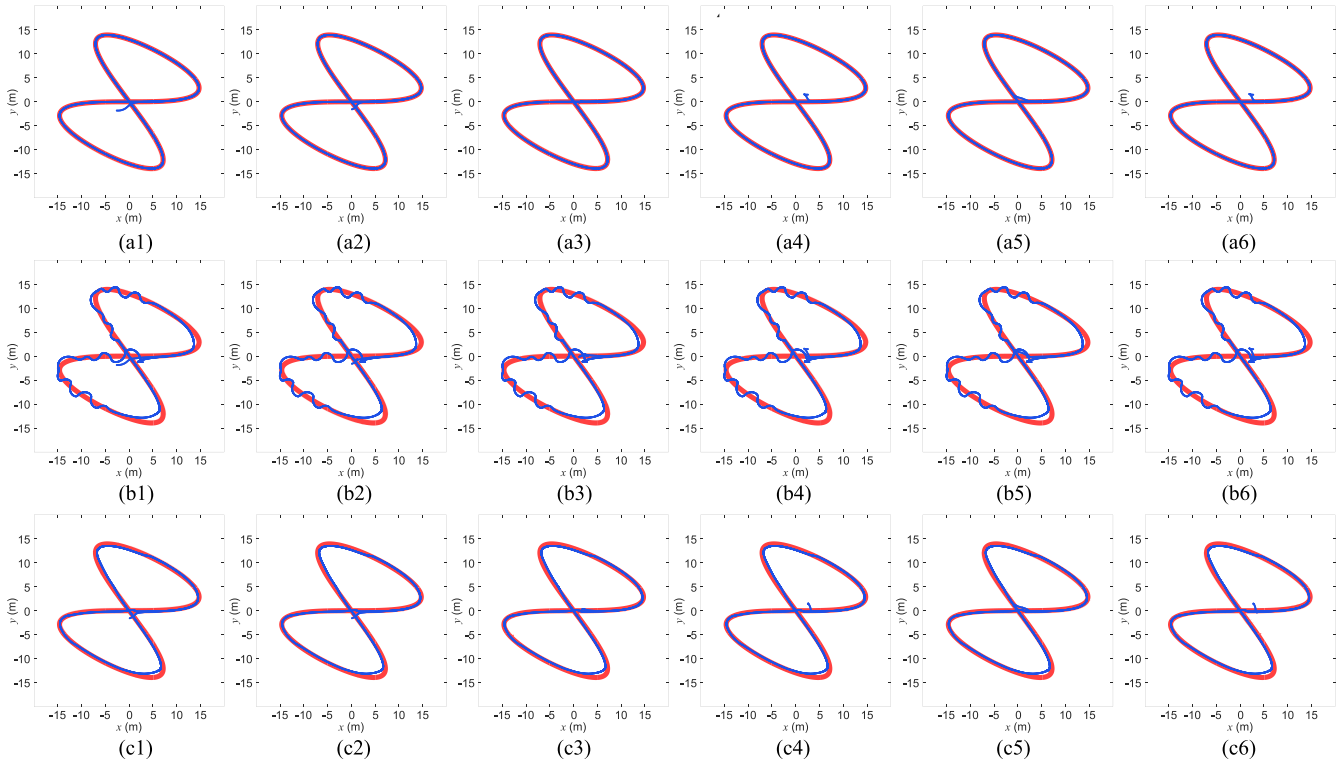


Fig. 7. Simulation results of the eight-type trajectory. Top: Tracking control of a CMR using the nominal controller in Lemma 1. Middle: Tracking control of a CMR under false data injection attacks designed by Algorithm 1. Bottom: Tracking control of a CMR under the secure controller learned using Algorithm 2.

TABLE II
HYPERPARAMETERS FOR ALGORITHM 1

Hyperparameters	Value
Length of sampling trajectories	800
Minibatch size	256
Learning rate ι_{π_a}	1e 4
Learning rate ι_{Q_a}	3e 4
Learning rate ι_{α_a}	1e 4
Target entropy $\bar{\mathcal{H}}_a$	2
Soft replacement τ	0.005
Discount factor γ	0.9
Structure of neural networks for actor	(64, 32)
Structure of neural networks for critic	(128, 128)

TABLE III
HYPERPARAMETERS FOR ALGORITHM 2

Hyperparameters	Value
Length of sampling trajectories	800
Minibatch size	256
Learning rate ι_{Q_d}	1e 4
Learning rate ι_{π_d}	6e 4
Learning rate ι_{α_d}	3e 4
Learning rate ι_{λ}	1e 4
Target entropy $\bar{\mathcal{H}}_d$	-2
Soft replacement(τ)	0.005
Discount factor ($\bar{\gamma}$)	0.9
β	0.06
Structure of neural networks for actor	(64, 32)
Structure of neural networks for critic	(256, 128, 64)

