Q-learning based fault estimation and fault tolerant iterative learning control for MIMO systems

Rui Wang[a], Zhihe Zhuang[a], Hongfeng Tao[a,*], Wojciech Paszke[b], Vladimir Stojanovic[c,*]

[a]*Key Laboratory of Advanced Process Control for Light Industry (Ministry of Education), Jiangnan University, Wuxi 214122, China*
[b]*Institute of Automation, Electronic and Electrical Engineering, University of Zielona Góra, ul. Szafrana 2, 65-516 Zielona Góra, Poland*
[c]*Faculty of Mechanical and Civil Engineering, Department of Automatic Control, Robotics and Fluid Technique, University of Kragujevac, Kraljevo 36000, Serbia*

Abstract

This paper proposes a Q-learning based fault estimation (FE) and fault tolerant control (FTC) scheme under iterative learning control (ILC) framework. Due to its repetitive property of the demand on the control actuator, ILC is sensitive to actuator faults. Moreover, unknown faults could bring uncertainties to the system dynamics, which is a challenge to the control performance. Therefore, this paper introduces Q-learning algorithm to estimate the unknown actuator faults without need of prior knowledge for controller reconfiguration. Then, the design of FTC adopts the norm-optimal iterative learning control (NOILC) framework, where the controller is adjusted based on the FE results from Q-learning in real time to counteract the influence of faults. Finally, the simulation of a mobile robot verifies the effectiveness of the proposed algorithm.

*Corresponding author
Email addresses: taohongfeng@hotmail.com (Hongfeng Tao), vladostojanovic@mts.rs (Vladimir Stojanovic)

Keywords:
iterative learning control, fault estimation, fault tolerant control,
Q-learning, MIMO systems

1. Introduction

Iterative learning control (ILC) is an intelligent control algorithm for systems executing repetitive task during a finite interval, whose core idea is to modify the input signals by incorporating information from the previous trial with the purpose of driving the system to follow the desired reference more precisely. In recent years, ILC has been widely applied to various control areas on account of its simplicity and high tracking performance, typically for instance, industrial robotic systems [1], network systems [2], batch processes [3], automotive systems [4] and clinical mechanical support [5], etc. For comprehensive information of ILC, please refer to [6], [7] and therein.

Due to the uncertainties and challenges in complex working conditions, it is not practical for repetitive task systems to maintain the same system parameters and more probability is brought to expose the system to faults. Meanwhile, ILC schemes could be especially sensitive to faults due to the repeated nature of the demand on the control actuator [8], which means more consideration should be taken into actuator faults under ILC framework. As the modern technology becomes more sophisticated, the controlled systems are increasingly vulnerable to faults. Fault tolerant control (FTC) aims to maintain the safe operating of the systems with faults and mitigate the influence of the faults, in which case the minor faults can be prevented from

developing into major problems. Therefore, although fault tolerant control (FTC) issues arise in other field [9], approaches to FTC under the ILC framework still have profound research value. At present, limited work has been devoted to ILC schemes with actuator faults. The work in [10] proposes a design of closed-loop ILC which shortens the control time period with the iteration proceeding to improve the robustness of the system when actuator faults occur. In [11, 12], reliable control, a traditional passive FTC method is introduced to the design of ILC update law with actuator faults. However, the above works only pay attention to the reliability of the systems in the presence of actuator faults, which motivates the design objective of this paper to not only focus on the reliability but also raise more concerns about the tracking performance of the systems under the ILC framework with the existence of actuator faults.

High tracking performance is a main focus of the ILC framework, in order to take its full advantage, there are plenty of literatures studying on performance optimization [13]. In [14, 15], conventional optimal methods such as gradient descent method and newton method are employed to derive the optimal ILC update law. In [16], the general optimization form of ILC is proposed and the controlled system is redescribed in form of super vectors in the trial domain through lifted technique, which lays a foundation for the following work about norm-optimal iterative learning control (NOILC) [17, 18]. A parameter optimal iterative learning control (POILC) can perform well in convergence speed but has a limitation of the convergence property for the practical systems, e.g. [19]. It can be seen from the above work that general optimal ILC requires the exact model of the controlled system. Then

based on the model information, the performance criterion is optimized to complete the optimal control task. However, the existence of the unknown faults introduces uncertainties into a definite system dynamics, which results in the poor control performance. In addition, with the negative effect of the faults gradually accumulating, the systems will finally deviate from the desired control objective. Hence, the other design objective of this paper is to compensate the uncertain change of the system dynamics caused by actuator faults.

The key to maintaining the performance of the system with actuator faults is to mitigate the influence of the unknown faults. Fault estimation (FE) method can directly reconstruct the fault signals and provide powerful support for FTC, which can counteract the uncertainties resulting from the unknown faults. It is a thorny issue to estimate the complicated and unknown faults, whereas reinforcement learning (RL) provides a feasible way to learn in an unknown environment without need for abundant prior data [20]. Regarding the unknown faults as the unknown environment, Q-learning algorithm, one of the most significant advances in RL field, can be introduced to learn and estimate the unknown faults, which takes state action function into consideration and controls the system based on temporal difference method without need of model information [21]. In other words, Q-learning algorithm has a relatively simple structure with no need of prior knowledge and can provide real-time faults information for FTC. Recently, Q-learning has been involved to estimation related work in limited study. The work in [22], Q-learning is used for estimating the probability matrix of random graphs. The study in [23], the secure state is estimated by Q-learning in

cyber-physical system. These works inspire this paper to utilize the learning ability of Q-learning to estimate the unknown faults. Then, the FE results are used to support FTC, which is realized by adjusting the controller in real time to accommodate to the influence of faults and maintain the performance of the controlled systems with actuator faults.

In conclusion, this paper aims at dealing with the FE and FTC task under the ILC framework for the systems with actuator faults. Q-learning algorithm is employed for FE task to counteract the negative effects of faults. The design of FTC considers the NOILC framework. This paper accomplishes the FTC by reconfiguring the controller based on FE results in real time to maintain the control performance. The main contributions of this paper are listed as follows:

(1) A scheme is developed for the FE and FTC task under the ILC framework for discrete-time multi-input multi-output (MIMO) systems with actuator faults.

(2) Q-learning algorithm is introduced to estimate the actuator faults. In the meantime, the FE results provide supports for FTC in order to maintain the reliability and performance of the system with actuator faults.

(3) A convergence condition of fault tolerant ILC update law is derived and proved, which increases the theoretical reliability of the proposed scheme.

This paper is structured as follows. In Section 2, the preliminary knowledge about ILC and Q-learning is introduced. In Section 3, the problem of the fault tolerant ILC design is formulated. In Section 4, the proposed algorithm is presented, which includes the design of FTC and FE. Meanwhile, the analysis of convergence property is provided. In Section 5, a numerical

simulation of a mobile robot is presented to verify the effectiveness of the proposed algorithm. Finally in Section 6, the conclusions are given.

Notation: $\mathbb{N}$ is the set of non-negative integers. $\mathbb{R}^n$ denotes the set of n-dimensional real vectors. $\mathbb{R}^{n\times m}$ denotes the sets of $n \times m$ matrices.$\|\cdot\|$ is the Euclidean norm of a vector. $l_2^n[a,b]$ denotes the space of $\mathbb{R}^n$ valued Lebesgue square-summable sequences defined on an interval $[a,b]$. $\|\cdot\|_R^2$ is the induced norm of matrix defined in Hilbert space with weighting matrix R. $\hat{\delta}_{ik}(t)$ denotes the estimated effectiveness factor of ith actuator at tth sample time in the kth trial. $A^\dagger$ denotes the pseudo-inverse of matrix A.

2. Preliminaries

This section first introduces the basic framework and conventional objective of ILC. Then, the main components of RL and the procedures of Q-learning is presented.

2.1. Knowledge for ILC

Consider the state space form of a discrete-time, ℓ-input, m-output linear system operating on the time interval $t \in [0, N]$ as

$$\begin{cases} x_k(t+1) = Ax_k(t) + Bu_k(t), \\ \quad y_k(t) = Cx_k(t), \end{cases} \tag{1}$$

where the subscript $k \in \mathbb{N}$ denotes the trial number index. $x_k(t) \in \mathbb{R}^n$, $u_k(t) \in \mathbb{R}^\ell$ and $y_k(t) \in \mathbb{R}^m$ represent the state, input and output respectively. Meanwhile, t is the time index with N denoting the total sample number in a trial. The system matrices A, B and C have compatible dimensions, and $CB \neq 0$ needs to be guaranteed to ensure the controllability of system.

Without loss of generality, the state $x_k(t)$ should be reset to an identical initial value x_0 at the end of each trial.

Reformulate the system (1) into a lifted system framework in the trial domain, i.e.,

$$y_k = Gu_k + d, \tag{2}$$

where the input signals $u_k \in l_2^\ell[0, N-1]$ and output signals $y_k \in l_2^\ell[1, N]$ are denoted as

$$u_k = [u_k^T(0), u_k^T(1), \ldots, u_k^T(N-1)]^T, \tag{3}$$

$$y_k = [y_k^T(1), y_k^T(2), \ldots, y_k^T(N)]^T. \tag{4}$$

The matrices G and d representing the system model and the initial state response are denoted as

$$G = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ CA^2B & CAB & CB & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ CA^{N-1}B & CA^{N-2}B & CA^{N-3}B & \cdots & CB \end{bmatrix}, \tag{5}$$

$$d = [(CA)^T, (CA^2)^T, \ldots, (CA^N)^T]^T x_0. \tag{6}$$

The input Hilbert space $l_2^\ell[0, N-1]$ and the output Hilbert space $l_2^\ell[1, N]$ are equipped with inner products and associated induced norms as

$$\langle u, v\rangle_R = u^T R v, \ \|u\|_R = \sqrt{\langle u, u\rangle_R}, \tag{7}$$

$$\langle y, z\rangle_Q = y^T Q z, \ \|y\|_Q = \sqrt{\langle y, y\rangle_R}, \tag{8}$$

where $R \in \mathbb{R}^{\ell N \times \ell N}$ and $Q \in \mathbb{R}^{mN \times mN}$ are the real positive definite weighting matrices.

