Data-driven online learning control using interval type-2 evolving fuzzy systems and error compensation

Qinyin Su
College of Electrical Engineering
Sichuan University
Chengdu, China
qinyinsu@163.com

Tao Zhao*
College of Electrical Engineering
Sichuan University
Chengdu, China
zhaotaozhaogang@126.com

Hainan Yang
College of Electrical Engineering
Sichuan University
Chengdu, China
yang12261204@163.com

Peng Qin
College of Electrical Engineering
Sichuan University
Chengdu, China
qinpeng2@stu.scu.edu.cn

***Abstract*—In this article, a data-driven online learning controller (DD-oLC), which is a completely data-based control strategy, is proposed. The method generates feed-forward control signals by learning the inverse dynamics of the system from online data via an interval type-2 evolving fuzzy system (IT2EFS), which has the advantage of utilizing the structural evolving and parameter adaptive capabilities of IT2EFS to achieve incremental and continuous learning, and is better suited for robotic manipulators that have been in open environments for a long-term period. Meanwhile, a QR decomposition-based recursive least squares (QRD-RLS) residual model online learning module is designed to enhance control performance, which complements the residual torques in the control process. Finally, the advancement of the proposed controller is demonstrated by simulation verification.**

***Index Terms*—Online learning control, evolving fuzzy systems, robotic manipulators, error compensation.**

I. Introduction

Robotic manipulators, as industrial automation equipment, have achieved a wide range of applications in various fields. Nowadays, driven by the expansion of application scenarios and technological innovation, the control performance requirements of robotic manipulators in multiple industries have also put forward higher demands. Therefore, the problem of how to improve the high-precision tracking control of such strongly coupled nonlinear systems as robotic manipulators has become a major difficulty in this field [1], [2].

Nowadays, many control strategies for robotic manipulators have been applied to improve their tracking control accuracy, such as neural networks (NNs) control [3], sliding mode control (SMC) [4], and fuzzy control [5], etc. In [6], an adaptive NNs control method is proposed to solve the n-link constrained robotic manipulator control problem driven by practical requirements. In [7], a prescribed-time fuzzy adaptive control is designed to minimize the effects of the initial values of the system and the controller parameters on the settling-time function of the finite/fixed time control, and to achieve precise control of the n-link robotic manipulator. In [8], an adaptive fixed-time SMC strategy is proposed for solving control problems with parameter uncertainty and input-saturated robots. These control strategies can solve the robotic manipulator tracking control problem well in some scenarios, but their controller design relies on model information. However, the high degree of nonlinearity and uncertainty in the robotic manipulator make it still a challenge to accurately track its trajectory in the lack of model information.

Noticeably, in recent years, advanced methods such as machine learning [9], [10] and reinforcement learning [11], [12] have gradually entered the field of intelligent control, and such methods can have excellent performance even when the system model is uncertain. Among them, machine learning is capable of extracting knowledge from training samples or data streams, and is often used to construct dynamic models of nonlinear systems, its powerful learning ability can learn inverse dynamics models of robotic systems from data, thus receiving extensive attention from scholars in related fields [13], [14]. However, the existing control methods based on inverse dynamics model learning in the field of robotics are mostly based on fixed structure controllers, and the learning process mainly focuses on the model parameters, which may be inappropriate for robotic manipulators that have been in an open environment for a long-term period. Therefore, it is crucial to design controllers with the ability of structure

*Corresponding Author

This work was supported in part by the National Natural Science Foundation of China under Grant 62473273, in part by Sichuan Science and Technology Program under Grant 2024ZYD0029, and in part by Special Project for University-Local Science and Technology Cooperation between Sichuan University and Zigong City under Grant 2024CDZG-4.

evolution and knowledge extraction [15].

Evolving fuzzy system (EFS), as an online learning algorithm with stepwise evolving structure and adaptive parameters, has achieved great success in feature extraction and system modeling [16], [17], and also provides new ideas and tools for online learning control of robotic manipulators. In [18], an online learning control strategy is proposed to achieve the complex dynamics trajectory tracking control problem in robotic systems. In [19], a robust evolving cloud-based controller (RECCo) is proposed to achieve process control without any mathematical model of the controlled process. However, noteworthy is that although the evolving fuzzy system is able to establish the inverse dynamic model, $\tau = f^{-1}(q, \dot{q}, \ddot{q})$, of the robotic manipulators model using structural evolving and parameter learning, residuals will inevitably exist between the total joint torques at the output of the controller and the actual values due to the nonlinear factors that are difficult to model physically and the effects of the disturbances at the torque end [20].

