Neural Network Model-Based Control for Manipulator: An Autoencoder Perspective

Zhan Li and Shuai Li, *Senior Member, IEEE*

***Abstract*— Recently, neural network model-based control has received wide interests in kinematics control of manipulators. To enhance learning ability of neural network models, the autoencoder method is used as a powerful tool to achieve deep learning and has gained success in recent years. However, the performance of existing autoencoder approaches for manipulator control may be still largely dependent on the quality of data, and for extreme cases with noisy data it may even fail. How to incorporate the model knowledge into the autoencoder controller design with an aim to increase the robustness and reliability remains a challenging problem. In this work, a sparse autoencoder controller for kinematic control of manipulators with weights obtained directly from the robot model rather than training data is proposed for the first time. By encoding and decoding the control target though a new dynamic recurrent neural network architecture, the control input can be solved through a new sparse optimization formulation. In this work, input saturation, which holds for almost all practical systems but usually is ignored for analysis simplicity, is also considered in the controller construction. Theoretical analysis and extensive simulations demonstrate that the proposed sparse autoencoder controller with input saturation can make the end-effector of the manipulator system track the desired path efficiently. Further performance comparison and evaluation against the additive noise and parameter uncertainty substantiate robustness of the proposed sparse autoencoder manipulator controller.**

***Index Terms*— Autoencoder, manipulator, neural network control.**

I. Introduction

NOWADAYS, various manipulators have been widely applied in numerous industrial applications by fulfilling heavy labors that people cannot afford to sustain [1]–[3]. By taking advantage of flexibility and redundancy of industrial manipulators, multi-level and large-scale industrial operations can be accomplished efficiently. In past decades, accurate control of manipulators toward various manipulation tasks have attracted intensive interests from research and engineering [4]–[7]. No matter what kinds of applications encountered, a manipulator always has to map its joint space into a Cartesian workspace through forward kinematics modeling and actuate its end-effector to follow the desired path through inverse kinematics resolution. Generally, the goal of kinematic control for the manipulator is to seek for a suitable control input in the joint space that can generate a motion in the workspace as desired with satisfying accuracy [8].

However, the coupled nonlinearity of the kinematic mapping exerts obstacles and burdens for kinematic control of manipulators from aspect of joint space level, and complete analytical solution for producing feasible control actions is difficult or even impossible to obtain. Some early works tried to find the joint controller solution by directly computing pseudoinverse of the Jacobian matrix of a manipulator. However, such way of solving for the suitable control input might suffer from local instability problems and unnecessary excessive computation [9], [10]. To remedy such issue, some researchers made attempts to model the nonlinear mapping for the manipulator system by constructing various efficient dynamic models to solve for suitable control actions [11]–[16], or by developing different efficient optimization paradigms and additional criteria to handle with secondary performance indices or novel convergence properties [17]–[22].

In order to approximate the highly nonlinear coupled kinematics relationship between the joint space and the Cartesian workspace for computing the suitable control input, dynamic neural network model-based control approaches can be regarded as an efficient alternative to accomplish motion modeling and perform accurate control so as to overcome shortcomings of previous conventional methods. In [23], a neural-learning-based method was proposed to control the constrained flexible manipulator with uncertainties and improved its security. In [12], dead zone issues for manipulators were well tackled adaptively by neural network. In [6], an adaptive projection neural network was utilized to control the manipulator with unknown physical parameters and shown promising tracking performance. In [24], discrete zeroing neural network models were reformulated as an equality-constrained quadratic programming to perform kinematic control of manipulators with $O(h^3)$ approximation errors. In [25], neural networks were applied to enhance adaptivity for admittance control of manipulator with environment interaction capability. In [26], the mobile manipulator's time-varying disturbances could be elegantly suppressed by a robust zeroing neural dynamics. All of these aforementioned works have shown that neural networks or their variants possess strong ability for control of robotic manipulators.

Although neural networks can promisingly deal with nonlinear relationship existing in the motion generation of manipulators, the learning ability is still needed to be enhanced

Manuscript received May 26, 2020; revised December 16, 2020, June 17, 2021, and August 14, 2021; accepted August 31, 2021. *(Corresponding author: Shuai Li.)*

Zhan Li was with the School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China. He is now with the Department of Computer Science and the Faculty of Science and Engineering, Swansea University, Swansea SA2 7QL, U.K. (e-mail: zhan.li@uestc.edu.cn).