The conventional objective of ILC is to revise the input signal u_k during the repetitive task in an iterative manner. Eventually, the input u_k converges to a unique value u_d. The output y_k tracks on the desired reference profile r, which means the tracking error e_k converging to zero, i.e.,

$$\lim_{k\to\infty} u_k = u_d, \quad \lim_{k\to\infty} e_k = 0. \tag{9}$$

The tracking error at kth trial is denoted as

$$e_k = r - y_k. \tag{10}$$

In addition, based on (2), the desired output y_d, which tracks on the reference profile r, can be defined by the desired input u_d as

$$r = y_d, \tag{11}$$

$$y_d = Gu_d + d. \tag{12}$$

Based on the lifted technique, the controlled system is redescribed in form of super vectors in the trial domain, which simplifies the computation in the time domain and provides more convenience for introducing the optimal technology into the ILC framework. General optimization methods need the exact model of the controlled system, but faults could bring uncertainties to a definite system dynamics. Therefore, Q-learning is introduced to handle the negative effects of faults in real time.

2.2. Knowledge for Q-learning

As is shown in Figure 1, the RL problem consists of agent, environment, states, actions and rewards. The agent is the learner and the decision maker. The environment is the thing interacted with the agent, which comprises

everything outside the agent. With the interaction continuing, for each step $t \in 0, 1, 2, 3, \ldots$, the agent selects action $\mathcal{A}_t$ in the state $\mathcal{S}_t$ at t step. Then, the agent transfers to the state $\mathcal{S}_{t+1}$ and receives the reward $\mathcal{R}_{t+1}$ from the environment at $(t+1)$ step. Finally, the agent takes actions by learning the optimal policy to maximize the cumulative rewards or achieve other objectives such as reaching the desired end point as fast as possible. RL tasks are commonly described by the Markov Decision Process (MDP). A typical finite MDP is represented by the quintuple $(\mathcal{S}, \mathcal{A}, p, \mathcal{R}, \gamma)$, which is shown as follows:

- $\mathcal{S}$ is the set of states, where the state $s \in \mathcal{S}$ and the state at t step is denoted as $\mathcal{S}_t \in \mathcal{S}$.

- $\mathcal{A}$ is the set of actions, where the action selected on the basis of the state s at t step is denoted as $\mathcal{A}_t \in \mathcal{A}(s)$.

- p is the state transition function, which is the probability of the state $s \in \mathcal{S}$ transfering to the next state $s' \in \mathcal{S}$.

$$p(s', r|s, a) = Pr\left\{\mathcal{S}_{t+1} = s', \mathcal{R}_{t+1} = r|\mathcal{S}_t = s, \mathcal{A}_t = a\right\} \tag{13}$$

  where p is a deterministic function with four parameters, where $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$.

- $\mathcal{R}$ is the reward function, the instant reward at $(t+1)$ step is denoted as $\mathcal{R}_{t+1} \in \mathcal{R} \subset \mathbb{R}$.

- $\gamma \in [0, 1]$ is the discount factor, determining the present value of future rewards.

Through the quintuple, the following concepts can be represented. The policy π determines the probability distribution of different actions. The Q-learning algorithm aims at obtaining the optimal strategy by iteratively updating the state-action value function $Q_\pi(s,a)$, which is denoted as the reward expectation when adopting the policy π in the state $s \in \mathcal{S}$ of taking action $a \in \mathcal{A}$ as follows

$$Q_\pi(s,a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{t+k+1} | \mathcal{S}_t = s, \mathcal{A}_t = a\right]. \tag{14}$$

Generally, $Q_\pi(s,a)$ is adopted for value evaluation. Hence, the optimal policy is defined as π^*, which maximizes the action-value function $Q_\pi(s,a)$:

$$\pi^* = \arg\max_{a \in \mathcal{A}} \ Q_\pi(s,a). \tag{15}$$

The objective of Q-learning is the optimal action-value function $Q(\mathcal{S}_t, \mathcal{A}_t)$, which is updated by

$$Q(\mathcal{S}_t, \mathcal{A}_t) \leftarrow Q(\mathcal{S}_t, \mathcal{A}_t) + \alpha \left[\mathcal{R}_{t+1} + \gamma \max_a Q(\mathcal{S}_t, a) - Q(\mathcal{S}_t, \mathcal{A}_t)\right]. \tag{16}$$

where $\alpha \in [0,1]$ is the learning rate, which is usually a small scalar to ensure that the result does not oscillate at the expense of convergence speed. Through Q-learning, the optimal policy π^* can be extracted by (15) via the optimal action-value function $Q(\mathcal{S}_t, \mathcal{A}_t)$ without information of environment or the state transition function p.

This paper regards the unknown faults as the unknown environment and the fault estimator as the agent. Q-learning algorithm is employed to estimate the varying faults. In addition, the estimated fault information is provided for designed FTC to reconfigure the controller in real time. Finally, the performance of system with actuator faults can be guaranteed.

3. Problem formulation

This section firstly introduces the system dynamics with actuator faults under ILC framework. Then, the fault tolerant ILC problem definition is given.

3.1. System dynamics with actuator faults

Faults are usually divided into actuator, sensor and component faults. Due to its repetitive property of the demand on the control actuator, ILC is sensitive to actuator faults. Therefore, this paper focuses on addressing the influence of actuator faults under ILC framework. Considering the practical operating condition, the actuator faults vary along with the time and trials. In that case, the system with actuator faults of (1) can be defined as

$$\begin{cases} x_k(t+1) = Ax_k(t) + B\delta_k(t)u_k(t), x_k(0) = x_0, \\ \quad y_k(t) = Cx_k(t), \end{cases} \tag{17}$$

where the input signal $u_k(t)$ changes into $u_k^F(t) = \delta_k(t)u_k(t)$ and $u_{i,k}^F(t) = \delta_{i,k}(t)u_k(t)$, i.e.,

$$u_k^F(t) = \left[u_{1,k}^F(t), u_{2,k}^F(t), \cdots, u_{m,k}^F(t)\right]^T, \tag{18}$$

which means there are some losses in the actuator driving power.

The fault matrix $\delta_k(t)$ represents the effectiveness factor of actuator, which is denoted as

$$\delta_k(t) = diag\left\{\delta_{1,k}(t), \delta_{2,k}(t), \ldots, \delta_{m,k}(t)\right\} \tag{19}$$

and the elements inside have ranges as follows

$$0 \leqslant \underline{\delta}_i \leqslant \delta_{i,k}(t) \leqslant \overline{\delta}_i, i = \{1, 2, \cdots, m\}. \tag{20}$$

Similarly, the estimated fault mode is defined as

$$\hat{\delta}_k(t) = diag\left\{\hat{\delta}_{1,k}(t), \hat{\delta}_{2,k}(t), \ldots, \hat{\delta}_{m,k}(t)\right\}, \tag{21}$$

$$0 \leqslant \underline{\delta}_i \leqslant \hat{\delta}_{i,k}(t) \leqslant \overline{\delta}_i, i = \{1, 2, \cdots, m\}. \tag{22}$$

The fault modes of the bound are denoted as

$$\underline{\delta} = diag\left\{\underline{\delta}_1, \underline{\delta}_2, \ldots, \underline{\delta}_m\right\}, \tag{23}$$

$$\overline{\delta} = diag\left\{\overline{\delta}_1, \overline{\delta}_2, \ldots, \overline{\delta}_m\right\}. \tag{24}$$

The minimum value and the maximum value of the above fault modes are defined as

$$\underline{\delta}_{min} = \underset{1 \leqslant i \leqslant m}{min}\ \underline{\delta}_i, \tag{25}$$

$$\overline{\delta}_{max} = \underset{1 \leqslant i \leqslant m}{max}\ \overline{\delta}_i. \tag{26}$$

Note that both fault scalars $\underline{\delta}_i (0 \leqslant \underline{\delta}_i \leqslant 1)$ and $\overline{\delta}_i (\overline{\delta}_i \geqslant 1)$ are assumed to be known, which means that the entries $\delta_{i,k}(t)$ of the fault matrix $\delta_k(t)$ and the entries $\hat{\delta}_{i,k}$ of the fault matrix $\hat{\delta}_k(t)$ are unknown but vary in a known range. In particular, $\delta_i = 0$ represents the ith actuator completely failed. Otherwise, $\delta_i = 1$ represents the ith actuator operates normally. $0 < \delta_i < 1$ represents the residual driving power of the ith actuator under fault. $\delta_i > 1$ represents the overmuch driving power of the ith actuator under fault.