Here, we show that optimally designed false data injection attacks can deteriorate the CMR tracking performance. Table II gives the hyperparameters used in Algorithm 1. The rectified linear unit is chosen as the activation function for deep neural networks. Each attack policy is trained to 1000 episodes under the nominal controller. Ten optimal attack policies are learned, from which we choose one to attack the CMR. Note that the CMR needs to run a time period before it follows the desired trajectories. Such a time period depends on the initial pose of the CMR. If the initial states of the CMR are close to those of the desired trajectories, the time period is short; otherwise, it is long. Considering the fact that malicious adversaries often execute attacks when the CMR runs stably, the attacks are implemented at $k = 20$ (i.e., 2s) in training. The second row in Figs. 6 and 7 demonstrate the simulation results of the CMR under the learned false data injection attacks. The circular and eight-type paths cannot be followed with a satisfying performance. Although a robust tracking controller is adopted, the tracking performance of the CMR deteriorates. In this case, the CMR can crash into obstacles, other robots, and humans, causing damage to robots and even injuring humans. If robots in an industrial production line are attacked, defective products will be dramatically produced. Simulation results in this case not only show the effectiveness of the proposed attack scheme but also highlight the necessity and significance of securing robots.

Case 3: Tracking control of a CMR under the secure controller learned using Algorithm 2

In this case, we show that the tracking performance of an attacked CMR can be recovered by using the secure controller in Algorithm 2. The hyperparameters used in Algorithm 2 are provided in Table III. The activation function of the neural

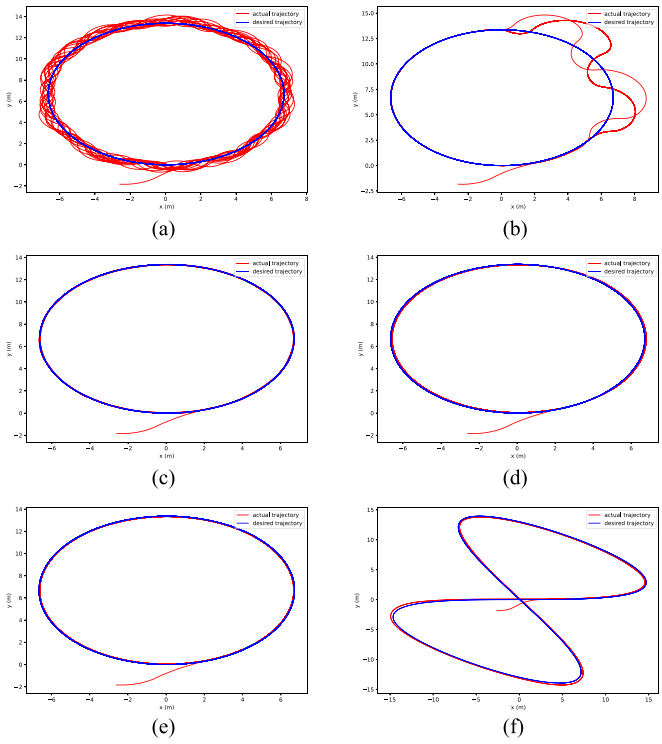


Fig. 8. Tracking performance with other attacks and without attacks under Algorithm 2. (a) Tracking performance with random attacks. (b) Tracking performance with state-dependent attacks. (c) Tracking performance under Algorithm 2 when random attacks occur. (d) Tracking performance under Algorithm 2 when state-dependent attacks occur. (e) Tracking performance without attacks: circle trajectory. (f) Tracking performance without attacks: eight-type trajectory.

network and the number of training episodes are the same as those in Case 2. One secure policy is chosen from the ten trained secure policies to mitigate attacks. The simulation results are depicted in the third row in Figs. 6 and 7, from which we can see that the tracking performance is recovered. To further show the effectiveness of the proposed secure control algorithm, random and state-dependent attacks are taken into account. Fig. 8(a) and (c) demonstrates the tracking performance under random attacks and Algorithm 2, respectively. Although both random attacks and state-dependent attacks often considered in the literature can deteriorate tracking performance, the optimal attacks designed in this article outperform the above two, further indicating that the performance of the attack depends both on the amount of information available and on the way attackers use it. Tracking performance under state-dependent attacks and Algorithm 2 are, respectively, depicted in Fig. 8(b) and (d). These results demonstrate that the proposed secure control algorithm mitigates optimal attacks and works well under other kinds of attacks. When no attacks occur, Fig. 8(e) and (f), respectively, show the simulation results for circle and 8-type trajectories under Algorithm 2, demonstrating the effectiveness of the proposed secure control algorithm without attacks occurring.