Shuai Li is with the College of Engineering, Swansea University, Swansea, U.K. (e-mail: shuai.li@swansea.ac.uk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3109953.

Digital Object Identifier 10.1109/TNNLS.2021.3109953

if such nonlinear relationship tends to be more complicated. As a powerful tool to achieve nonlinearity approximation toward deep learning, the autoencoder-based method can be an efficient tool and has gained success in data-based learning areas [27]–[33]. The autoencoder-based methods can let the actual output to learn the reference input by taking advantages of the natural feature reconstruction ability of multiple-layer neural networks, making the errors between the input and output as small as possible. As the autoencoder uses the input layer as the encoder and the output layer as the decoder, the autoencoders can keep the main features of the manipulator kinematic relation with constraints and thus possess promising anti-noise ability. It can be as a sparse modeling to reduce dimensions. Specifically for redundant robot/manipulators, it can compress information by letting some units of the hidden layers being inactive. The performance of existing autoencoder approaches may be still largely dependent on the quality of data when applied to manipulator control, and for extreme cases with noisy data it may even fail. How to incorporate the model knowledge for the autoencoder controller design with a goal to increase the robustness and reliability remains a challenging issue. As a result, autoencoder control of a manipulator is still unexplored. Inspired by the learning capability of autoencoder in a neural network model, this work makes breakthrough by proposing a sparse autoencoder controller for kinematic control of the manipulator system. Moreover, as input saturation holds for almost all operations of manipulators but usually is ignored for analysis simplicity, it is also considered in constructing the sparse autoencoder controller. The contributions of this article are summarized as follows.

1) To the best of our knowledge, this is the first work on a unified sparse and non-sparse autoencoder framework associated with the dynamic neural network model-based method for manipulator control with and without input saturation considered.
2) This work proposes to transform the sparse autoencoder control problem and the non-sparse autoencoder control problem to an L_1 and an L_2 optimization-based control paradigm, respectively, and a resultant new primal–dual neural network solver for the sparse optimization is developed.
3) As demonstrated by the kinematic control results, the proposed sparse autoencoder controller is efficient for manipulator control and is robust even in noisy and uncertain parameter circumstances.

The remainder of this article is organized as follows. In Section II, preliminaries on the general neural network model for mapping nonlinearity is introduced, and the corresponding sparse autoencoder is formulated based on the neural network model. In Section III, a specific dynamic recurrent neural network model for autoencoding the manipulator control problem is proposed, and the dynamic optimization solver for the autoencoder controller is addressed. In Section IV, an autoencoder controller without input saturation based on the dynamic recurrent neural network model is proposed for manipulator control, including both non-sparse and sparse cases. In Section V, an autoencoder controller with saturation input based on the dynamic recurrent neural network model is proposed for manipulator control, including both non-sparse and sparse cases as well. In Section VI, simulations with comparisons are performed to validate the effectiveness of the proposed method and its robust performance. Section VII finally concludes this article.

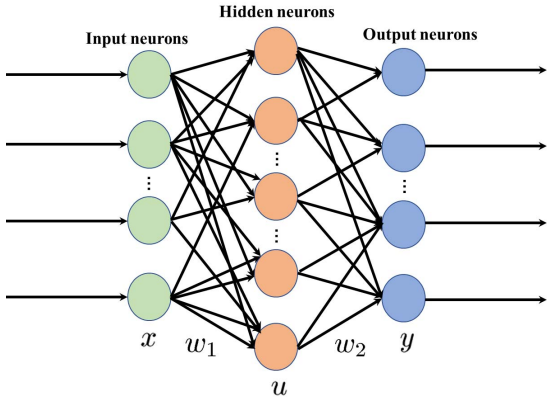

Fig. 1. Neural network model-based control for the autoencoder of nonlinear systems.

II. Preliminaries

In this section, preliminaries on the neural network model with an autoencoder paradigm for control of the general nonlinear system is introduced.