Adopting the lifted technique as (2) in the trial domain, controlled system (17) with actuator faults can be reformulated into

$$y_k = G\delta_k u_k + d, \tag{27}$$

where G and d are in accordance with (5) and (6), and δ_k is a diagonal matrix defined as

$$\delta_k = \begin{bmatrix} \delta_k(0) & & & & \\ & \delta_k(1) & & & \\ & & \delta_k(2) & & \\ & & & \ddots & \\ & & & & \delta_k(N-1) \end{bmatrix}. \tag{28}$$

The lifted nominal model of the lifted system (27) with estimated actuator faults can be described as

$$\hat{y}_k = G\hat{\delta}_k u_k + d, \tag{29}$$

where the estimated $\hat{\delta}_k$ is defined as

$$\hat{\delta}_k = \begin{bmatrix} \hat{\delta}_k(0) & & & & \\ & \hat{\delta}_k(1) & & & \\ & & \hat{\delta}_k(2) & & \\ & & & \ddots & \\ & & & & \hat{\delta}_k(N-1) \end{bmatrix}. \tag{30}$$

The corresponding tracking error at kth trial based on (10) and (27) is defined as

$$e_k = r - G\delta_k u_k - d. \tag{31}$$

The numerically computed error at kth trial using the nominal model (29) is defined as

$$\hat{e}_k = r - G\hat{\delta}_k u_k - d. \tag{32}$$

3.2. Fault tolerant ILC problem definition

The fault tolerant ILC design problem of systems with actuator faults considering in this paper is to design an fault tolerant ILC update law

$$u_{k+1} = f(u_k, e_k, \hat{\delta}_k, \hat{\delta}_{k+1}) \tag{33}$$

to actively be reconfigured through the estimated fault information in each trial. Note that the estimated fault information is provided by the Q-learning based FE process, which is described in section 4.2. With the update procedure going on, the input signal u_k is iteratively modified in order to enable the output y_k to maintain good tracking performance with actuator faults, i.e.,

$$\lim_{k\to\infty} \|e_{k+1}\| \leqslant \epsilon_e, \tag{34}$$

where ϵ_e, a relatively small constant, is the upper bound of the tracking error e_k. Note that due to the existence of faults which is not taken in to consideration by the classic ILC task, the tracking error e_k can not converge to zero. It makes sense in practical condition that the tracking error e_k obtains a bounded convergence.

Remark 1. The fault tolerant ILC strikes a balance between reliability of fault tolerance and tracking performance of ILC, which means that the objective of the proposed scheme is to maintain relatively high tracking performance of the controlled systems with the existence of faults. Therefore, the design objective does not require the tracking error converging to zero. In a sense, the tracking error only need to converge to a bounded range in order to maintain the reliability and the tracking performance.

4. Design of FE and FTC under ILC framework

This section firstly introduces NOILC framework to design a fault tolerant ILC update law with the purpose of dealing with the problem in subsection 3.2. Then the specific settings of Q-learning based FE is given. Next, the detail of the proposed scheme for FE and FTC under the ILC framework is described. Finally, the convergence property of the proposed ILC update law is analyzed.

4.1. Fault tolerant ILC design

To deal with the problem presented in subsection 3.2, the norm-optimal ILC framework is considered in this paper to optimize the multi-objective performance criterion at each trial [24]. The performance criterion is defined as

$$J_{k+1} \triangleq \left\| r - G\hat{\delta}_{k+1}u_{k+1} - d \right\|_Q^2 + \|u_{k+1} - u_k\|_R^2 , \tag{35}$$

in which the performance criterion is made of two components, the numerically computed tracking error and the input change between adjacent trials. Minimizing the tracking error aims at tracking on the reference to obtain the essential ILC objective. Meanwhile, smoothing the input index intends to introduce robustness into the algorithm. The symmetric positive weighting matrices Q and R denote the priority of error deduction and robustness during the optimization procedure.

The optimal control input u_{k+1} is derived from minimizing the performance criterion

$$u_{k+1} = \arg\min_{u_{k+1}} \{J_{k+1}\} . \tag{36}$$

Theorem 1 (ILC Update Law)**.** *The solution followed from the necessary condition of optimality (36) is given by*

$$u_{k+1} = L^u_{k+1} u_k + L^e_{k+1} e_k, \tag{37}$$

where the operators L^u_{k+1} *and* L^e_{k+1} *are defined as*

$$L^u_{k+1} = \left(\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_{k+1} + R\right)^{-1} \left(\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_k + R\right), \tag{38}$$

$$L^e_{k+1} = \left(\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_{k+1} + R\right)^{-1} \hat{\delta}^T_{k+1} G^T Q. \tag{39}$$

Proof. Based on the induced norms (7) and (8), substitute (31) into (35) to give

$$\begin{aligned} J_{k+1} = & \left(u_{k+1} - u_k\right)^T R \left(u_{k+1} - u_k\right) \\ & + \left(r - G\hat{\delta}_{k+1} u_{k+1} - d\right)^T Q \left(r - G\hat{\delta}_{k+1} u_{k+1} - d\right). \end{aligned} \tag{40}$$

Then, differentiate the performance criterion with respect to u_{k+1} and let $\partial J_{k+1}/\partial u_{k+1} = 0$,which yields

$$R\left(u_{k+1} - u_k\right) - \hat{\delta}^T_{k+1} G^T Q \left(\hat{e}_k + G\hat{\delta}_k u_k + d - G\hat{\delta}_{k+1} u_{k+1} - d\right) = 0. \tag{41}$$

Merging the similar terms gives rise to

$$\begin{aligned} & \left(\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_{k+1} + R\right) u_{k+1} \\ & = \left(\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_k + R\right) u_k + \hat{\delta}^T_{k+1} G^T Q \hat{e}_k. \end{aligned} \tag{42}$$

Since the matrices $\hat{\delta}_{k+1}$, G and R are positive definite, $\left(\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_{k+1} + R\right)$ is invertible. After replacing the numerical computed tracking error $\hat{e}_k$ with the measured tracking error e_k, the ILC update law (37) can be derived, which completes the proof.

The numerical computed tracking error $\hat{e}_k$ depends on the estimated faults information $\hat{\delta}_k$ and lacks real fault information δ_k, which will result in poor robustness of the system. Therefore, in order to improve the robustness, the measured tracking error e_k is employed to introduce real fault information into the ILC update law.

Remark 2. There is no strict rule for selection of the weighting matrices Q and R. However, the effect of the choice about Q and R has been commented in [25]. Generally, increasing the value of Q or decreasing the value of R will increase the speed of convergence but decrease the robustness, which will be verified by the simulation in Section 5.

4.2. FE design using Q-learning

Fault estimation aims at providing fault information $\hat{\delta}_k$ and $\hat{\delta}_{k+1}$ for fault tolerant ILC update law (37). Since the upper and lower bounds of the fault modes are known, searching for the value of the unknown fault could be regarded as searching for the terminal point in a grid world having fixed range. Therefore, this paper employs the Q-learning algorithm, which is frequently used to solve similar cases. The FE task is to estimate $\delta_{k+1}(t)$ in kth trial at time t.

Next, the concepts presented in subsection 2.2 are successively given the specific meaning in the FE task. In the Q-learning based FE process, assume the agent to be the fault estimator and assume the environment E to be the controlled system. Other Q-learning related settings are as follows:

State. Denote the state space as $\mathcal{S}^{n_\delta \times n_\delta}$, where the state $s \in \mathcal{S}^{n_\delta \times n_\delta}$, and n_δ depends on the estimation precision of $\hat{\delta}_{i,k}(t)$. The state can be defined

as

$$s = \left[\hat{\delta}_{1,k}(t), \hat{\delta}_{2,k}(t), , \cdots, \hat{\delta}_{m,k}(t)\right]. \tag{43}$$

Action. Denote the action space $\mathcal{A}^{m\times m}$, where $a \in \mathcal{A}^{m\times m}$. The action can be described as

$$a = \left[\Delta\hat{\delta}_{1,k}(t), \Delta\hat{\delta}_{2,k}(t), \cdots, \Delta\hat{\delta}_{m,k}(t)\right]. \tag{44}$$

State transition formula. The state transition formula is followed:

$$\begin{aligned} s' &= s + a \\ &= \left[\hat{\delta}_{1,k}(t) + \Delta\hat{\delta}_{1,k}(t), \hat{\delta}_{2,k}(t) + \Delta\hat{\delta}_{2,k}(t), \cdots, \hat{\delta}_{m,k}(t) + \Delta\hat{\delta}_{m,k}(t)\right] \end{aligned} \tag{45}$$

Policy. The policy for action selection is ϵ-greedy policy:

$$\mathcal{A} = \pi(s) = \begin{cases} \arg\max\limits_{a} Q(a), & \text{if } p < (1-\epsilon) \\ \text{random action}, & \text{if } p \leq \epsilon \end{cases} \tag{46}$$

where ϵ is the greedy probability. The controller has probability of ϵ to randomly choose an action and has probability of $(1-\epsilon)$ to choose the action returning the maximum value according to the current Q-table as (15).

Q-table. As what has been mentioned in (16), the action-value function $Q^{\pi}(s,a)$ is updated by

$$Q^{\pi}(s,a) \leftarrow Q^{\pi}(s,a) + \alpha\left[\mathcal{R}^{a}_{s\to s'} + \gamma \max_{a'} Q^{\pi}(s',a') - Q^{\pi}(s,a)\right]. \tag{47}$$

Reward. Aiming at accurately estimate the effectiveness factor of actuator $\delta_k(t)$, design a loss function as

$$\mathcal{L} = \left\| x_k(t+1) - Ax_k(t) - B\hat{\delta}_k(t)u_k(t) \right\|^2. \tag{48}$$

According to the loss function, the reward can be designed as

$$\mathcal{R} = \begin{cases} \mathcal{R}_c \ \ , if \ \mathcal{L} \leqslant \varepsilon_{\mathcal{L}} \\ -1, if \ \mathcal{L} > \varepsilon_{\mathcal{L}} \end{cases} \tag{49}$$

where $\mathcal{R}_c = n_\delta \times n_\delta$ is a constant related with the number of the states and the loss function threshold $\varepsilon_{\mathcal{L}} > 0$ is a small scalar representing the accuracy of the Q-learning based estimation.