Under the secure controller, there still exists a small tracking error. If the fine-tuning method is used to train deep neural networks continuously, such errors can also decrease. Alternative approaches are to tune the parameters in the nominal

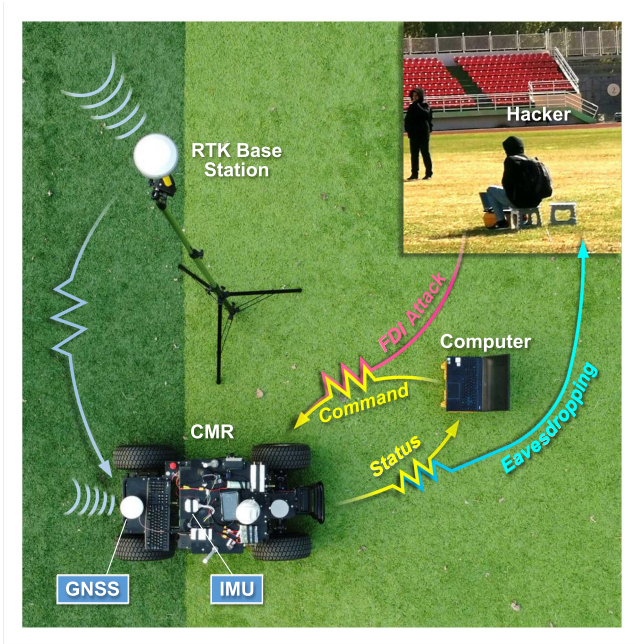


Fig. 9. Experiment setup. The real-time kinematic position system (RTK) is used to locate the CMR. The inertial measurement unit (IMU) is utilized to measure the heading angle of the CMR. The pose information obtained by the RTK and IMU is transmitted to the computer to calculate the control commands. Then, these control signals are sent to the actuators of the CMR. The hacker eavesdrops on the sensing data, based on which false data injection (FDI) attacks are computed and injected into control commands to disturb the locomotion of the CMR.

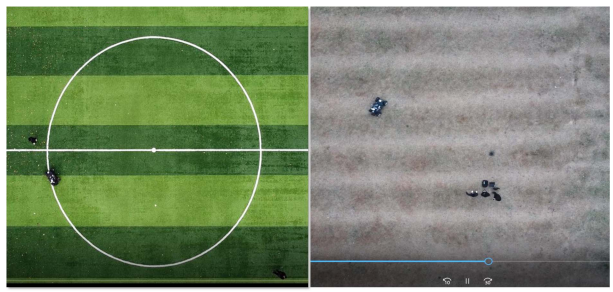


Fig. 10. Snapshots of different experiment environments.

controller. Readers may wonder why defenders design another control scheme to mitigate attacks rather than directly tuning the parameters of the nominal controller. The relation between these parameters and the performance is implicit. Nobody knows how long it takes if defenders tune parameters to recover the performance. Robots can cause damage before expedient parameters are found. On the other hand, since malicious adversaries can alter attacks, it is impossible always to tune these parameters.

B. Experiments

Here, we directly deploy the trained neural networks in the CMR. The setup of the experiment is given in Fig. 9. The experiments are conducted in two different real-world environments, snapshots of which are provided in Fig. 10. Three scenarios are also considered in these two environments. The desired circular trajectory is $x_r(k) = 9\cos(\frac{k\pi}{90})$,