A. Neural Network Model for Autoencoder

Generally, a three-layer feedforward neural network model is used for dealing with control of the nonlinear system that the manipulator system falls into, and the architecture of the neural network is shown in Fig. 1. The discipline of information flow of the neural network model is governed by

$$\begin{cases} u = f(w_1, x) \\ y = g(w_2, u) \end{cases} \tag{1}$$

where $x \in R^m$ is the model input vector of the neural network, $u \in R^n$ is the intermediate variable vector of the hidden layer of the neural network that stores the nonlinear-mapping information from the input layer, $y \in R^l$ is the model output vector of the neural network that presents the nonlinear-mapping information from the hidden layer, $f(\cdot)$:$R^m \to R^n$ denotes the nonlinear mapping function array between the input layer and the hidden layer of the neural network, $g(\cdot)$:$R^n \to R^l$ denotes the nonlinear mapping function array between the hidden layer and the output layer of the neural network, $w_1 \in R^{n\times m}$ denotes the weight matrix for linking the model input vector $x \in R^m$ and the hidden layer variable vector $u \in R^n$, and $w_2 \in R^{l\times n}$ denotes the weight matrix for linking the hidden layer variable vector $u \in R^n$ and the model output vector $y \in R^l$. The neural network in Fig. 1 makes the error function associated with y, x, and u as small as possible by adjusting weight matrices w_1 and w_2 through model learning/training. As an autoencoder paradigm is utilized for the aforementioned neural

network model, the weights w_1 and w_2 of the neural network are trained such that $y \approx x$ with $m = l$. The first and second layers (i.e., the input layer and the hidden layer) encode the model input variable vector $x \in R^m$ into the hidden layer variable vector $u \in R^n$ and then the second and the third layers (i.e., the hidden layer and the output layer) decode the hidden layer variable vector $u \in R^n$ into the model output variable vector $y \in R^l$. Especially, when $m > n$, we can use $u \in R^n$ in low dimensions to represent $x \in R^m$ in high dimensions, and $x \in R^m$ can be reconstructed through the trained neural network model's output $y \in R^l$. In this way, the autoencoder can be regarded as a compressed sensing model which can be further formulated in a sparse manner.

B. Sparse Autoencoder

For the general three-layer neural network model (1) to encode and decode the nonlinear system for manipulator control in a sparse manner, in this article, a sparse autoencoder with an additional constraint to make the hidden layer variable u sparse is proposed, which can be formulated as the following optimization problem:

$$\arg\min_{w_1, w_2} \sum_{i=1}^{m} \|y_i - x_i\|_2^2 + \sum_{j=1}^{n} k_0 \|u_i\|_1 \tag{2}$$

where y_i and x_i, respectively, denote the ith model output and input of the neural network, $u_i \in \Omega$ denotes the ith control action constrained from the solution set Ω, $k_0 > 0$ denotes the scaling parameter, $\|\cdot\|_2$ denotes the L_2 norm of a vector, and $\|\cdot\|_1$ denotes the L_1 norm of a vector. It can be evidently seen here that the neural network variables x_i, u, and y_i for constructing nonlinear mapping are static without their time-derivative information of some orders involved (i.e., $\dot{y}_i$, $\dot{x}_i$, and $\dot{u}$ are not appearing in the neural network model), which means that the neural network model (1) may lack sufficient dynamic information of the manipulator system. So, the sparse autoencoder for time sequence of nonlinear system modeling and control is still not clear. In Section III, a dynamic autoencoder neural network model for kinematic control of the manipulator system is presented.

III. Dynamic Autoencoder Neural Network Model for Manipulator Control and Optimization Solver

A. Dynamic Neural Network Model for Autoencoder Controller

In order to remedy the static autoencoder based on the aforementioned neural network for control of manipulator system, in this article, we propose a dynamic autoencoder recurrent neural network for manipulator system control as follows:

$$\begin{cases} \dot{u} = f(u, x, w_1) \\ \dot{y} = g(y, u, w_2). \end{cases} \tag{3}$$

Different from the neural network model (1), the neural network model (3) is of a recurrent type and contains time-derivative information $\dot{u}$ and $\dot{y}$ that have additional recursive information for the hidden and output layers, which enables (3) can describe the dynamic process of the manipulator system and it is different from the feedforward neural network model in topology [34].