Therefore, this study focuses on online learning of the inverse dynamics of the manipulators using interval type-2 evolving fuzzy systems (IT2EFS), and a novel online learning controller is designed based on the structural evolving and parameter adapting capabilities of the IT2EFS. Meanwhile, a QR decomposition-based recursive least squares (QRD-RLS) residual model online learning module is designed to compensate for the residual torques during the control process to improve the control accuracy.

The rest of the article can be outlined as below. An introduction to the inverse dynamics model and the IT2 Takagi-Sugeno fuzzy system (IT2-TSFS) is given in section II. In Section III, the details of the proposed data-driven online learning control framework is introduced. In Section IV, the simulation experiment is conducted to validate the advancement of the proposed method. Finally, Section V gives a conclusion of the article.

II. Problem Formulation

A. Interval Type-2 TS Fuzzy System

The form of the first-order IT2-TSFS can be expressed as follows:

$$R^i: \text{If } x_1 \sim \tilde{A}_1^i \text{ and } ... \text{ and } x_n \sim \tilde{A}_n^i, \quad \text{Then } y^i \tag{1}$$

where $i = 1, 2, ..., C$, and $\boldsymbol{x} = [x_1, x_2, ..., x_n]^\mathsf{T}$ is the inputs; $\tilde{A}_j^i$ is the IT2 fuzzy set; $y^i = a_0^i + \sum_{j=1}^n a_j^i x_j$, where $\boldsymbol{a}_i = [a_0^i, a_1^i, ..., a_j^i]^\mathsf{T}$ is the consequent parameters. The membership function (MF) is chosen as:

$$\mu_{\tilde{A}_j^i} = \exp\Big(-\frac{\left\|x_j - \upsilon_j^i\right\|^2}{2(\sigma_j^i)^2}\Big) \equiv \mathbb{N}(\upsilon_j^i, \sigma_j^i; x_j) \tag{2}$$

where σ_j^i and υ_j^i are the center and width values of the MF, respectively. Each MF in the antecedent part can be represented as the upper and lower range values $\overline{\mu}_j^i$ and $\underline{\mu}_j^i$, respectively, where

$$\overline{\mu}_j^i(x_j) = \begin{cases} \mathbb{N}(\upsilon_{j1}^i, \sigma_j^i; x_j) & x_j < \upsilon_{j1}^i \\ 1 & \upsilon_{j1}^i \le x_j \le \upsilon_{j2}^i \\ \mathbb{N}(\upsilon_{j2}^i, \sigma_j^i; x_j) & x_j > \upsilon_{j2}^i \end{cases} \tag{3}$$

$$\underline{\mu}_j^i(x_j) = \begin{cases} \mathbb{N}(\upsilon_{j2}^i, \sigma_j^i; x_j) & x_j \le \overline{\upsilon}_j^i \\ \mathbb{N}(\upsilon_{j1}^i, \sigma_j^i; x_j) & x_j > \overline{\upsilon}_j^i \end{cases} \tag{4}$$

where $\overline{\upsilon}_j^i = (\upsilon_{j1}^i + \upsilon_{j2}^i)/2$. And the lower/upper firing strength values are:

$$\underline{\phi}^i = \prod_{j=1}^n \underline{\mu}_j^i, \quad \overline{\phi}^i = \prod_{j=1}^n \overline{\mu}_j^i \tag{5}$$

And, by using the reduction procedure [15], the output $\hat{y}$ is as follows:

$$\begin{aligned} \hat{y} &= \zeta \frac{\sum_{i=1}^C \underline{\phi}^i y^i}{\sum_{i=1}^C \underline{\phi}^i} + (1-\zeta)\frac{\sum_{i=1}^C \overline{\phi}^i y^i}{\sum_{i=1}^C \overline{\phi}^i} \\ &= \sum_{i=1}^C \xi^i \bar{\boldsymbol{x}}^\mathsf{T} \boldsymbol{a}^i \end{aligned} \tag{6}$$

where $y^i = \bar{\boldsymbol{x}}^\mathsf{T}\boldsymbol{a}^i$, $\xi^i = \zeta\underline{\phi}^i \Big/ \sum_{i=1}^C \underline{\phi}^i + (1-\zeta)\overline{\phi}^i \Big/ \sum_{i=1}^C \overline{\phi}^i$, and ζ is the positive constants which satisfy $\zeta \in (0, 1]$ and $\bar{x} = [1, x^\mathsf{T}]^\mathsf{T}$.