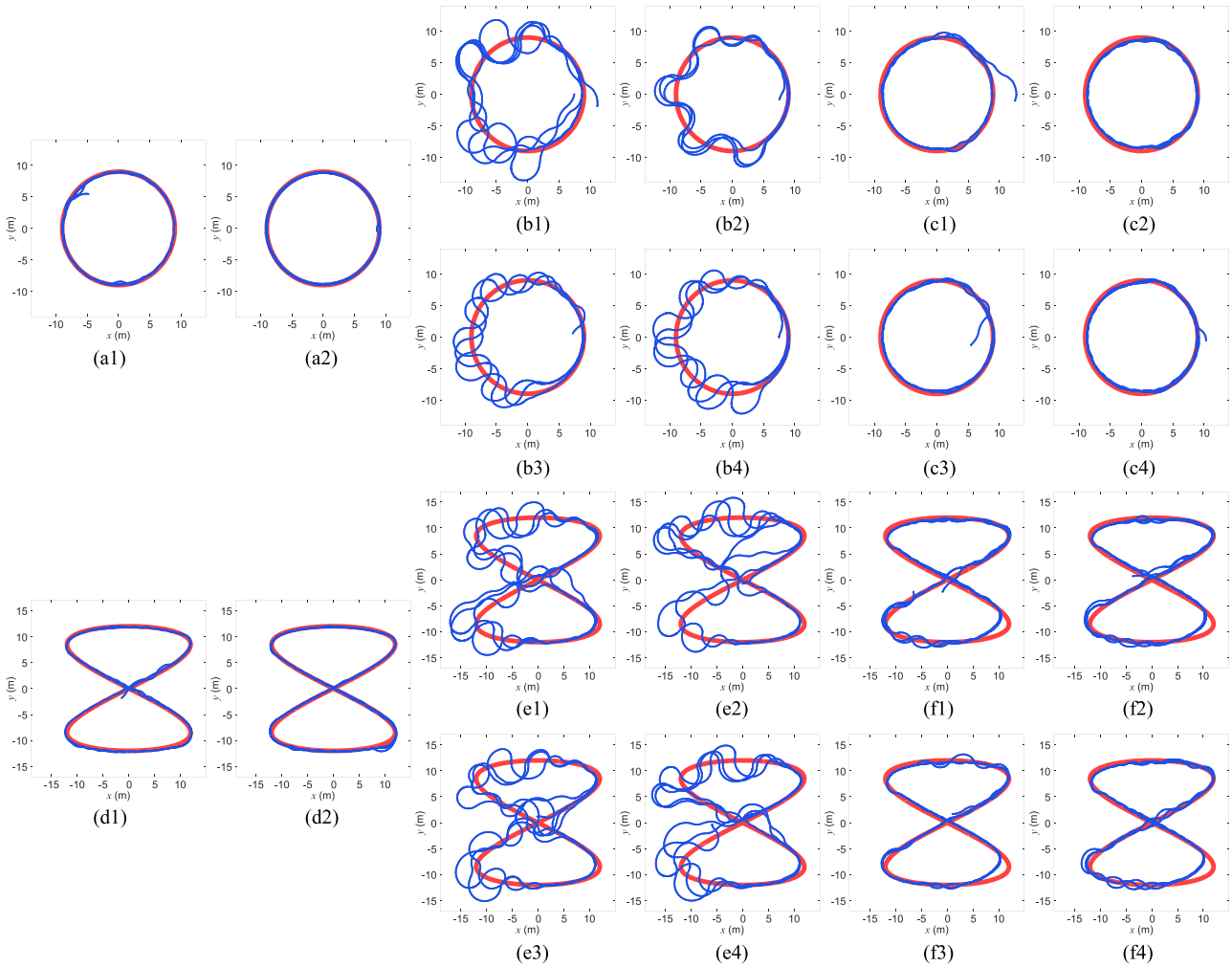


Fig. 11. Experiment results of circular and eight-type trajectories. First column: Tracking control of a CMR using the nominal controller in Lemma 1. Second column: Tracking control of a CMR under false data injection attacks designed by Algorithm 1. Last column: Tracking control of a CMR under the secure controller learned using Algorithm 2.

$y_r(k) = 9\sin(\frac{k\pi}{90})$. The desired eight-type trajectory is defined as $x_r(k) = 12\sin(\frac{k\pi}{75})$, $y_r(k) = 12\sin(\frac{k\pi}{150})$. Fig. 11 shows the results of the experiment. The experiment videos are available at http://aius.hit.edu.cn/ASecureRobotLearningFrameworkforCyberAttackSchedulingandCountermeasure/list.htm. From these experimental results, we can conclude that malicious adversaries can deteriorate the performance of the CMR by using the designed false data injection attack algorithm, and the performance of a CMR under attacks can be recovered by using the proposed secure learning control framework.

VI. Conclusion

The problems of constructing optimal false data injection attacks and countermeasure design for a robot have been solved in this article. By describing the dynamics of a CMR under false data injection attacks as a Markov decision process, a deep reinforcement learning algorithm has been given to determine the optimal attacks based on which the tracking performance of the robot can deteriorate. A secure robot learning control algorithm has been proposed combining both the reinforcement learning approach and the Lyapunov stable theory. Using such an algorithm, the performance of a robot under malicious attacks can be recovered effectively. More importantly, the stability of the tracking error system has been preserved by introducing the Lyapunov function, and the strict mathematical proof has been given. Since stability is a fundamental requirement of a control system, while most existing deep reinforcement learning approaches are empirical, the proposed framework can make deep reinforcement learning approaches widely applied to design control schemes, guaranteeing stability from the theoretical perspective. Rich simulations and experiments have been conducted to show the effectiveness of the proposed learning framework.