In this work, our goal is to perform kinematic control of the manipulator system whose motion is depicted by the following velocity kinematics equation:

$$\dot{r} = J\dot{\theta} \tag{4}$$

where $J \in R^{3\times n}$ denotes the Jacobian matrix of the manipulator, $r \in R^3$ denotes the Cartesian space variable of the end-effector, and $\theta \in R^n$ denotes the joint angle variable. The forward kinematics relation is governed by $r = f(\theta)$ where the mapping function $f(\cdot)$ is coupled nonlinear, and the Jacobian matrix J is derived from $\partial f/\partial\theta$. The modeling of such forward kinematics of the manipulator is given by the transformation matrix between the base coordinate and the end-effector coordinate

$$T(\theta) = T_1(\theta_1), \ldots, T_i(\theta_i), \ldots, T_n(\theta_n) \tag{5}$$

where the homogeneous transformation matrix is

$$T_i(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where α_i, a_i, d_i, and θ_i are the Denavit–Hartenberg (D–H) parameters for the manipulator.

The dynamic model (4) for kinematic control can be described by the autoencoder recurrent neural network (3) such that the desired path variable $x := r_d$ can approximate $y := r$ with control input $u := \dot{\theta}$. The proposed autoencoder based on the recurrent neural network (3) for control of the manipulator is shown by Fig. 2. Under this situation, the reference path variable r_d is encoded as the control input u, and the control input u is decoded to the path variable r. Through the learning process of the dynamic recurrent neural network (3), the control input u is optimally obtained. For the autoencoder controller based on the recurrent neural network model for the manipulator system, we consider the four autoencoder control paradigms in both non-sparse and sparse cases in the ensuing sections as follows: 1) a non-sparse autoencoder controller without saturation control input; 2) a sparse autoencoder controller without saturation control input; 3) a non-sparse autoencoder controller with saturation control input; and 4) a sparse autoencoder controller with saturation control input. These autoencoder controllers are obtained by solving the corresponding optimization problems.

B. Dynamic Optimization Solver

As the autoencoder controller is established based on the dynamic recurrent neural network (3), the optimization solver should also be dynamic to compute the controller. Now, we present the dynamic optimization solver as follows.

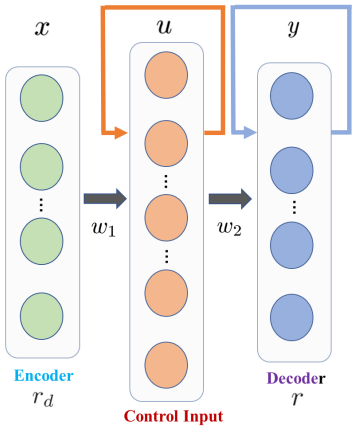

Fig. 2. Proposed autoencoder based on the recurrent neural network for kinematic control of the manipulator system.

1) Model Description: For the convex objective function $F(u)$ that is associated with the dynamic autoencoder neural network for manipulator control, the optimization can be formulated as follows:

$$\begin{aligned} \min \quad & F(u) \\ \text{s.t.} \quad & G(u) \leq 0 \end{aligned} \tag{6}$$

where $F(\cdot)$:$R^n \rightarrow R$ and $G(\cdot)$:$R^n \rightarrow R^n$ denote the mapping functions for the control input u.

To solve this convex optimization, according to the Karush–Kuhn–Tucker (KKT) condition [35] and letting the partial derivatives of the Lagrange function $L(u, \lambda)$ being zero, we equivalently obtain the following dynamic equations as the optimization solver:

$$\begin{cases} \epsilon \dot{u} = u - P_\Omega\left(u - \dfrac{\partial L(u,\lambda)}{\partial u}\right) \\ \epsilon \dot{\lambda} = G(u) \end{cases} \tag{7}$$

where $\epsilon > 0$ is the parameter to scale the convergence to the solution, and $P_\Omega(\cdot)$ denotes linear projection to a normal cone Ω, that is, $P_\Omega(u > u_+) = u_+$, $P_\Omega(u \leq u_-) = u_-$, and $P_\Omega(u_- < u < u_+) = u$, where u_- and u_+, respectively, denote the lower and upper bounds for the input of the linear projection operator. As the objective function $F(u)$ is strictly convex, existence and uniqueness of the optimal solutions can be guaranteed. Since the differential kinematics of the manipulator can be established through the corresponding D–H parameters, it is not necessary for the autoencoder controller based on the primal–dual neural network solver to satisfy persistent conditions, because the system identification of the solution process is not necessary when the convergence parameter $\epsilon > 0$ is configured already and the bounds for the projection operator are known. The parameter $\epsilon > 0$ is the only parameter for tuning to scale the convergence rate. The autoencoder-based control loop for the manipulator based on the dynamic optimization solver is shown by Fig. 3.