B. Inverse dynamics

Based on the reference trajectory, the joint torques required to achieve the target trajectory are calculated, a process often referred to as inverse dynamics. The inverse dynamics formulation, based on the equations of motion of the robotic manipulators with n degrees of freedom, can be described as

$$\mathbb{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbb{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \mathbb{G}(\boldsymbol{q}) + \boldsymbol{\varepsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \boldsymbol{F}_{\boldsymbol{\tau}} \tag{7}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathcal{R}^n$ represent the position, velocity, and acceleration of each joint. $\boldsymbol{F}_{\boldsymbol{\tau}} \in \mathcal{R}^n$ is the torque. $\mathbb{M}(\boldsymbol{q}) \in \mathcal{R}^{n\times n}$ stands for the inertia matrix, $\mathbb{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathcal{R}^{n\times n}$ represents the centripetal force and the Coriolis force matrix. $\mathbb{G}(\boldsymbol{q}) \in \mathcal{R}^n$ signifies the gravity, $\boldsymbol{\varepsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \in \mathcal{R}^n$ indicates some uncertainties in the model, including unmodeled dynamics, complex frictions, possible disturbances, etc.

In this study, a data-driven learning-based control method is used to construct the inverse dynamics model of the robotic manipulators, whereby Eq. (7) can be rewritten in the following form:

$$\boldsymbol{F}_{\boldsymbol{\tau}} = f^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \tag{8}$$

III. Proposed Data-Driven Online Learning Control Framework

A. Controller Structure

The data-driven online learning control (DD-oLC) framework proposed is shown in Fig. 1. The control strategy contains IT2EFS, a feedback controller (the PID controller is used in this article), and a residual compensation module.

The feedback controller provides sufficient learning data to IT2EFS in the initial stage of the system to help it construct an

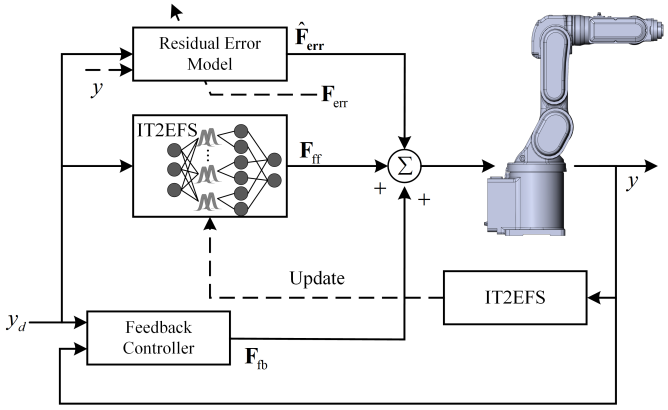

Fig. 1. The diagram for the proposed controller framework.

inverse dynamic of the system. IT2EFS generates feedforward control torques during the control process. The residual module is capable of learning the control residual model online to approximate the mismatch torques and compensate for residual torques to improve the control accuracy. The final control torques can be expressed as:

$$\boldsymbol{F} = \boldsymbol{F}_{\text{fb}} + \boldsymbol{F}_{\text{ff}} + \boldsymbol{F}_{\text{err}} \tag{9}$$

where $\boldsymbol{F}_{\text{fb}}$ is the output of the feedback controller, $\boldsymbol{F}_{\text{ff}}$ is the output of IT2EFS, and $\boldsymbol{F}_{\text{err}}$ is the output of the residual error model.

B. Interval type-2 evolving fuzzy systems

1) Rule generation: The IT2EFS is initialized based on the first data $\boldsymbol{x}_1$, where the global parameters are initialized as [15]:

$$\begin{aligned} \upsilon_{j1}^1 &= x_j^1 - \Delta x \\ \upsilon_{j2}^1 &= x_j^1 + \Delta x \\ \sigma_j^1 &= \sigma_{\text{init}} \\ \boldsymbol{a}_1 &= \boldsymbol{O}_{(n+1)\times 1} \\ \boldsymbol{\Sigma}_1 &= \kappa \boldsymbol{I}_{(n+1)\times(n+1)} \end{aligned} \tag{10}$$

where Δx is the uncertain mean and $\sigma_{\text{init}} = 0.5$ is the initial value of the width. $\boldsymbol{\Sigma}_1$ is the the corresponding covariance matrix. $\kappa = 1000$ is a constant for initialization. The mean value of firing strength is determined as:

$$\phi_\dagger^i = \frac{1}{2}\left(\underline{\phi}^i + \overline{\phi}^i\right) \tag{11}$$