VII. Discussion

Security is critical for robots equipped with modern devices and communication. Several cyber attack incidents mentioned in Section I show the necessity of designing secure control schemes for robots.

Although some secure algorithms have been proposed, most depend on the linear model. When attacks occur, there exists no guarantee that linear models can still describe the actual system dynamics. In the literature, the proposed algorithms are applied to linear models rather than practical applications. Consequently, the effectiveness of existing results is not thoroughly evaluated. The optimal false data injection attack design and secure control have been formulated in a nonlinear form in this article. Using the deep reinforcement learning approach, a secure robot learning framework was established, using which the performance of a robot under malicious cyber threats can be recovered effectively, and the stability of the tracking error can be guaranteed.

This article has two critical improvements. One is that the trained neural networks in the simulation are directly deployed in a CMR of our lab. Surprisingly, trained neural networks drive the CMR well (see Section V). This can be explained by using a nominal controller, which could guarantee both the simulation and the experiment achieve satisfying results. Moreover, we conducted another experiment on a cable-driven robot in which the trained neural networks in the simulation were directly deployed. The experimental results[6] also show that the use of a nominal controller helps solve the sim-to-real transfer problem. The other lies in that optimal attack and secure control design problems formulated in nonlinear forms are solved using the deep reinforcement learning approach. This not only ensures that the proposed framework can be more widely applied but also suggests a possibility of making deep reinforcement learning approaches widely applied to designing control schemes due to the stability guarantee of our approach.

REFERENCES

[1] G. W. Clark, M. V. Doran, and T. V. Andel, "Cybersecurity issues in robotics," in *Proc. IEEE Conf. Cogn. Comput. Aspects Situation Manage.*, 2017, pp. 1–5.

[2] M. Olivato, O. Cotugno, L. Brigato, D. Bloisi, A. Farinelli, and L. Iocchi, "A comparative analysis on the use of autoencoders for robot security anomaly detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 984–989.

[3] Wikipedia, "Iran–U.S. RQ-170 incident," [Online]. Available: https://en.wikipedia.org/wiki/Iran%E2%80%93 U.S._RQ-170_incident

[4] A. H. Rutkin, ""spoofers" use fake GPS signals to knock a yacht off course," [Online]. Available: https://www.technologyreview.com/2013/08/14/177015/spoofers-use-fake-gps-signals-to-knock-a-yacht-off-course/

[5] F. Maggi, D. Quarta, M. Pogliani, M. Polino, A. M. Zanchettin, and S. Zanero, "Rogue robots: Testing the limits of an industrial robot's security," Trend Micro, Politecnico di Milano, Mlan, Italy, Tech. Rep., 2017, pp. 1–21.

[6] W. Dong, "Tracking control of multiple-wheeled mobile robots with limited information of a desired trajectory," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 262–268, Feb. 2012.

[7] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.

[8] Q. Zhou, S. Zhao, H. Li, R. Lu, and C. Wu, "Adaptive neural network tracking control for robotic manipulators with dead zone," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3611–3620, Dec. 2019.

[9] V. M. Gonçalves, B. V. Adorno, A. Crosnier, and P. Fraisse, "Stable-by-design kinematic control based on optimization," *IEEE Trans. Robot.*, vol. 36, no. 3, pp. 644–656, Jun. 2020.

[10] N. Bezzo, J. Weimer, M. Pajic, O. Sokolsky, G. J. Pappas, and I. Lee, "Attack resilient state estimation for autonomous robotic systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 3692–3698.

[11] Y. Jin, Y. Zhang, Y. Jing, and J. Fu, "An average dwell-time method for fault-tolerant control of switched time-delay systems and its application," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 3139–3147, Apr. 2019.

[12] J. Sun, H. Zhang, Y. Wang, and S. Sun, "Fault-tolerant control for stochastic switched IT2 fuzzy uncertain time-delayed nonlinear systems," *IEEE Trans. Cybern.*, vol. 52, no. 2, pp. 1335–1346, Feb. 2022.

[13] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, "Attack models and scenarios for networked control systems," in *Proc. 1st Int. Conf. High Confidence Networked Syst.*, 2012, pp. 55–64.

[14] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, "Unmanned aircraft capture and control via GPS spoofing," *J. Field Robot.*, vol. 31, no. 4, pp. 617–636, 2014.

[15] F. Fei et al., "Cross-layer retrofitting of UAVs against cyber-physical attacks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 550–557.

[16] P. Guo, H. Kim, N. Virani, J. Xu, M. Zhu, and P. Liu, "RoboADS: Anomaly detection against sensor and actuator misbehaviors in mobile robots," in *Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, 2018, pp. 574–585.