Especially, when the objective function $F(u)$ is a quadratic function with respect to u, that is, $F(u) = u^{\mathrm{T}}Qu/2 + q^{\mathrm{T}}u$,

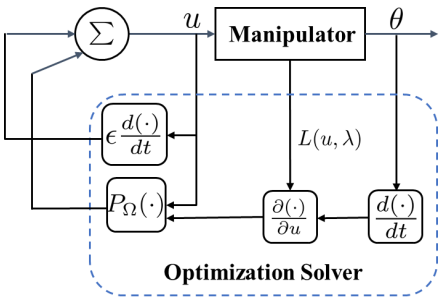

Fig. 3. Control loop for the manipulator based on the dynamic optimization solver for the autoencoder controller.

the dynamic optimization solver can further become

$$\begin{cases} \epsilon \dot{u} = u - P_\Omega(u - Qu - q) \\ \epsilon \dot{\lambda} = G(u) \end{cases} \tag{8}$$

where Q is a positive-definite matrix.

By letting $\dot{z} = g(t, z)$, where $z := [u \;\; \lambda]^{\mathrm{T}}$, the corresponding ordinary-differential-equation (ODE) updating law (the fourth-order Runge–Kutta method) is

$$z_{k+1} = z_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{9}$$

with

$$\begin{cases} k_1 = g(t_k, y_k) \\ k_2 = g\left(t_k + \dfrac{h}{2}, y_k + \dfrac{k_1}{2}\right) \\ k_3 = g\left(t_k + \dfrac{h}{2}, y_k + \dfrac{k_2}{2}\right) \\ k_4 = g(t_k + h, y_k + k_3) \end{cases}$$

where $h > 0$ denotes the step size of the ODE iterative updating equations, and t_k denotes the kth time sampling. Algorithm 1 shows the data-driven version of the autoencoder controller for manipulator control based on the aforementioned ODE updating law.

2) Theoretical Analysis: Due to its convexity of the optimization problem, the aforementioned expression $\partial L/\partial u \in \Omega$ includes the normal cone operator on u. Recall the property on the linear projection to a normal cone, the solution of optimization problem is equivalent to the solution of the linear projection equations $P_\Omega(\cdot)$. The dynamic optimization solver based on the primal–dual neural network model solve the linear projection equations when the equilibrium point u is reached [36], [37]. Therefore, one can define a Lyapunov function $V(u) = u^{\mathrm{T}}u/2 \geq 0$ and obtain its time derivative $\dot{V} = -u^{\mathrm{T}}[u - P_\Omega(u - (\partial L(u, \lambda)/\partial u))]/\epsilon \leq -u^{\mathrm{T}}u/\epsilon \leq 0$, thus the convergence to the equilibrium point u can be achieved.

Remark 1: In order to approximate the highly nonlinear coupled kinematics relationship between the joint space and the Cartesian workspace for computing the suitable control input, the inverse kinematic resolution is performed in the joint velocity level based on inverse control synthesized by solving the pseudoinverse of Jacobian matrix. Such model-based approach may neglect the physical limits on joint control action (e.g., joint velocity limits) and cannot produce sparse

Algorithm 1 Data-Driven Autoencoder Control Algorithm for Kinematic Control of Manipulator

Input : Forward kinematic equation $f(\theta(t_k), t_k)$; Jacobian matrix $J(\theta(t_k), t_k)$; desired path $r_d(t_k)$; convergence parameter $\epsilon > 0$, joint velocity boundaries $\dot{\theta}_+$ and $\dot{\theta}_-$; initial joint angle $\theta(0)$; step size h; tolerance error ε