Remark 3. The advantage of the FE methods proposed in this paper is that the Q-learning algorithm is introduced to disassemble the continually changing fault estimation task into sub-tasks of fault values estimation at each sample time. Therefore, the real-time support can be provided for FTC procedure.

Remark 4. Due to the causality of time lapsing, the estimated $\hat{\delta}_{k+1}$ is actually the direct approximation of δ_k. However the estimated $\hat{\delta}_{k+1}$ can always keep up with the change of actual δ_k in each trial. Therefore, the fault tolerant ILC based on FE results can still obtain good control performance.

4.3. Algorithm description

In this subsection, the detail of the proposed scheme is described in Algorithm 1 and Algorithm 2, where Algorithm 1 presents the procedure of Fault tolerant ILC and Algorithm 2 presents the procedure of FE. In addition, the operation procedure is illustrated in Figure 2. As is shown, in kth trial, firstly FE is realized by estimating the effectiveness factor of actuator $\delta_{k+1}(t)$ during sample time $t \in [0, N-1]$ through Q-learning algorithm. Then, FTC

Algorithm 1 Fault tolerant ILC

Input: The total sample time N; the total trial number k_{max}; the tracking reference r; The system matrices A, B and C; The weighting matrices Q and R.

1: **Initialization:** The initial input u_0; the initial state x_0; the initial estimated effectiveness factor of actuator $\hat{\delta}_0(0)$.
2: Run u_0 to the system (2) and obtain y_0. Then, compute e_0 and obtain u_1 through the designed ILC update law (37).
3: **for** $k = 1, 2, \cdots, k_{max}$
4: **for** $i = 1, 2, \cdots, N$
5: Employ Q-learning based FE Algorithm 2 to obtain $\hat{\delta}_{k+1}(i)$.
6: **end for**
7: Update the next trial input u_{k+1} through the designed ILC update law (37) by the $\hat{\delta}_k$, $\hat{\delta}_{k+1}$, u_k and e_k.
8: Obtain the next trial output y_{k+1} and error e_{k+1}.
9: **end for**

is accomplished by using the estimated $\hat{\delta}_{k+1}(t)$ to actively reconfigure the ILC update law.

Due to the real-time adjustment to the controller, the negative effects of faults can be counteracted and good control performance can be maintained for systems with actuator faults.

4.4. Analysis on convergence property

In this subsection, the analysis on convergence property of ILC update law (37) is explained. Since the faults and the estimated faults are uncertain

Algorithm 2 FE using Q-learning

Input: The state space $\mathcal{S}$, where the state $s \in \mathcal{S}$ and $s = \left[\hat{\delta}_{1,k}(t), \hat{\delta}_{2,k}(t), , \cdots, \hat{\delta}_{m,k}(t)\right]$; the action space $\mathcal{A}$, where the action $a \in \mathcal{A}$ and $a = \left[\Delta\hat{\delta}_{1,k}(t), \Delta\hat{\delta}_{2,k}(t), \cdots, \Delta\hat{\delta}_{m,k}(t)\right]$; the learning rate α; the discount factor γ; the greedy probability ϵ; the loss function threshold $\varepsilon_{\mathcal{L}}$; the state x_k and input u_k.

1: Initialize Q-table and the initial state s_0.
2: Choose a_0 from s_0 using policy ϵ-greedy.
3: **while** $\mathcal{L} > \varepsilon_{\mathcal{L}}$:
4: Take action a, obtain the corresponding $\mathcal{R}$, s'.
5: Choose a' from s' using policy ϵ-greedy.
6: Update the action-value function $Q^{\pi}(s, a)$ through (47) .
7: $s \leftarrow s'$, $a \leftarrow a'$.
8: **end while**

and they both have the upper bound and lower bound, it is not necessary to discuss the convergence property of the FE in the analysis. Therefore, considering the faults and the estimated faults as bounded matrices varying with the trail k, then the convergence of the ILC update law (37) can be analyzed. The convergence condition and the proof are given, where the following lemma will be used in the process of proof.

Lemma 1. *For any given matrix $A \in \mathbb{R}^{m\times n}$ which satisfies*

$$\rho(A) < 1, \tag{50}$$

where $\rho(A)$ is the spectral radius of the matrix A. Then, there exists at least

one type of matrix norm $\|A\|_S$ *such that*

$$\lim_{k\to\infty} \|A\|_S^k = 0. \tag{51}$$

PROOF. Please refer to Appendix A in [26] for the detailed proof.

Theorem 2 (Convergence Property)**.** *Adopt the ILC update law (37) to the system (17) with actuator faults, Then, if the condition*

$$\left\|\delta_{k+1} L_{k+1}^u \delta_k^\dagger - \delta_{k+1} L_{k+1}^e G\right\| \leqslant \rho < 1 \tag{52}$$

is satisfied, the norm of tracking error can obtain bounded convergence, i.e.,

$$\lim_{k\to\infty} \|e_{k+1}\| \leqslant \frac{b_u c}{1-\rho}, \tag{53}$$

where the positive constant $b_u = b\,\|u_d\|$, $c = \|G\|$. *In addition,* b *is a positive scalar defined as*

$$\left\|I - \hat{\delta}_{k+1} L_{k+1}^u \hat{\delta}_k^\dagger\right\| \leqslant b < \frac{\overline{\delta}_{max}}{\underline{\delta}_{min}} \left(\overline{\delta}_{max}^2 \left\|R^{-1}\right\| \left\|G^T\right\| \|Q\| \|G\| + 1\right) + 1. \tag{54}$$

PROOF. According to (11), (12) and (31) , the tracking error of kth trial can be reformulated as

$$\begin{aligned} e_k &= y_d - y_k \\ &= Gu_d + d - G\delta_k u_k - d \\ &= G(u_d - \delta_k u_k). \end{aligned} \tag{55}$$

Define the input error Δu_k as

$$\Delta u_k = u_d - \delta_k u_k. \tag{56}$$

Then, (55) can be reformulated as

$$e_k = G\Delta u_k. \tag{57}$$

Based on (56) and the ILC update law (37), the input error at $(k+1)$th trial can be obtained as

$$\begin{aligned}
\Delta u_{k+1} &= u_d - \delta_{k+1} u_{k+1} \\
&= u_d - \delta_{k+1} L^u_{k+1} u_k - \delta_{k+1} L^e_{k+1} e_k \\
&= u_d - \delta_{k+1} L^u_{k+1} (\delta^\dagger_k u_d - \delta^\dagger_k \Delta u_k) - \delta_{k+1} L^e_{k+1} G \Delta u_k \\
&= \left[\delta_{k+1} L^u_{k+1} \delta^\dagger_k - \delta_{k+1} L^e_{k+1} G\right] \Delta u_k + \left[I - \delta_{k+1} L^u_{k+1} \delta^\dagger_k\right] u_d.
\end{aligned} \tag{58}$$

Taking forms of norm on both side of the formula, then there exists

$$\begin{aligned}
\|\Delta u_{k+1}\| \leqslant &\left\|\delta_{k+1} L^u_{k+1} \delta^\dagger_k - \delta_{k+1} L^e_{k+1} G\right\| \|\Delta u_k\| \\
&+ \left\|I - \delta_{k+1} L^u_{k+1} \delta^\dagger_k\right\| \|u_d\|.
\end{aligned} \tag{59}$$

Next, prove that $\left\|I - \delta_{k+1} L^u_{k+1} \delta^\dagger_k\right\|$ has an upper bound. According to the compatibility and the triangle inequality of norm, there exists an inequality

$$\left\|I - \delta_{k+1} L^u_{k+1} \delta^\dagger_k\right\| \leqslant \|\delta_{k+1}\| \left\|L^u_{k+1}\right\| \left\|\delta^\dagger_k\right\| + 1. \tag{60}$$

Since δ_k and $\hat{\delta}_k$ are diagonal matrices, the upper bound of $\|\delta_k\|$, $\left\|\delta^\dagger_k\right\|$ and $\left\|\hat{\delta}_k\right\|$ can be obtained

$$\|\delta_k\| \leqslant \overline{\delta}_{max}, \ \left\|\delta^\dagger_k\right\| \leqslant \frac{1}{\underline{\delta}_{min}}, \ \left\|\hat{\delta}_k\right\| \leqslant \overline{\delta}_{max}. \tag{61}$$

According to (38), (61), there exists

$$\left\|L^u_{k+1}\right\| = \left\|\left(R^{-1}\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_{k+1} + I\right)^{-1} \left(R^{-1}\hat{\delta}^T_{k+1} G^T Q G \hat{\delta}_k + I\right)\right\|. \tag{62}$$

Because the matrices R^{-1}, $\hat{\delta}_{k+1}$, G and Q are all positive and definite, it is cleared that $\left\|\left(R^{-1}\hat{\delta}_{k+1}^{T}G^{T}QG\hat{\delta}_{k+1}+I\right)^{-1}\right\|<1$. Therefore, based on (61), (62) can be reformulated as

$$\begin{aligned}\left\|L_{k+1}^{u}\right\| &< \left\|R^{-1}\hat{\delta}_{k+1}^{T}G^{T}QG\hat{\delta}_{k}+I\right\| \\ &< \left\|R^{-1}\right\|\left\|\hat{\delta}_{k+1}^{T}\right\|\left\|G^{T}\right\|\|Q\|\|G\|\left\|\hat{\delta}_{k}\right\|+1 \\ &< \overline{\delta}_{max}^{2}\left\|R^{-1}\right\|\left\|G^{T}\right\|\|Q\|\|G\|+1. \end{aligned} \tag{63}$$