[17] S. Lee and B.-C. Min, "Distributed control of multi-robot systems in the presence of deception and denial of service attacks," 2021, *arXiv:2102.00098*.

[18] S. Banik and S. D. Bopardikar, "Attack-resilient path planning using dynamic games with stopping states," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 25–41, Feb. 2022.

[19] M. Santilli, M. Franceschelli, and A. Gasparri, "Dynamic resilient containment control in multirobot systems," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 57–70, Feb. 2022.

[20] F. Mallmann-Trenn, M. Cavorsi, and S. Gil, "Crowd vetting: Rejecting adversaries via collaboration with application to multirobot flocking," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 5–24, Feb. 2022.

[21] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure estimation and control for cyber-physical systems under adversarial attacks," *IEEE Trans. Autom. Control*, vol. 59, no. 6, pp. 1454–1467, Jun. 2014.

[22] C. Wu, Z. Hu, J. Liu, and L. Wu, "Secure estimation for cyber-physical systems via sliding mode," *IEEE Trans. Cybern.*, vol. 48, no. 12, pp. 3420–3431, Dec. 2018.

[23] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, "Kalman filtering with intermittent observations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1453–1464, Sep. 2004.

[24] A.-Y. Lu and G.-H. Yang, "Secure switched observers for cyber-physical systems under sparse sensor attacks: A set cover approach," *IEEE Trans. Autom. Control*, vol. 64, no. 9, pp. 3949–3955, Sep. 2019.

[25] L. An and G.-H. Yang, "LQ secure control for cyber-physical systems against sparse sensor and actuator attacks," *IEEE Trans. Control Netw. Syst.*, vol. 6, no. 2, pp. 833–841, Jun. 2019.

[26] C. Wu, W. Pan, G. Sun, J. Liu, and L. Wu, "Learning tracking control for cyber–physical systems," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 9151–9163, Jun. 2021.

[27] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *Proc. 47th Annu. Allerton Conf. Commun., Control, Comput.*, 2009, pp. 911–918.

[28] F. Miao, M. Pajic, and G. J. Pappas, "Stochastic game approach for replay attack detection," in *Proc. 52nd IEEE Conf. Decis. Control*, 2013, pp. 1854–1859.

[29] K. Manandhar, X. Cao, F. Hu, and Y. Liu, "Detection of faults and attacks including false data injection attack in smart grid using kalman filter," *IEEE Trans. Control Netw. Syst.*, vol. 1, no. 4, pp. 370–379, Dec. 2014.

[30] X. Luo, Y. Li, X. Wang, and X. Guan, "Interval observer-based detection and localization against false data injection attack in smart grids," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 657–671, Jan. 2021.

[31] L. Hu, Z. Wang, Q.-L. Han, and X. Liu, "State estimation under false data injection attacks: Security analysis and system protection," *Automatica*, vol. 87, pp. 176–183, 2018.

[32] Z. Kazemi, A. A. Safavi, F. Naseri, L. Urbas, and P. Setoodeh, "A secure hybrid dynamic-state estimation approach for power systems under false data injection attacks," *IEEE Trans. Ind. Inform.*, vol. 16, no. 12, pp. 7275–7286, Dec. 2020.

[33] A. Abbaspour, A. Sargolzaei, P. Forouzannezhad, K. K. Yen, and A. I. Sarwat, "Resilient control design for load frequency control system under false data injection attacks," *IEEE Trans. Ind. Electron.*, vol. 67, no. 9, pp. 7951–7962, Sep. 2020.

[6]The video is available at http://aius.hit.edu.cn/ASecureRobotLearningFrameworkforCyberAttackSchedulingandCountermeasure/list.htm.