Output: Resolved/updated joint angular velocity $\dot{\theta}(t_{k+1})$

while $|r(t_k) - r_d(t_k)| > \varepsilon$ **do**
  if *Non-sparse autoencoder is used* **then**
    $\Omega := \Omega_1$
  end
  if *Sparse autoencoder is used* **then**
    $\Omega := \Omega_2$
  end
  $\epsilon g(t_k, \begin{bmatrix}\dot{\theta}(t_k)\\ \lambda(t_k)\end{bmatrix}) = \begin{bmatrix}\dot{\theta}(t_k) - P_\Omega(\dot{\theta}(t_k) - \frac{\partial L(\dot{\theta}(t_k), \lambda(t_k))}{\partial \dot{\theta}(t_k)})\\ G(\dot{\theta}(t_k))\end{bmatrix}$
  $k_1 = g(t_k, \begin{bmatrix}\dot{\theta}(t_k)\\ \lambda(t_k)\end{bmatrix})$
  $k_2 = g(t_k + \frac{h}{2}, \begin{bmatrix}\dot{\theta}(t_k)\\ \lambda(t_k)\end{bmatrix} + \frac{k_1}{2})$
  $k_3 = g(t_k + \frac{h}{2}, \begin{bmatrix}\dot{\theta}(t_k)\\ \lambda(t_k)\end{bmatrix} + \frac{k_2}{2})$
  $k_4 = g(t_k + h, \begin{bmatrix}\dot{\theta}(t_k)\\ \lambda(t_k)\end{bmatrix} + k_3)$
  $\begin{bmatrix}\dot{\theta}(t_{k+1})\\ \lambda(t_{k+1})\end{bmatrix} = \begin{bmatrix}\dot{\theta}(t_k)\\ \lambda(t_k)\end{bmatrix} + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$
  $\dot{r}(t_{k+1}) = J(\theta(t_{k+1}), t_{k+1})\dot{\theta}(t_{k+1})$
end

solutions by compressing redundant information of manipulators. Under these considerations, the autoencoder controller with physical constraints on joints in both non-sparse and sparse cases is proposed in this work. Model-based methods usually lack adaptive learning ability. Compared with model-based methods, the autoencoder controller can let the actual output to learn the reference input by taking advantages of the natural feature reconstruction ability of multiple-layer neural networks, making the errors between the input and output as less as possible. As it uses the input layer as the encoder, it also uses the output layer as the decoder. The autoencoders can keep the main features of the manipulator kinematic relation with constraints and thus possess promising anti-noise ability. Table I shows the comparison between the proposed method and other methods.

IV. Autoencoder Controller Without Input Saturation

In this section, the autoencoder controller without input saturation is introduced. Both non-sparse and sparse types of autoencoder controllers are exploited, and the corresponding optimization solvers are presented as well.

TABLE I
Comparison With Other Methods on Manipulator Control

	Saturation	Anti-Noise	Sparsity
This paper	Yes	Yes	Yes
Pseudo-inverse methods in [38]	No	No	No
Method in [39]	Yes	No	No
Method in [40]	Yes	Yes	No

A. Non-Sparse Autoencoder Controller Without Input Saturation

First, the non-sparse autoencoder controller without input saturation is proposed to be solved by the following optimization formulation:

$$\min \ \|u\|_2^2/2 \tag{10}$$

$$\text{s.t. } Ju + k(r - r_d) - \dot{r}_d = 0 \tag{11}$$

where $k > 0$ denotes the scaling parameter for converge of solution. To solve the optimization problem above, one can construct the Lagrange function as follows:

$$L(u, \lambda) = \|u\|_2^2/2 + \lambda^{\mathrm{T}}[Ju + k(r - r_d) - \dot{r}_d] \tag{12}$$

where $\lambda \in R^n$ denotes the Lagrange multiplier. According to the aforementioned design principle of primal–dual neural network in Section III-B, the corresponding optimization solver is

$$\begin{cases} \epsilon \dot{u} = -u + P_\Omega\left(u - \dfrac{\partial L(u, \lambda)}{\partial u}\right) \\ \epsilon \dot{\lambda} = Ju + k(r - r_d) - \dot{r}_d \end{cases} \tag{13}$$

where $P_\Omega(\cdot)$ denotes the linear projection operator. Here, as the control input u is without saturation, the lower bound and the upper bound can be set, respectively, as $u_- = -\infty$ and $u_+ = +\infty$. Practically, it is commonly configured that $u_- = -N$ and $u_+ = N$, where $N > 0$ is a very large real-valued constant.