Substitute (61) into (60) to obtain

$$\left\|I-\hat{\delta}_{k+1}L_{k+1}^{u}\hat{\delta}_{k}^{\dagger}\right\|<\frac{\overline{\delta}_{max}}{\underline{\delta}_{min}}\left(\overline{\delta}_{max}^{2}\left\|R^{-1}\right\|\left\|G^{T}\right\|\|Q\|\|G\|+1\right)+1. \tag{64}$$

Then, define a positive scalar b as

$$\left\|I-\hat{\delta}_{k+1}L_{k+1}^{u}\hat{\delta}_{k}^{\dagger}\right\|\leqslant b<\frac{\overline{\delta}_{max}}{\underline{\delta}_{min}}\left(\overline{\delta}_{max}^{2}\left\|R^{-1}\right\|\left\|G^{T}\right\|\|Q\|\|G\|+1\right)+1. \tag{65}$$

Based on (65), (59) can be represented as

$$\|\Delta u_{k+1}\|\leqslant\left\|\delta_{k+1}L_{k+1}^{u}\delta_{k}^{\dagger}-\delta_{k+1}L_{k+1}^{e}G\right\|\|\Delta u_{k}\|+b\|u_{d}\|. \tag{66}$$

Define $b_u = b\|u_d\|$. After kth trials, there exists

$$\begin{aligned}\|\Delta u_{k+1}\|\leqslant&\left\|\delta_{k+1}L_{k+1}^{u}\delta_{k}^{\dagger}-\delta_{k+1}L_{k+1}^{e}G\right\|^{k}\|\Delta u_{0}\| \\ &+\frac{1-\left\|\delta_{k+1}L_{k+1}^{u}\delta_{k}^{\dagger}-\delta_{k+1}L_{k+1}^{e}G\right\|^{k}}{1-\left\|\delta_{k+1}L_{k+1}^{u}\delta_{k}^{\dagger}-\delta_{k+1}L_{k+1}^{e}G\right\|}b_{u}.\end{aligned} \tag{67}$$

Based on Lemma 1, if the condition (52) holds, it can be derived that $\lim\limits_{k\to\infty}\left\|\delta_{k+1}L_{k+1}^{u}\delta_{k}^{\dagger}-\delta_{k+1}L_{k+1}^{e}G\right\|^{k}=0$ when the trial number $k\to\infty$, the norm of tracking error can be represented as

$$\lim_{k\to\infty}\Delta u_{k+1}\leqslant\frac{b_{u}}{1-\rho}. \tag{68}$$

Combine (57) and (68) to give

$$\begin{aligned} \lim_{k\to\infty} e_{k+1} &= \lim_{k\to\infty} G\Delta u_{k+1} \\ &\leqslant \frac{b_u \|G\|}{1-\rho}. \end{aligned} \tag{69}$$

Finally, let $c = \|G\|$ to obtain

$$\lim_{k\to\infty} e_{k+1} \leqslant \frac{b_u c}{1-\rho}, \tag{70}$$

which completes the proof.

Therefore, if the condition (52) is satisfied, the norm of tracking error can obtain bounded convergence, which corresponds to what is mentioned in subsection 3.2. The analysis of the convergence property provides theoretical reliability for the proposed algorithm.

Remark 5. Due to the influence of the faults, the tracking error can not decrease monotonically along the trials but oscillate with the varying faults in a bounded range after the convergence of the first few trials, which can be intuitively seen through the simulation in Section 5.

5. Simulation verification

To verify the performance of the proposed algorithm, a numerical model of the mobile robot with two independent driving wheels is employed [27]. Decoupling and then discretizing the MIMO system with the purpose of controlling the linear velocity v and the azimuth ϕ individually by the driving voltage u_v and u_ϕ, in which way the mobile robot can make rigid move in the absolute coordinate system $o-xy$. The results illustrate the effectiveness of

the proposed algorithm and the comparisons with other algorithms implying its advantages.

5.1. Simulation specification

The double-input and triple-output mobile robot system with two independent wheels is shown in Figure 3. Define the state variable as $x = \begin{bmatrix} v & \phi & \dot{\phi} \end{bmatrix}^T$, the input variable as $u = [u_r \quad u_l]^T$, and the output variable as $y = [v \quad \phi]^T$. Then the state space model matrices are expressed as

$$A = \begin{bmatrix} a_1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & a_2 \end{bmatrix}, B = \begin{bmatrix} b_1 & b_1 \\ 0 & 0 \\ b_2 & -b_2 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

In addition, the a_1, a_2, b_1 and b_2 are denoted as

$$\begin{aligned} a_1 &= -\frac{2c}{Mr^2 + 2I_w}, \; a_2 = -\frac{2cl^2}{I_v r^2 + 2I_w l^2}, \\ b_1 &= \quad \frac{kr}{Mr^2 + 2I_w}, \; b_2 = \quad \frac{krl}{I_v r^2 + 2I_w l^2}, \end{aligned}$$

where the parameters inside are defined as Table 1.

The model of the mobile robot is a linear coupling system. In order to be controlled as expectation, the system should be decoupled firstly by

$$\begin{bmatrix} u_r \\ u_l \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} u_v \\ u_\phi \end{bmatrix}, \tag{71}$$

where the decoupled input variable is defined as $\tilde{u} = \begin{bmatrix} u_v & u_\phi \end{bmatrix}^T$. Meanwhile, u_v is the driving voltage to directly control the linear velocity of the robot and u_ϕ is also the driving power to directly control the azimuth of the robot.

Eventually, the discrete-time state space form is defined as

$$\begin{cases} x_k(t+1) = \tilde{A}x_k(t) + \tilde{B}\tilde{u}_k(t) \\ y_k(t) = \tilde{C}x_k(t), \end{cases} \tag{72}$$

where the state space model matrices are derived as

$$\tilde{A} = \begin{bmatrix} 0.9975 & 0 & 0 \\ 0 & 1 & 0.0499 \\ 0 & 0 & 0.9956 \end{bmatrix}, \tilde{B} = \begin{bmatrix} 0.0248 & 0 \\ 0 & 0.0037 \\ 0 & 0.1483 \end{bmatrix},$$
$$\tilde{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

During the discretization, consider using a zero-order holder and set the sample time T_s as 0.01s. In the meantime, set the trial length T as 2s, i.e., the number of sampling points in each trial $N = 40$.

The ILC task is to track on the the desired linear velocity and the desired azimuth of the robot

$$v_d = 2 \text{ m/s}, \ \phi_d = \pi t \text{ rad}, \tag{73}$$

which means the desired tracking trajectory is a circle.

The partial degradation and wear during the repeated control operations may result in faults. Therefore, considering the following actuator faults in the simulation, in which the effectiveness factor of actuator is expressed as

$$\delta_k(t) = \begin{bmatrix} \delta_{1,k} & \\ & \delta_{2,k}(t) \end{bmatrix}, \tag{74}$$

where the entries inside are set as

$$\delta_{1,k} = 0.15sin(\pi k/10 - \pi/2) + 0.7, \tag{75}$$

$$\delta_{2,k}(t) = 0.1sin(\pi k/8 - \pi/2) + 0.75 + 0.1sin(2\pi t), t \in [0, N-1]. \tag{76}$$

and the entries inside have ranges, i.e.,

$$0.55 \leqslant \underline{\delta}_i \leqslant \delta_{i,k}(t) \leqslant \overline{\delta}_i \leqslant 0.95, i = 1, 2 \tag{77}$$

$$0.55 \leqslant \underline{\delta}_i \leqslant \hat{\delta}_{i,k}(t) \leqslant \overline{\delta}_i \leqslant 0.95, i = 1, 2 \tag{78}$$

In order to intuitively observe the performance of FE using Q-learning along the time domain and trial domain at the same time, therefore set $\delta_{1,k}$ varying only along trials and set $\delta_{2,k}(t)$ varying along the sample time and trials simultaneously.

Referring to the parameters settings of FE using Q-learning, consider the learning rate $\alpha = 0.1$, the cost factor $\gamma = 1$, the ϵ-greedy probability threshold $\epsilon = 0.1$ and the loss function threshold $\varepsilon_{\mathcal{L}} = 10^{-11}$.

5.2. Performance of proposed algorithm

The proposed algorithm is applied to the control task specified in Section 5.1 for a total number of 30 trials with the weighting matrices as $Q = I$ and $R = 0.001I$. Figure 4 and Figure 5 respectively illustrate the desired trajectory and the output trajectories of the linear velocity and the azimuth at the first few trials and the final trial. Figure 6 implies the tracking procedure of the mobile robot at the first few trials and the final trial. From these figures, it is obvious that the output trajectory is rapidly tracking on the reference at the first few trials, and nearly approaches the desired trajectory at the final trial. Moreover, the final output can not perfectly tracking on the desired reference because with the influence of the faults, the tracking error will reach bounded convergence but not converge to zero, which is accordance with what is mentioned in subsection 3.2. Figure 7 and Figure 8

show the corresponding input signal of u_v and u_ϕ. It can be seen that the input signals are gradually refined to achieve the tracking task.

Figure 9 and Figure 10 illustrate the mean square error of the linear velocity and azimuth under both ordinary and logarithmic coordinates. The tracking error of the linear velocity and the tracking error of the azimuth monotonically reduce along the trial and then oscillate relatively smoothly in a bound, which corresponds to what is explained in subsection 3.2 and Remark 5. Figure 11 represents the comparison of different selection of the weighting matrices Q and R. The mean square error of linear velocity is taken for a example. It is clear that increasing the value of Q or decreasing the value of R could speed up the convergence manner and improve the performance, which is a verification of Remark 2.