Then, the following rule generation condition can be obtained. If $\phi_\dagger^{i^\star} < \Xi_{th}^g$, generate a new rule, where $\Xi_{th}^g = e^{0.5}(2/(1+(\tilde{\Lambda}_k)) - 1)$ is the rule generation threshold, where $\tilde{\Lambda}_k = 0.85\tilde{\Lambda}_{k-1} + \|y_k - \hat{y}_k\|$ and $i^\star = \arg\max_{1\le i\le C} \phi_\dagger^i(\boldsymbol{x}_k)$. Once rule generation condition holds, a new rule R_{C+1} is generated, then

$$\upsilon_{j1}^{C+1} = x_j^k - \Delta x \tag{12}$$

$$\upsilon_{j2}^{C+1} = x_j^k + \Delta x \tag{13}$$

$$\sigma_j^{C+1} = \beta \left| x_j^k - \frac{\upsilon_{j1}^{i^\star} + \upsilon_{j2}^{i^\star}}{2} \right| \tag{14}$$

$$\boldsymbol{a}^{C+1} = \boldsymbol{a}^{i^\star} \tag{15}$$

$$\boldsymbol{\Sigma}^{C+1} = \kappa \boldsymbol{I}_{(n+1)\times(n+1)} \tag{16}$$

where $\overline{v}_j^{i^\star} = (\upsilon_{j1}^{i^\star} + \upsilon_{j2}^{i^\star})/2$ and $\beta = 0.5$.

2) Rule base optimization: If the rule generation conditions are not met, the relevant rules $R_{i^\star}$ need to be updated as follows [21]:

$$\overline{\boldsymbol{v}}_k^{i^\star} = \overline{\boldsymbol{v}}_{k-1}^{i^\star} + \frac{1}{N_{k-1}^{i^\star}+1}\left(\boldsymbol{x}^k - \overline{\boldsymbol{v}}_{k-1}^{i^\star}\right) \tag{17}$$

$$\begin{aligned} \boldsymbol{\sigma}_k^{i^\star} = \Bigg[(\boldsymbol{\sigma}_{k-1}^{i^\star})^2 + \frac{(\boldsymbol{x}^k - \overline{\boldsymbol{v}}_k^{i^\star})^2 - (\boldsymbol{\sigma}_{k-1}^{i^\star})^2}{N_{k-1}^{i^\star}+1} \\ + \frac{N_{k-1}^{i^\star}(\overline{\boldsymbol{v}}_k^{i^\star} - \overline{\boldsymbol{v}}_{k-1}^{i^\star})^2}{N_{k-1}^{i^\star}+1} \Bigg]^{\frac{1}{2}} \end{aligned} \tag{18}$$

$$N_k^{i^\star} = N_{k-1}^{i^\star} + 1 \tag{19}$$

In the online learning process, as the data samples increase, it may result in partially constructed rules becoming similar, and therefore they need to be merged. Before merging, it is necessary to determine which rules need to be merged, therefore, the following rule overlap measure criterion is defined:

$$O(R^{s1}, R^{s2}) = \frac{\mu_\dagger^{s1}(\overline{v}^{s2}) + \mu_\dagger^{s2}(\overline{v}^{s1})}{2}, \quad O(\cdot) \in (0, 1] \tag{20}$$

where $\mu_\dagger^{s1} = \frac{1}{2}(\overline{\mu}^{s1}(\overline{v}^{s2}) + \underline{\mu}^{s1}(\overline{v}^{s2}))$,$\mu_\dagger^{s2} = \frac{1}{2}(\overline{\mu}^{s2}(\overline{v}^{s1}) + \underline{\mu}^{s2}(\overline{v}^{s1}))$.

When rule merging condition holds, i.e. $S(R^{s1}, R^{s2}) > \Xi_{th}^m$, where $\Xi_{th}^m = 0.95$ is the rule merging threshold and $S(R_{s1}, R_{s2}) = \max_{\forall s1,s2}(O(R^{s1}, R^{s2}))$, then rule R^{s1} is merged with R^{s2} as follows:

$$\overline{\boldsymbol{v}}_k^{i^m} = \frac{N_k^{s1}\overline{\boldsymbol{v}}_k^{s1} + N_k^{s2}\overline{\boldsymbol{v}}_k^{s2}}{N_k^{i^m}} \tag{21}$$

$$\begin{aligned} \boldsymbol{\sigma}_k^{i^m} = \Bigg[\frac{1}{N_k^{i^m}} \Bigg(N_k^{s1}(\boldsymbol{\sigma}_k^{s1})^2 + N_k^{s2}(\boldsymbol{\sigma}_k^{s2})^2 \\ + N_k^{s1}\mathcal{A} + N_k^{s2}\mathcal{B} \Bigg) \Bigg]^{\frac{1}{2}} \end{aligned} \tag{22}$$

$$\boldsymbol{a}^{i^m} = \frac{N_k^{s1}\boldsymbol{a}^{s1} + N_k^{s2}\boldsymbol{a}^{s2}}{N_k^{i^m}} \tag{23}$$

where $N_k^{i^m} = N_k^{s1} + N_k^{s2}$, $\mathcal{A} = \left\|\overline{\boldsymbol{v}}_k^{s1} - \overline{\boldsymbol{v}}_k^{i^m}\right\|^2$ and $\mathcal{B} = \left\|\overline{\boldsymbol{v}}_k^{s2} - \overline{\boldsymbol{v}}_k^{i^m}\right\|^2$. Rule merging will use R^{i^m} instead of R^{s1} and R^{s2}, and the number of rules will be reduced by one.