As the partial derivative of $L(u, \lambda)$ with respect to u is

$$\frac{\partial L(u, \lambda)}{\partial u} = u + J^{\mathrm{T}}\lambda. \tag{14}$$

The primal–dual neural network as the optimization solver of the control input u without saturation further becomes

$$\begin{cases} \epsilon \dot{u} = -u + P_\Omega(-J^{\mathrm{T}}\lambda) \\ \epsilon \dot{\lambda} = Ju + k(r - r_d) - \dot{r}_d \end{cases} \tag{15}$$

where the convergence of the solution $r \to r_d$ can be modulated by parameters ϵ and k, and the solution set Ω can be seen without solution boundary for the linear projection. The number of neurons of such network is $n + 3$.

B. Sparse Autoencoder Controller Without Input Saturation

To solve for the control input u without input saturation based on the sparse autoencoder recurrent neural network, we have to satisfy the following two conditions: 1) the solved control input u is sparse and 2) the convergence of the solution $\lim_{t \to +\infty} r = r_d$. In this article, we propose to transform the unsaturation control problem in a sparse perspective into the following constrained L_1 sparse optimization problem:

$$\min \quad \|u\|_1 \tag{16}$$

$$\text{s.t. } \ Ju = -k(r - r_d) + \dot{r}_d. \tag{17}$$

As the sparse optimization formulation is without saturation bounds $[u_-, u_+]$, thus the sparse optimization problem further becomes

$$\begin{aligned} &\min \ \sum_{i=1}^{n} \alpha_i \\ &\text{s.t. } Ju = -k(r - r_d) + \dot{r}_d \end{aligned} \tag{18}$$

where α_i denotes the newly involved variable to be optimized, and the optimization above can be reformulated as

$$\begin{aligned} &\min \ h^{\mathrm{T}}\alpha \\ &\text{s.t. } Ju = -k(r - r_d) + \dot{r}_d \end{aligned} \tag{19}$$

where $h = [1, 1, \ldots, 1]^{\mathrm{T}} \in R^n$ denotes the coefficient vector, and $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n]$ denotes the variable vector to be optimized.

In order to solve the sparse optimization problem without input saturation, similarly, it is needed to construct the primal–dual neural network solver, and the following Lagrange function is defined:

$$L = h^{\mathrm{T}}\alpha + \lambda^{\mathrm{T}}(Ju + k(r - r_d) - \dot{r}_d).$$

Differentiate the Lagrange function above with respect to the control input u, α, and λ, one can get the following equations:

$$\begin{cases} \dfrac{\partial L}{\partial u} = J^{\mathrm{T}}\lambda \\ \dfrac{\partial L}{\partial \alpha} = h \\ \dfrac{\partial L}{\partial \lambda} = Ju + k(r - r_d) - \dot{r}_d. \end{cases} \tag{20}$$

Based on the design principle of the primal–dual neural network in Section III-B, we have

$$\begin{cases} \epsilon \begin{bmatrix} \dot{u} \\ \dot{\alpha} \end{bmatrix} = -\begin{bmatrix} u \\ \alpha \end{bmatrix} + P_{\widetilde{\Omega}}\left\{ \begin{bmatrix} u \\ \alpha \end{bmatrix} - \begin{bmatrix} \dfrac{\partial L}{\partial u} \\ \dfrac{\partial L}{\partial \alpha} \end{bmatrix} \right\} \\ \epsilon\dot{\lambda} = Ju + k(r - r_d) - \dot{r}_d \end{cases} \tag{21}$$

with its number of neurons being $2n + 3$.