[34] W. Lucia, B. Sinopoli, and G. Franze, "A set-theoretic approach for secure and resilient control of cyber-physical systems subject to false data injection attacks," in *Proc. Sci. Secur. Cyber- Phys. Syst. Workshop*, 2016, pp. 1–5.
[35] H. Zhang, P. Cheng, L. Shi, and J. Chen, "Optimal denial-of-service attack scheduling with energy constraint," *IEEE Trans. Autom. Control*, vol. 60, no. 11, pp. 3023–3028, Nov. 2015.
[36] J. Qin, M. Li, J. Wang, L. Shi, Y. Kang, and W. X. Zheng, "Optimal denial-of-service attack energy management against state estimation over an SINR-based network," *Automatica*, vol. 119, 2020, Art. no. 109090.
[37] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, pp. 1–33, 2011.
[38] S. Gao, H. Zhang, Z. Wang, and C. Huang, "A class of optimal switching mixed data injection attack in cyber-physical systems," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1598–1605, Apr. 2021.
[39] A. Gupta, C. Langbort, and T. Başar, "Dynamic games with asymmetric information and resource constrained players with applications to security of cyberphysical systems," *IEEE Trans. Control Netw. Syst.*, vol. 4, no. 1, pp. 71–81, Mar. 2017.
[40] Q. Zhu and S. Rass, "Game theory meets network security: A tutorial," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 2163–2165.
[41] C. Wu, X. Li, W. Pan, J. Liu, and L. Wu, "Zero-sum game based optimal secure control under actuator attacks," *IEEE Trans. Autom. Control*, vol. 66, no. 8, pp. 3773–3780, Aug. 2021.
[42] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[43] Z.-P. Jiang, T. Bian, and W. Gao, "Learning-based control: A tutorial and some recent results," *Found. Trends Syst. Control*, vol. 8, no. 3, pp. 176–284, 2020.
[44] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[45] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2016.
[46] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.
[47] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[48] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.
[49] J. Zheng, T. Zhang, C. Wang, M. Xiong, and G. Xie, "Learning for attitude holding of a robotic fish: An end-to-end approach with sim-to-real transfer," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 1287–1303, Apr. 2022, doi: 10.1109/TRO.2021.3098239.
[50] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Sci. Robot.*, vol. 5, no. 47, 2020, Art. no. eabc5986.
[51] Q. Zhang, W. Pan, and V. Reppa, "Model-reference reinforcement learning for collision-free tracking control of autonomous surface vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 8770–8781, Jul. 2022, doi: 10.1109/TITS.2021.3086033.
[52] M. Han, L. Zhang, J. Wang, and W. Pan, "Actor-critic reinforcement learning for control with stability guarantee," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6217–6224, Oct. 2020.
[53] L. Hu, C. Wu, and W. Pan, "Lyapunov-based reinforcement learning state estimator," 2020, *arXiv:2010.13529*.
[54] Y. Tang, L. Hu, Q. Zhang, and W. Pan, "Reinforcement learning compensated extended Kalman filter for attitude estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 6854–6859.
[55] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2021.3121870.
[56] X. Y. Lee, Y. Esfandiari, K. L. Tan, and S. Sarkar, "Query-based targeted action-space adversarial policies on deep reinforcement learning agents," in *Proc. ACM/IEEE 12th Int. Conf. Cyber- Phys. Syst.*, 2021, pp. 87–97.
[57] G. Wu, J. Sun, and J. Chen, "Optimal data injection attacks in cyber-physical systems," *IEEE Trans. Cybern.*, vol. 48, no. 12, pp. 3302–3312, Dec. 2018.
[58] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *Proc. IEEE Intell. Veh. Symp.*, 2015, pp. 1094–1099.
[59] D. Wang and C. B. Low, "Modeling and analysis of skidding and slipping in wheeled mobile robots: Control design perspective," *IEEE Trans. Robot.*, vol. 24, no. 3, pp. 676–687, Jun. 2008.
[60] Y. Wang, Z. Miao, H. Zhong, and Q. Pan, "Simultaneous stabilization and tracking of nonholonomic mobile robots: A Lyapunov-based approach," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 4, pp. 1440–1450, Jul. 2015.
[61] T. Fukao, H. Nakagawa, and N. Adachi, "Adaptive tracking control of a nonholonomic mobile robot," *IEEE Trans. Robot. Autom.*, vol. 16, no. 5, pp. 609–615, Oct. 2000.
[62] J. Huang, C. Wen, W. Wang, and Z.-P. Jiang, "Adaptive output feedback tracking control of a nonholonomic mobile robot," *Automatica*, vol. 50, no. 3, pp. 821–831, 2014.

Chengwei Wu (Member, IEEE) received the B.S. degree in management from the Arts and Science College, Bohai University, Jinzhou, China, in 2013, the M.S. degree in engineering from Bohai University, in 2016, and the Ph.D. degree in engineering from Harbin Institute of Technology, China, 2021.

From July 2015 to December 2015, he was a Research Assistant with the Department of Mechanical Engineering, The Hong Kong Polytechnic University. From 2019 to 2021, he was a joint-Ph.D. student with the Department of Cognitive Robotics, Delft University of Technology, The Netherlands. He is currently an Assistant Professor with the Harbin Institute of Technology, Harbin, China. His research interests include reinforcement learning, sliding mode control and cyber-physical systems.

Weiran Yao (Member, IEEE) received the bachelor's (with Hons.) degree, the master's degree, and the doctor's degree in aeronautical and astronautical science and technology from the School of Astronautics, Harbin Institute of Technology (HIT), Harbin, China, in 2013, 2015, and 2020, respectively. From 2017 to 2018, he was a visiting Ph.D. student with the Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada.