To reduce redundancy in the rule base, the utility [21] is used to determine which rules are redundant. The utility can be calculated as follows:

$$U_k^i = U_{k-1}^i + \frac{\theta_k^i - U_{k-1}^i}{T - T_g + 1} \tag{24}$$

where T is the time k, and T_g is the time when the R^i was generated, and θ_k^i is the normalized firing strength. When the utility of a rule is below the threshold ($U_k^i < \Xi_{th}^p$), the rule is pruned, where $\Xi_{th}^p = \Xi_{th}^g$ is the rule pruning threshold.

3) Parameters Learning: In addition to structural learning, the parameters of IT2EFS need to be learned. Hence, the consequent parameters are updated by the local weighted recursive least square [15]. The error functions is defined as:

$$E^i = \sum_{k=1}^{n} \xi_k^i (y_k - \bar{\boldsymbol{x}}_k^\mathsf{T} \boldsymbol{a}_k^i)^2 \tag{25}$$

Hence, the parameter is updated as:

$$\boldsymbol{a}_{k+1}^i = \boldsymbol{a}_k^i + \frac{\xi_k^i \boldsymbol{\Sigma}_k^i \bar{\boldsymbol{x}}_k (y_k - \bar{\boldsymbol{x}}_k^\mathsf{T} \boldsymbol{a}_k^i)}{1 + \xi_k^i \bar{\boldsymbol{x}}_k^\mathsf{T} \boldsymbol{\Sigma}_k^i \bar{\boldsymbol{x}}_k} \tag{26}$$

$$\boldsymbol{\Sigma}_{k+1}^i = \boldsymbol{\Sigma}_k^i - \frac{\xi_k^i \boldsymbol{\Sigma}_k^i \bar{\boldsymbol{x}}_k \bar{\boldsymbol{x}}_k^\mathsf{T} \boldsymbol{\Sigma}_k^i}{1 + \xi_k^i \bar{\boldsymbol{x}}_k^\mathsf{T} \boldsymbol{\Sigma}_k^i \bar{\boldsymbol{x}}_k} \tag{27}$$

And for the antecedent parameters, there is the following error function:

$$E_k = \frac{1}{2}(y_k - \hat{y}_k)^2 \tag{28}$$

Based on the gradient descent algorithm, the antecedent parameters in rule i and variable j can be updated as

$$\overline{v}_{j,k+1}^i = \overline{v}_{j,k}^i - \varsigma \frac{\partial E_k}{\partial \overline{v}_{j,k}^i} \tag{29}$$

$$\Delta x_{j,k+1}^i = \Delta x_{j,k}^i - \varsigma \frac{\partial E_k}{\partial \Delta x_{j,k}^i} \tag{30}$$

$$\sigma_{j,k+1}^i = \sigma_{j,k}^i - \varsigma \frac{\partial E_k}{\partial \sigma_{j,k}^i} \tag{31}$$

where ς is the learning rate. The derivation of the antecedent part in (29) to (31) is as follows:

$$\frac{\partial E_k}{\partial \overline{v}_j^i} = \tilde{y}\left(\frac{\partial \hat{y}_k}{\partial \overline{\phi}^i} \frac{\partial \overline{\phi}^i}{\partial \overline{v}_j^i} + \frac{\partial \hat{y}_k}{\partial \underline{\phi}^i} \frac{\partial \underline{\phi}^i}{\partial \overline{v}_j^i}\right) \tag{32}$$

$$\frac{\partial E_k}{\partial \Delta x_j^i} = \tilde{y}\left(\frac{\partial \hat{y}_k}{\partial \overline{\phi}^i} \frac{\partial \overline{\phi}^i}{\partial \Delta x_j^i} + \frac{\partial \hat{y}_k}{\partial \underline{\phi}^i} \frac{\partial \underline{\phi}^i}{\partial \Delta x_j^i}\right) \tag{33}$$

$$\frac{\partial E_k}{\partial \sigma_j^i} = \tilde{y}\left(\frac{\partial \hat{y}_k}{\partial \overline{\phi}^i} \frac{\partial \overline{\phi}^i}{\partial \sigma_j^i} + \frac{\partial \hat{y}_k}{\partial \underline{\phi}^i} \frac{\partial \underline{\phi}^i}{\partial \sigma_j^i}\right) \tag{34}$$

where $\tilde{y} = \hat{y}_k - y_k$.