Figure 12 and Figure 13 describe the mean square estimation error of $\hat{\delta}_{1,k}$ and $\hat{\delta}_{2,k}(t)$ along trials. It is obvious to see that the estimation error can oscillate in a small range which verifies the effectiveness of the Q-learning based FE. The specific FE procedures along the sample time and trials are illustrated in Figure 14 and Figure 15. From Figure 14, one can see that there exists a delay which has a length of a trial during the estimation procedure, as what is mentioned in Remark 4. Furthermore, Figure 15 illustrates that at every sample time the current-trial estimated faults are consistent with the last-trial practical faults. The delay is aroused because of the update procedure, in which case the faults are estimated through the information of the previous trial. The delay also results in the estimation error in Figure 12 and Figure 13.

To further elucidate the effectiveness and advantages of the proposed

algorithm, two other algorithms are applied to this simulation task for comparison. One is classic P-type ILC in with a proportional gain as 0.012. The other one is typical norm-optimal ILC, with $Q = I, R = 0.001I$ and the ILC update law maintain the same with the time-varying and trial-varying fault existing. Figure 16 and Figure 17 give an illustration of the mean square error of the linear velocity and azimuth using different methods. It can be seen that the tracking error of three algorithms all rapidly converge at first few trials and then oscillate in a certain accuracy. In terms of convergence speed, the proposed algorithm is faster than norm-optimal and P-type ILC in both linear velocity and azimuth. In terms of control precision, P-type ILC has obviously poorer performance than norm-optimal ILC and the proposed algorithm in both linear velocity and azimuth. Meanwhile, norm-optimal ILC and the proposed algorithm both oscillate in a certain range after the first few trials. However, in only a few sample times, norm-optimal ILC has a little smaller tracking error than the proposed algorithm. In majority of sample times, the proposed algorithm has obviously better performance in tracking error than the norm-optimal ILC. The improvement in convergence speed and control precision is attributed to that the algorithm estimates the fault and reconfigured the ILC update law in real time, which in a sense plays a role in adaptively controlling the system with time-varying and trail-varying faults. With the combination of FE and FTC, the reliability and the performance of the systems with actuator faults can be maintained. Consequently, the effectiveness and the advantages of the proposed algorithm can be verified.

6. Conclusions

This paper proposed a scheme for Q-learning based FE and FTC under the ILC framework. During the control procedure, Q-learning is introduced for FE and controller reconfiguration. The design of FTC employed the NOILC framework, where the controller is configured in real time based on the FE results from Q-learning to counteract the system dynamic uncertainties brought by faults. Moreover, this paper gave the analysis on convergence property to enhance the theoretical reliability. In the meantime, the proposed algorithm verified its effectiveness and advantages through a mobile robot numerical simulation by comparison with two other algorithms, which makes an improvement on the convergence speed and control precision.

Future will be concerned with verifying the effectiveness of the proposed algorithm with practical experiment. In addition, the design of fault tolerant ILC for nonlinear system with faults could attempt to be solved by RL.

Declaration of competing interest

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

Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grant No. 61773181, 61203092, 62103293, the 111 Project under Grant No. B12018, the Fundamental Research Funds for the Central Universities under Grant No. JUSRP51733B, the Serbian Ministry of

Education, Science and Technological Development under Grant No. 451-03-9/2021-14/200108, the National Science Centre in Poland under Grant No. 2020/39/B/ST7/01487, the Natural Science Foundation of Jiangsu Province under Grant No. BK20210709.

References

[1] C. E. Boudjedir, M. Bouri, D. Boukhetala, Model-free iterative learning control with nonrepetitive trajectories for second-order mimo nonlinear systems—application to a delta robot, IEEE Trans. Ind. Electron. 68 (8) (2020) 7433–7443.
URL https://doi.org/10.1109/TIE.2020.3007091

[2] D. Shen, X. Yu, Learning tracking control over unknown fading channels without system information, IEEE Trans. Neural Netw. Learn. Syst. 32 (6) (2020) 2721–2732.
URL https://doi.org/10.1109/TNNLS.2020.3007765

[3] X. Zhao, Y. Wang, Improved point-to-point iterative learning control for batch processes with unknown batch-varying initial state, ISA Trans. 125 (2022) 290–299.
URL https://doi.org/10.1016/j.isatra.2021.07.007

[4] X. Dai, Q. Quan, J. Ren, K.-Y. Cai, Iterative learning control and initial value estimation for probe–drogue autonomous aerial refueling of uavs, Aerosp. Sci. Technol. 82 (2018) 583–593.
URL https://doi.org/10.1016/j.ast.2018.09.034

[5] C. T. Freeman, Upper limb electrical stimulation using input-output linearization and iterative learning control, IEEE Trans. Control Syst. Technol. 23 (4) (2014) 1546–1554.
URL https://doi.org/10.1109/TCST.2014.2363412

[6] D. A. Bristow, M. Tharayil, A. G. Alleyne, A survey of iterative learning control, IEEE Control Syst. Mag. 26 (3) (2006) 96–114.
URL https://doi.org/10.1109/MCS.2006.1636313

[7] D. Shen, Iterative learning control with incomplete information: A survey, IEEE-CAA J. Automatica Sin. 5 (5) (2018) 885–901.
URL https://doi.org/10.1109/JAS.2018.7511123

[8] H. Tao, W. Paszke, E. Rogers, H. Yang, K. Gałkowski, Iterative learning fault-tolerant control for differential time-delay batch processes in finite frequency domains, J. Process Control 56 (2017) 112–128.
URL https://doi.org/10.1016/j.jprocont.2016.12.007

[9] A. A. Amin, K. M. Hasan, A review of fault tolerant control systems: advancements and applications, Measurement 143 (2019) 58–68.
URL https://doi.org/10.1016/j.measurement.2019.04.083

[10] Z. Lan, Iterative learning control algorithm for sensor fault nonlinear systems, J. Intell. Fuzzy Syst. 40 (4) (2021) 5927–5934.
URL https://doi.org/10.3233/JIFS-189432

[11] L. Wang, F. Liu, J. Yu, P. Li, R. Zhang, F. Gao, Iterative learning fault-tolerant control for injection molding processes against actuator faults,

J. Process Control 59 (2017) 59–72.
URL https://doi.org/10.1016/j.jprocont.2017.08.013

[12] L. Wang, L. Dong, Y. Chen, K. Wang, F. Gao, Iterative learning control for actuator fault uncertain systems, Symmetry 14 (10) (2022) 1969.
URL https://doi.org/10.3390/sym14101969

[13] D. H. Owens, J. Hätönen, Iterative learning control – An optimization paradigm, Annu. Rev. Control 29 (1) (2005) 57–70.
URL https://doi.org/10.1016/j.arcontrol.2005.01.003

[14] Z. Wang, C. Hu, Y. Zhu, S. He, M. Zhang, H. Mu, Newton-ILC contouring error estimation and coordinated motion control for precision multi-axis systems with comparative experiments, IEEE Trans. Ind. Electron. 65 (2) (2017) 1470–1480.
URL https://doi.org/10.1109/TIE.2017.2733455

[15] D. H. Owens, Multivariable norm optimal and parameter optimal iterative learning control: a unified formulation, Int. J. Control 85 (8) (2012) 1010–1025.
URL https://doi.org/10.1080/00207179.2012.673136

[16] N. Amann, D. H. Owens, E. Rogers, Iterative learning control using optimal feedback and feedforward actions, Int. J. Control 65 (2) (1996) 277–293.
URL https://doi.org/10.1080/00207179608921697

[17] Y. Chen, B. Chu, C. T. Freeman, Point-to-point iterative learning control with optimal tracking time allocation, IEEE Trans. Control Syst.

Technol. 26 (5) (2017) 1685–1698.
URL https://doi.org/10.1109/TCST.2017.2735358

[18] Z. Zhuang, H. Tao, Y. Chen, V. Stojanovic, W. Paszke, Iterative learning control for repetitive tasks with randomly varying trial lengths using successive projection, Int. J. Adapt. Control Signal Process. 36 (5) (2022) 1196–1215.
URL https://doi.org/10.1002/acs.3396

[19] Y. Liu, X. Ruan, Linearly monotonic convergence of nonlinear parameter-optimal iterative learning control to linear discrete-time-invariant systems, Int. J. Robust Nonlinear Control 31 (9) (2021) 3955–3981.
URL https://doi.org/10.1002/rnc.5448

[20] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.
URL https://mitpress.mit.edu/9780262039246/reinforcement-learning/

[21] I. Carlucho, M. De Paula, S. A. Villar, G. G. Acosta, Incremental Q-learning strategy for adaptive pid control of mobile robots, Expert Syst. Appl. 80 (2017) 183–199.
URL https://doi.org/10.1016/j.eswa.2017.03.002

[22] S. Blouin, M. Babahaji, H. Mahboubi, W. Lucia, M. M. Asadi, A. G. Aghdam, Estimation of the connectivity of random graphs through Q-

learning techniques, IEEE J. Radio Freq. Identif. 6 (2022) 318–331.
URL https://doi.org/10.1109/JRFID.2022.3178086

[23] Z. Jin, M. Ma, S. Zhang, Y. Hu, Y. Zhang, C. Sun, Secure state estimation of cyber-physical system under cyber attacks: Q-learning vs. SARSA, Electronics 11 (19) (2022) 3161.
URL https://doi.org/10.3390/electronics11193161

[24] H. Tao, J. Li, Y. Chen, V. Stojanovic, H. Yang, Robust point-to-point iterative learning control with trial-varying initial conditions, IET Contr. Theory Appl. 14 (19) (2020) 3344–3350.
URL https://doi.org/10.1049/iet-cta.2020.0557