He is currently an Assistant Professor with the School of Astronautics, HIT. His research interests include unmanned vehicles, multirobot mission planning, and multiagent control systems.

Wensheng Luo (Member, IEEE) received the B.S. degree in mechanical design manufacturing and automation from Harbin Engineering University, Harbin, China, 2007; the M.S. degree in mechatronic engineering and the Ph.D. degree in control theory and control engineering both from Harbin Institute of Technology, Harbin, China, 2009 and 2019, respectively; and the Ph.D. degree in automatic, electronic, and telecommunication engineering from the University of Seville, Seville, Spain, 2019.

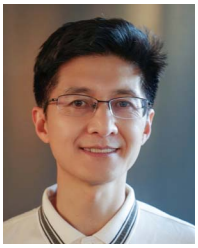
Dr. Luo is currently an Associate Professor with the School of Electrical Engineering and Automation, Harbin Institute of Technology, China. Her research interests include linear parameter varying system, sliding mode control, model predictive control, and their application in power converters. She has published more than 20 papers in journals. She is hosting research projects funded by National Natural Science Foundation of China, China Postdoctoral Science Foundation, etc.

Wei Pan received the Ph.D. degree in bioengineering from Imperial College London, London, U.K., 2017.

He is currently a Senior Lecturer (Associate Professor) in machine learning with the Department of Computer Science and a member of Centre for AI Fundamentals and Centre for Robotics and AI, University of Manchester, Manchester, U.K. Before that, he was an Assistant Professor in robot dynamics with the Department of Cognitive Robotics and co-Director of Delft SELF AI Lab, Delft University of Technology, Netherlands and a Project Leader at DJI, China. He is an Area Chair or Associate Editor of IEEE Robotics and Automation Letters, ACM Transactions on Probabilistic Machine Learning, Conference on Robot Learning, IEEE International Conference on Robotics and Automation, IEEE/RSJ International Conference on Intelligent Robots and Systems. He has broad interest in robot control using machine learning and the principles of dynamic control.

Guanghui Sun received the B.S. degree in automation and the M.S. and Ph.D. degrees in control science and engineering from Harbin Institute of Technology, Harbin, China, in 2005, 2007, and 2010, respectively.

He is currently a Professor with the Department of Control Science and Engineering, Harbin Institute of Technology. His research interests include fractional-order systems, networked control systems, and sliding mode control.



Hui Xie (Senior Member, IEEE) received the B.S. degree in mechanical engineering from Harbin University of Science and Technology, Harbin, China, in 2000 the M.S. and the Ph.D. degrees in mechatronics engineering from Harbin Institute of Technology, Harbin, China, in 2002 and 2006, respectively.

He is currently a Professor with the State Key Laboratory of Robotics and Systems, Harbin Institute of Technology. His current research interests include micro and nanorobotics, micro and nanomanipulation/characterization, bio-inspired small/soft robots and swimming micro and nanorobots.

Ligang Wu (Fellow, IEEE) received the B.S. degree in automation from Harbin University of Science and Technology, Harbin, China, in 2001, the M.E. degree in navigation guidance and control and the Ph.D. degree in control theory and control engineering both from Harbin University of Science and Technology, Harbin, China, in 2003, and 2006, respectively.

From 2006 to 2007, he was a Research Associate with the Department of Mechanical Engineering, The University of Hong Kong, Hong Kong. From 2007 to 2008, he was a Senior Research Associate with the Department of Mathematics, City University of Hong Kong, Hong Kong. From 2012 to 2013, he was a Research Associate with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K. In 2008, he joined the Harbin Institute of Technology, China, as an Associate Professor, and was then promoted to a Full Professor in 2012. He has published 7 research monographs and more than 170 research papers in international referred journals. His current research interests include switched systems, stochastic systems, computational and intelligent systems, sliding mode control, and advanced control techniques for power electronic systems.

Dr. Wu was the winner of the National Science Fund for Distinguished Young Scholars in 2015, and received China Young Five Four Medal in 2016. He was named as the Distinguished Professor of Chang Jiang Scholar in 2017, and was named as the Highly Cited Researcher in 2015–2019. He currently serves as an Associate Editor for a number of journals, including IEEE TRANSACTIONS ON AUTOMATIC CONTROL, IEEE/ASME TRANSACTIONS ON MECHATRONICS, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, *Information Sciences, Signal Processing,* and *IET Control Theory and Applications.* He is an Associate Editor for the Conference Editorial Board, IEEE Control Systems Society.