C. Online learning of the residual error model

Although IT2EFS has been able to approximate the inverse dynamic $\boldsymbol{F}_{\boldsymbol{\tau}} = f^{-1}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ of the robotic manipulators satisfactorily. However, due to the nonlinear factors and the influence of disturbances at the torque end, there is inevitably a residual between the total joint torque at the output of the controller and the actual value. Therefore, in this study, a recursive strategy is designed to learn the residual model online to compensate for the mismatched torque during the control process. The predicted residual torques are defined by the indirect method as follows [22]:

$$\boldsymbol{F}_{\text{err},k} = \boldsymbol{F}_{k-1} - \tilde{\boldsymbol{F}}_{\boldsymbol{\tau},k-1} = \mathbb{E}_k S_k \tag{35}$$

where $\mathbb{E}_k$ is the linearized residual error model, and $S_k = [\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k, \boldsymbol{q}_{d,k+1}, \dot{\boldsymbol{q}}_{d,k+1}]^\mathsf{T}$ is the system state. From this, online learning data can be defined as $\mathcal{D}_k = \langle \boldsymbol{F}_{\text{err},k}, S_k \rangle$. To learn the residual error model, the QR-decomposition based recursive least-squares (QRD-RLS) are introduced. $\mathbb{E}_k$ is determined by minimizing the following performance metrics:

$$J_K(\mathbb{E}_k) = \sum_{i=0}^{k} \rho^{k-i} \left\| \boldsymbol{F}_{\text{err},k} - \mathbb{E}_k S_k \right\|^2 \tag{36}$$

where $\rho \in (0, 1]$ is the forgetting factor. Construct the following augmented matrix:

$$Z_k = \begin{bmatrix} R_{k-1} \\ \sqrt{\rho} \cdot \tilde{S}_k^\mathsf{T} \end{bmatrix} \tag{37}$$

$$\tilde{b}_k = \begin{bmatrix} b_{k-1} \\ \sqrt{\rho} \cdot \boldsymbol{F}_{\text{err},k}^\mathsf{T} \end{bmatrix} \tag{38}$$

where R_{k-1} is the upper triangular matrix, b_{k-1} is the right-hand side item and $\tilde{S}_k = [1, S_k]^\mathsf{T}$. Performing QR decomposition with Givens rotation for Z_k and $\tilde{b}_k$:

$$Z_k = Q_k R_k \tag{39}$$

$$b_k = Q_k^\mathsf{T} \tilde{b}_k \tag{40}$$

where Q_k is the orthogonal matrix. Thus, the following updated $\mathbb{E}_k$ is obtained:

$$\mathbb{E}_k = (R_k^{-1} b_k)^\mathsf{T} \tag{41}$$

Finally, the following residual error model is obtained:

$$\hat{\boldsymbol{F}}_{\text{err},k} = \mathbb{E}_k \tilde{S}_k \tag{42}$$

where $\tilde{S}_k$ is $[1, S_k]^\mathsf{T}$. In the control process, the residual error model can be updated online to approximate the mismatch torques in the control process, providing a compensation signal to enhance the control accuracy.

IV. Simulation Results

To verify the advancement of the proposed control strategy, simulation experiments are conducted on the first three joints of the common benchmark PUMA560 (shown in Fig. 2). The initial value is specified as $\boldsymbol{q}_0 = [0, 90, -90]^\mathsf{T}$ (deg) and the sampling period is 1000 Hz. The ideal trajectory is given as:

$$\boldsymbol{q}_d = A \sin(t) + \boldsymbol{q}_0, \quad \text{(deg)} \tag{43}$$

where $A = [80, 70, 70]^\mathsf{T}$.

To simulate possible changes in real-world dynamics, the following uncertainties are introduced in the simulation process. For unmodeled dynamics and parameter perturbations, it is assumed that there is a 30% uncertainty in the length, mass, and inertia of the ith link. And the following external disturbances are set:

$$\boldsymbol{\tau}_{ex} = \begin{cases} 6\sin(0.3\pi t) + 2\sin(\boldsymbol{q})\dot{\boldsymbol{q}} & \text{if } t < 10s \\ 30\sin(0.3\pi t) + 10\sin(\boldsymbol{q})\dot{\boldsymbol{q}} & \text{if } x \geq 10s \end{cases} \tag{44}$$

In addition, to better determine the control performance, IAE, ITAE, and RMSE metrics are used to quantify the control accuracy.