[25] D. H. Owens, C. T. Freeman, T. Van Dinh, Norm-optimal iterative learning control with intermediate point weighting: Theory, algorithms, and experimental evaluation, IEEE Trans. Control Syst. Technol. 21 (3) (2012) 999–1007.
URL https://doi.org/10.1109/TCST.2012.2196281

[26] S. Yang, J.-X. Xu, D. Huang, Y. Tan, Optimal iterative learning control design for multi-agent systems consensus tracking, Syst. Control Lett. 69 (2014) 80–89.
URL https://doi.org/10.1016/j.sysconle.2014.04.009

[27] K. Watanabe, J. Tang, M. Nakamura, S. Koga, T. Fukuda, A fuzzy-gaussian neural network and its application to mobile robot control, IEEE Trans. Control Syst. Technol. 4 (2) (1996) 193–199.
URL https://doi.org/10.1109/87.486346

Table 1: The specific definition of the system parameters.

variable	definition	value
I_v	moment of inertia around the robot	10 kgm^2
M	mass of the robot	200 kg
l	distances between left or right wheel	0.3 m
I_w	moment of inertia of the robot	0.005 kgm^2
c	viscous friction factor	0.05 kgm^2/s
r	radius of wheel	0.1 m
k	driving gain factor	5

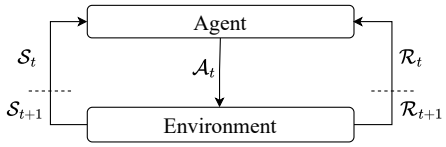

Figure 1: The interaction of agent and environment.

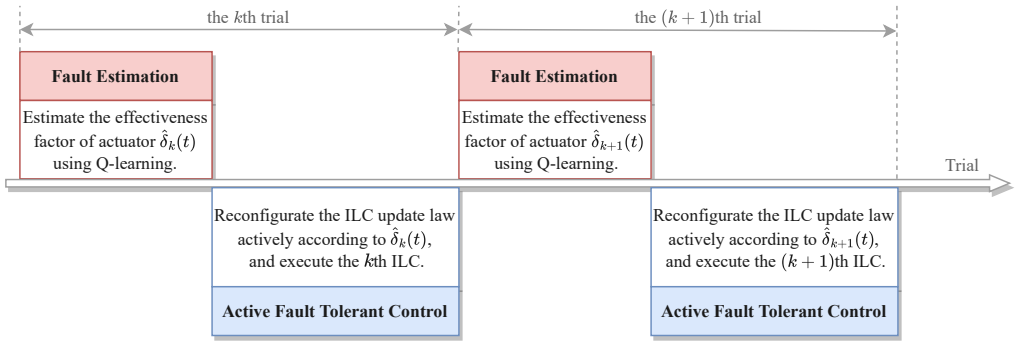

Figure 2: A scheme for FE and FTC under the ILC framework.

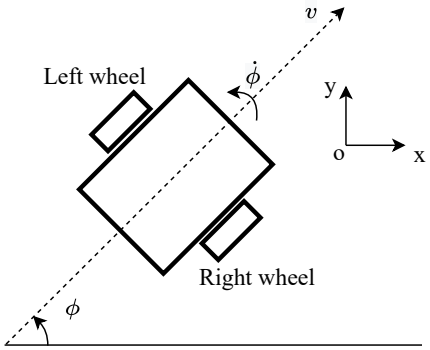

Figure 3: A mobile robot with two independent wheels.

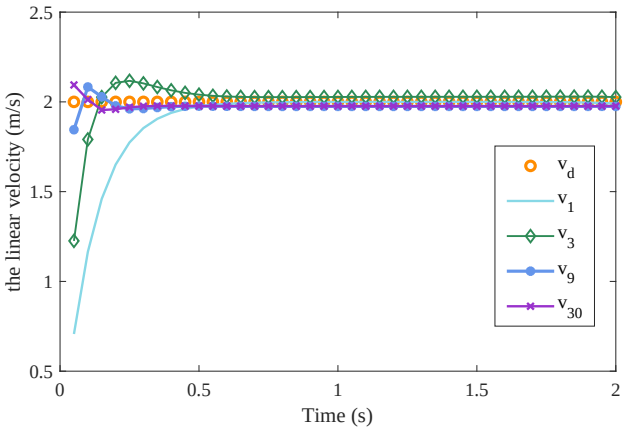

Figure 4: Output signals of linear velocity at the first few trials and the final trial.

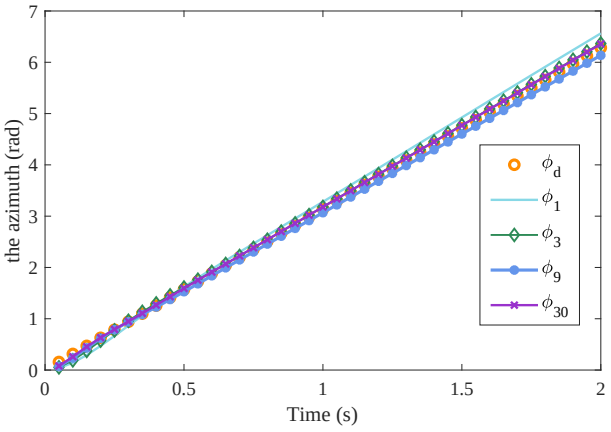

Figure 5: Output signals of azimuth at the first few trials and the final trial.

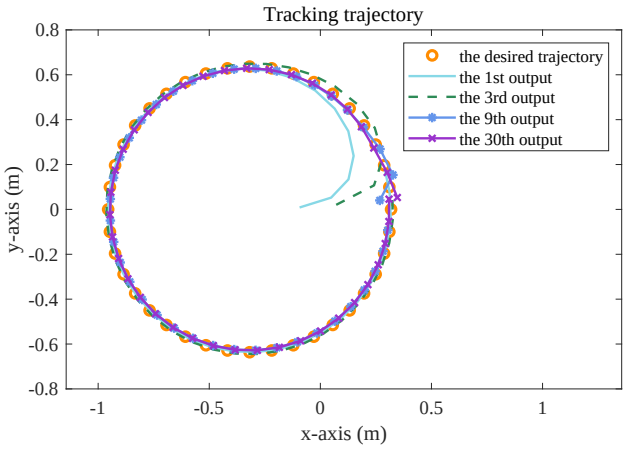

Figure 6: Tracking trajectory of the mobile robot at the first few trials and the final trial.

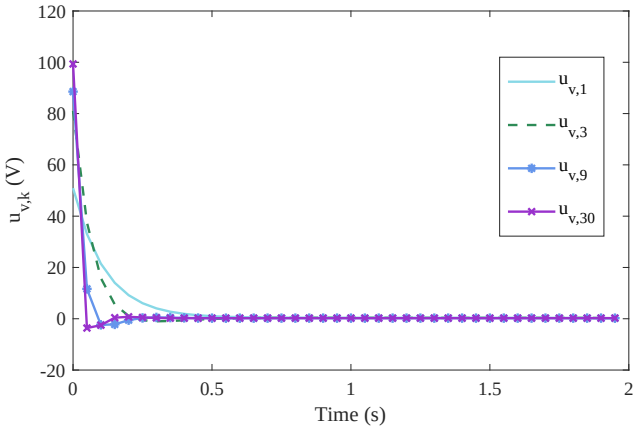

Figure 7: Input signals of linear velocity at the first few trials and the final trial.

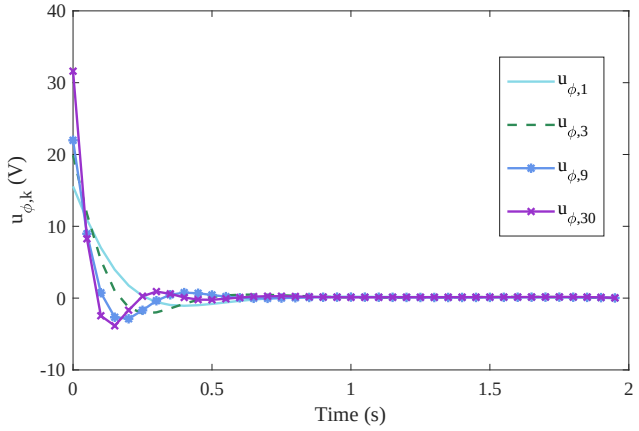

Figure 8: Input signals of azimuth at the first few trials and the final trial.

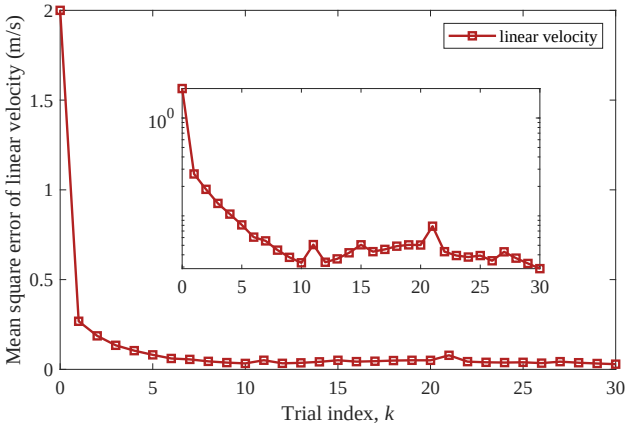

Figure 9: Mean square error of linear velocity under both ordinary and logarithm coordinates.