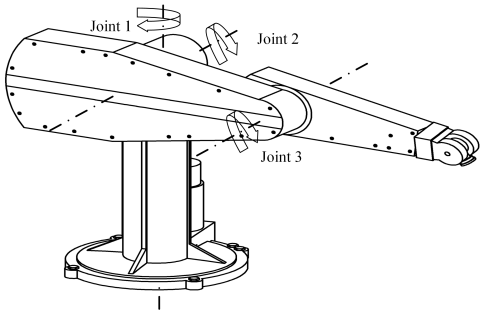

Fig. 2. The schematic diagram of the PUMA560.

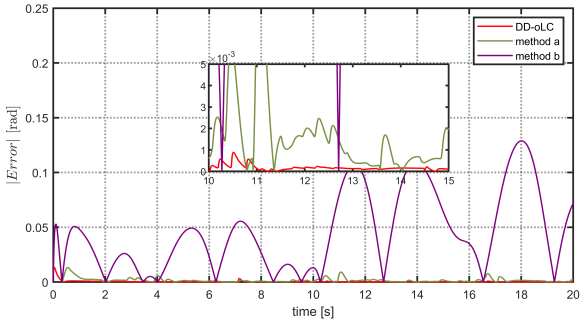

Fig. 3. The position tracking error of joint 1.

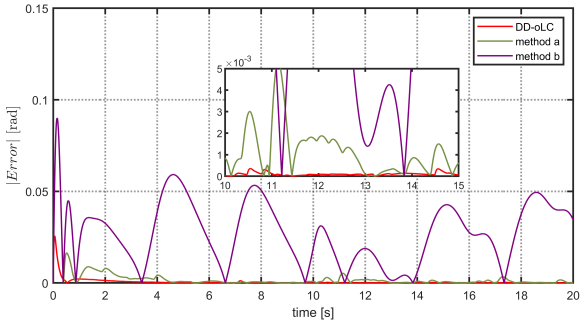

Fig. 4. The position tracking error of joint 2.

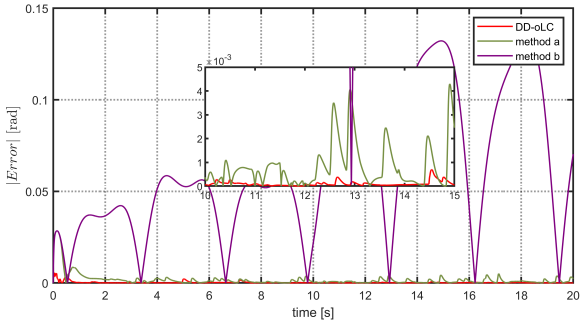

Fig. 5. The position tracking error of joint 3.

The position tracking error of joints are shown in Figs. 3, 4, 5. Method a is a PID controller, and method b is a control strategy that only uses IT2EFS without a residual compensation module. According to the results in Table I, it is clear that the control strategy proposed in this article has better tracking control performance compared to methods a and b. And the curve of the rule change is shown in Fig. 6, based on the change in the number of rules, it can be seen that DD-oLC can evolve its structure according to the dynamics of the control object throughout the control process.

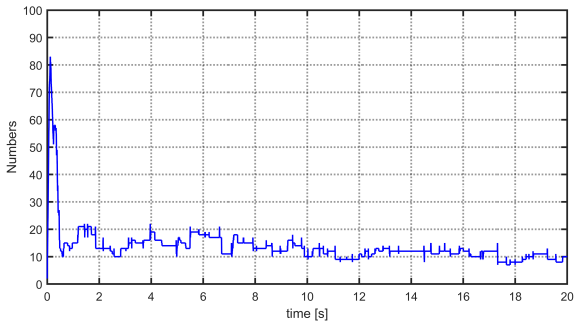

Fig. 6. The number of rules.

TABLE I
THE RESULTS OF THE PERFORMANCE COMPARISON.

Controller	RMSE(Avg.)	IAE(Avg.)	ITAE(Avg.)
method a	5.4245e-02	0.9043	0.5259
method b	0.6129e-02	0.0434	0.0110
DD-oLC	**0.1242e-02**	**0.0075**	**0.0015**

V. CONCLUSIONS

This article proposed a novel data-driven online learning controller (DD-oLC) for robotic manipulators that addresses uncertainty in practical applications. The strategy combines a feedback controller, IT2EFS, and a residual compensation module. The IT2EFS is based on its structure evolving and parameter adaptive capabilities to achieve incremental continuous learning to approximate the inverse dynamic model of the system to generate feed-forward control signals. Meanwhile, the residual error online learning module based on QRD-RLS can learn the residual error model online to provide the residual torque values during the control process. With the synergistic effect of the above two, DD-oLC can generate the final optimal control torques. The advancement of the proposed scheme was verified through simulation.

REFERENCES

[1] J. Cervantes, W. Yu, S. Salazar, and I. Chairez, "Takagi-sugeno dynamic neuro-fuzzy controller of uncertain nonlinear systems," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1601–1615, 2017.