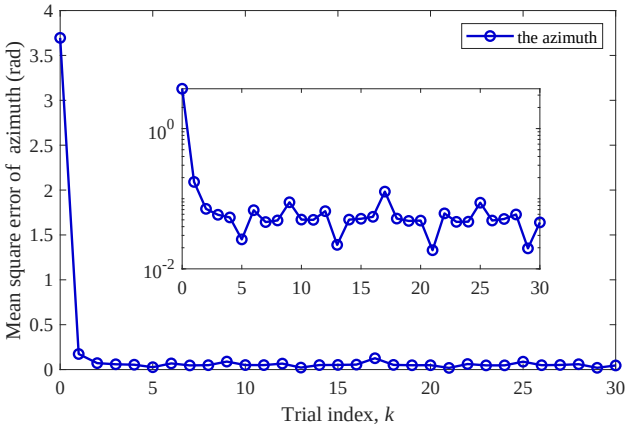

Figure 10: Mean square error of azimuth under both ordinary and logarithm coordinates.

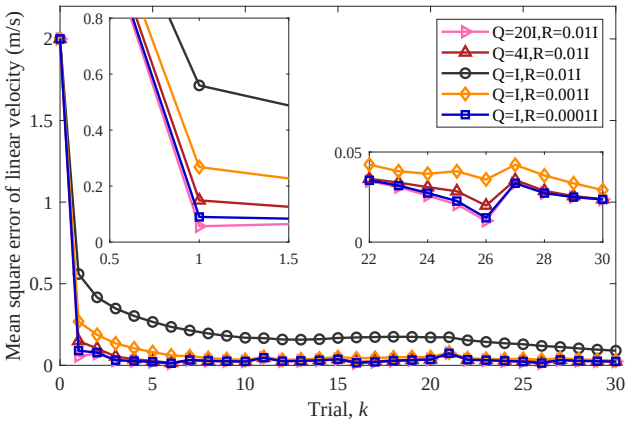

Figure 11: Mean square error of linear velocity with different selection of Q and R.

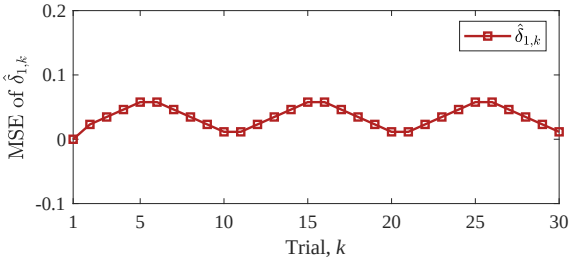

Figure 12: Mean square estimation error of $\hat{\delta}_{1,k}$ along the trials.

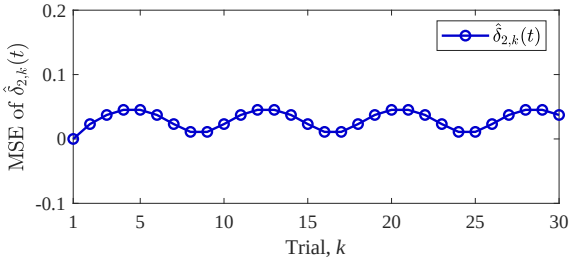

Figure 13: Mean square estimation error of $\hat{\delta}_{2,k}(t)$ along the trials.

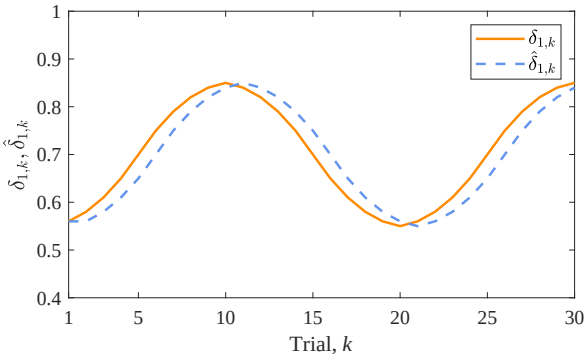

Figure 14: the comparison between the estimated $\hat{\delta}_{1,k}$ and the practical fault $\delta_{1,k}$.

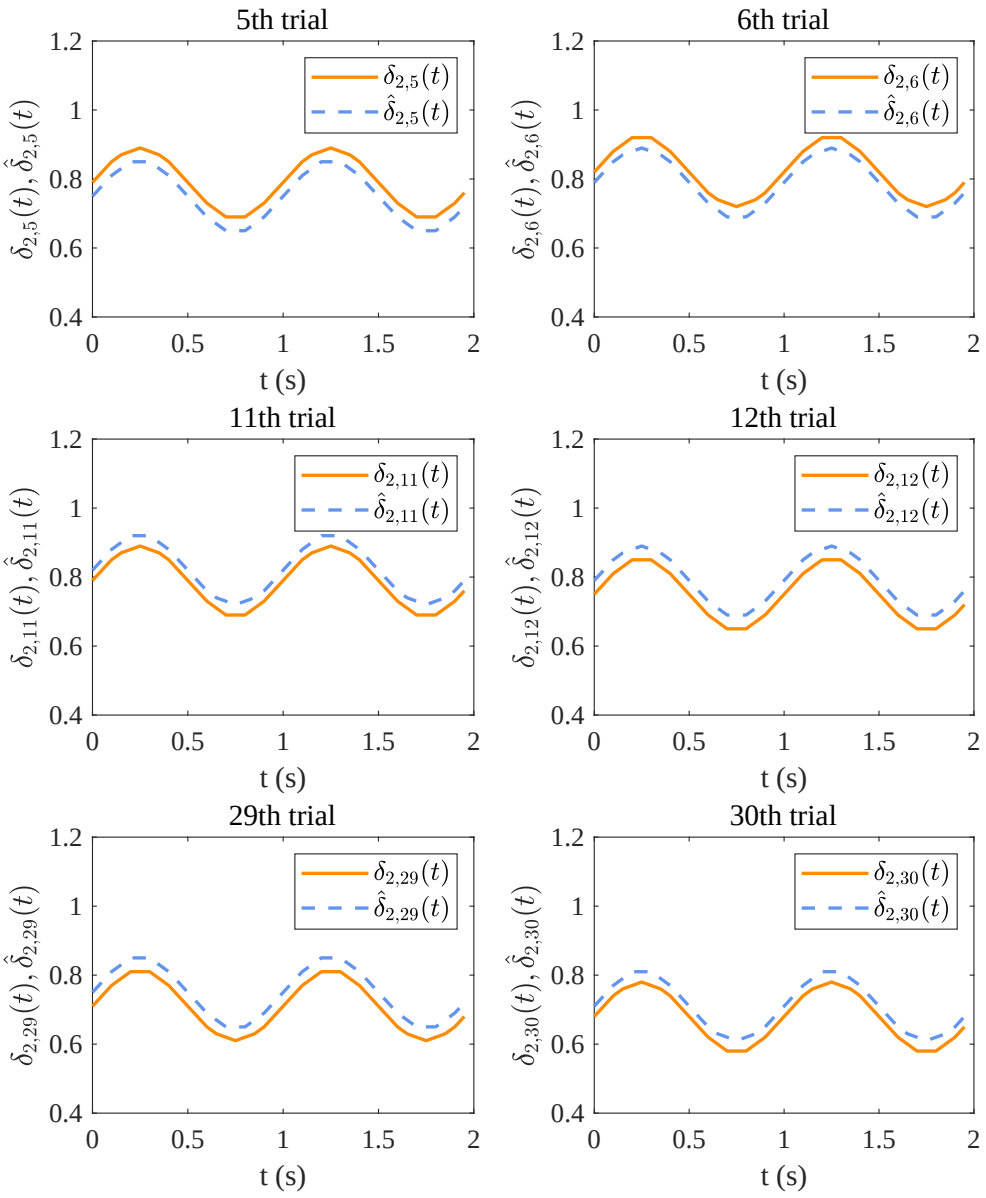

Figure 15: the comparison between the estimated $\hat{\delta}_{2,k}(t)$ and the practical fault $\delta_{2,k}(t)$ in some trials.

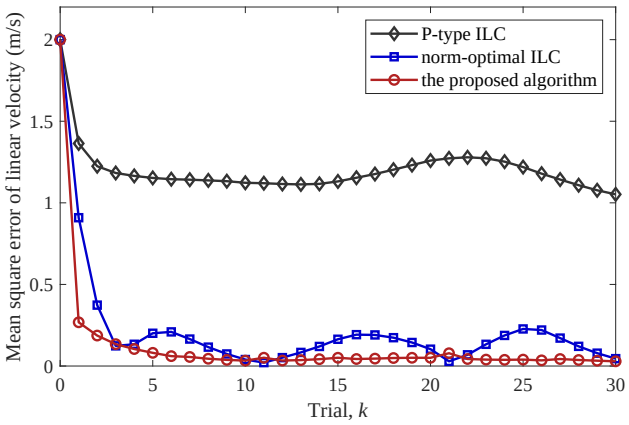

Figure 16: Mean square error of linear velocity using different methods.

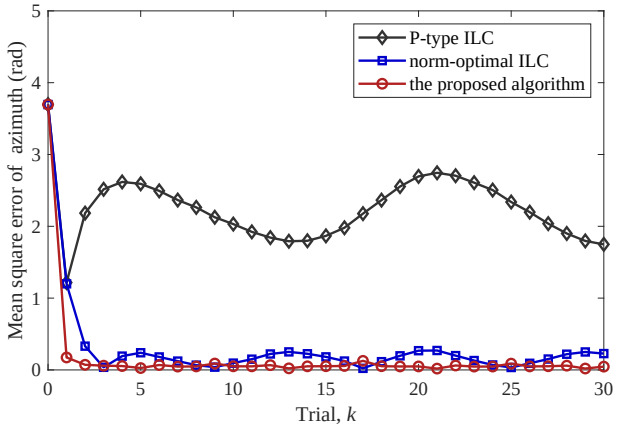

Figure 17: Mean square error of azimuth using different methods.