[2] W. Yang, W. Yu, Y. Lv, L. Zhu, and T. Hayat, "Adaptive fuzzy tracking control design for a class of uncertain nonstrict-feedback fractional-order nonlinear siso systems," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3039–3053, 2021.

[3] Q. Zhou, S. Zhao, H. Li, R. Lu, and C. Wu, "Adaptive neural network tracking control for robotic manipulators with dead zone," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 12, pp. 3611–3620, 2019.

[4] S. Jia and J. Shan, "Finite-time trajectory tracking control of space manipulator under actuator saturation," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 3, pp. 2086–2096, 2020.

[5] X. Guo, H. Zhang, J. Sun, and Y. Zhou, "Fixed-time fuzzy adaptive control of manipulator systems under multiple constraints: A modified dynamic surface control approach," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 4, pp. 2522–2532, 2023.

[6] W. Sun, Y. Wu, and X. Lv, "Adaptive neural network control for full-state constrained robotic manipulator with actuator saturation and time-varying delays," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3331–3342, 2022.

[7] S. Wang, H. Shan, M. Chadli, Y. Zhu, and Y. Jiang, "Prescribed-time fuzzy adaptive control for robotic manipulators with dead zone input," *IEEE Transactions on Industrial Electronics*, vol. 72, no. 5, pp. 5012–5021, 2025.

[8] Y. Hu, H. Yan, H. Zhang, M. Wang, and L. Zeng, "Robust adaptive fixed-time sliding-mode control for uncertain robotic systems with input saturation," *IEEE Transactions on Cybernetics*, vol. 53, no. 4, pp. 2636–2646, 2023.

[9] Z.-M. Zhai, M. Moradi, L.-W. Kong, B. Glaz, M. Haile, and Y.-C. Lai, "Model-free tracking control of complex dynamical trajectories with machine learning," *Nature Communications*, vol. 14, no. 1, 2023.

[10] J. Zhao and T. Zhao, "Deep interval type-2 generalized fuzzy hyperbolic tangent system for nonlinear regression prediction," *Engineering Applications of Artificial Intelligence*, vol. 141, p. 109737, 2025.

[11] I. A. Zamfirache, R.-E. Precup, and E. M. Petriu, "Adaptive reinforcement learning-based control using proximal policy optimization and slime mould algorithm with experimental tower crane system validation," *Applied Soft Computing*, vol. 160, p. 111687, 2024.

[12] P. Qin and T. Zhao, "Knowledge guided fuzzy deep reinforcement learning," *Expert Systems with Applications*, vol. 264, 2025, cited by: 1.

[13] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 2432–2455, 2017.

[14] S. Wang, W. Chaovalitwongse, and R. Babuska, "Machine learning algorithms in bipedal robot control," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 5, pp. 728–743, 2012.

[15] Q. Su and T. Zhao, "Online data-driven incremental life-long learning control for uncertain robotic manipulators via self-evolving interval type-2 fuzzy systems," *Nonlinear Dynamics*, 2025 MAY 29 2025.

[16] A. Kumar and V. Kumar, "Evolving an interval type-2 fuzzy pid controller for the redundant robotic manipulator," *Expert Systems with Applications*, vol. 73, pp. 161–177, 2017.

[17] J. Zhao, P. Qin, Z. Mei, and T. Zhao, "A multiple attentions based multi-level hybrid-guided deep fuzzy convolutional neural network for image recognition," *IEEE Transactions on Fuzzy Systems*, pp. 1–15, 2025.

[18] H. Yang and T. Zhao, "Data-driven interval type-2 fuzzy learning controller design for tracking complex dynamical trajectories in robotic systems," *Applied Soft Computing*, vol. 179, 2025.

[19] G. Andonovski, P. Angelov, S. Blažič, and I. Škrjanc, "Robust evolving cloud-based controller (recco)," in *2017 Evolving and Adaptive Intelligent Systems (EAIS)*, 2017, pp. 1–6.

[20] H. Hu, Z. Shen, and C. Zhuang, "A pinn-based friction-inclusive dynamics modeling method for industrial robots," *IEEE Transactions on Industrial Electronics*, vol. 72, no. 5, pp. 5136–5144, 2025.

[21] D. Ge and X.-J. Zeng, "Learning data streams online — an evolving fuzzy system approach with self-learning/adaptive thresholds," *Information Sciences*, vol. 507, pp. 172–184, 2020.

[22] D. Kappler, F. Meier, N. Ratliff, and S. Schaal, "A new data source for inverse dynamics learning," ser. IEEE International Conference on Intelligent Robots and Systems, 2017, pp. 4723–4730.