For the primal–dual neural network solver (21) for sparse optimization with unsaturated control input u, the corresponding linear piecewise projection function for the solution set cone $\widetilde{\Omega} = \bigcup_{i=1}^{n} \widetilde{\Omega}_i$ is constructed by

$$P_{\widetilde{\Omega}_i}\left(\begin{bmatrix} u_i \\ \alpha_i \end{bmatrix}\right) = \begin{cases} \begin{bmatrix} u_i \\ \alpha_i \end{bmatrix}, & |u_i| \le \alpha_i \\ \begin{bmatrix} (u_i + \alpha_i)/2 \\ (u_i + \alpha_i)/2 \end{bmatrix}, & u_i \ge |\alpha_i| \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, & |u_i| \le -\alpha_i \\ \begin{bmatrix} (u_i - \alpha_i)/2 \\ (-u_i + \alpha_i)/2 \end{bmatrix}, & u_i \le -|\alpha_i|. \end{cases} \tag{22}$$

The solution set $\widetilde{\Omega}$ is divided into four sub-parts with a variable α, and the no saturation bounds $[u_-, u_+]$ associated with the control input u are involved in the solution set cone $\widetilde{\Omega}$. As compared with the original solution set Ω in the non-sparse case, the solution set $\widetilde{\Omega}$ brings in a new variable α to limit the control input u to preserve sparsity.

V. Autoencoder Controller With Input Saturation

In this section, the input saturation, which may be more practical for actuators in servo control modes [14], [41]–[43], is investigated in the autoencoder controller establishment. Both non-sparse and sparse controllers are proposed with input saturation and the corresponding optimization solvers are presented.

A. Non-Sparse Autoencoder Controller With Input Saturation

In most of the applications, the control input u has to fall into the saturation bounds $[u_-, u_+]$ for kinematic control of manipulator systems, due to the power limitations of joint actuators and physical limitations of joint/link mechanisms. Under these considerations, the non-sparse autoencoder controller can be solved through the following saturation-constrained optimization formulation:

$$\begin{aligned} &\min \ \|u\|_2^2/2 \\ &\text{s.t. } Ju = -k(r - r_d) + \dot{r}_d \\ &\quad u^- \le u \le u^+. \end{aligned} \tag{23}$$

Its equivalent form for optimization can be

$$\begin{aligned} &\min \ \|u\|_2^2/2 + \|Ju + k(r - r_d) - \dot{r}_d\|_2^2/2 \\ &\text{s.t. } Ju + k(r - r_d) - \dot{r}_d = 0 \\ &\quad u_- \le u \le u_+. \end{aligned} \tag{24}$$

To solve the optimization problem above, we construct the following Lagrange function:

$$\begin{aligned} L(u, \lambda) = \|u\|_2^2/2 + \|Ju + k(r - r_d) - \dot{r}_d\|_2^2/2 \\ +\lambda^{\mathrm{T}}(Ju + k(r - r_d) - \dot{r}_d) \end{aligned} \tag{25}$$

where $\lambda \in R^n$ denotes the Lagrange multiplier. Therefore, similarly, according to the design principle of primal–dual neural network, we construct the following optimization solver model:

$$\begin{cases} ll\epsilon\dot{u} = -u + P_{\Omega}\left(u - \dfrac{\partial L(u, \lambda)}{\partial u}\right) \\ \epsilon\dot{\lambda} = Ju + k(r - r_d) - \dot{r}_d. \end{cases} \tag{26}$$

Since the partial derivative of $L(u, \lambda)$ with respect to u is

$$\frac{\partial L(u, \lambda)}{\partial u} = u + J^{\mathrm{T}}[Ju + k(r - r_d) - \dot{r}_d] + J^{\mathrm{T}}\lambda \tag{27}$$

which can further produce

$$\begin{cases} \epsilon\dot{u} = -u + P_{\Omega}\big(-J^{\mathrm{T}}(Ju + k(r - r_d) - \dot{r}_d) - J^{\mathrm{T}}\lambda\big) \\ \epsilon\dot{\lambda} = Ju + k(r - r_d) - \dot{r}_d. \end{cases} \tag{28}$$

Here, the solution set Ω is with the saturation bound $[u_-, u_+]$ involved, and one can regard that the input into the linear projection function $P_{\Omega}(\cdot)$ is with restrictions explicitly. As the optimization solver is not for sparse case, the solution set Ω is not be divided by the newly added variables. The number of neurons of such network is $n + 3$.

B. Sparse Autoencoder Controller With Input Saturation

To solve for the control input u by the sparse autoencoder with input saturation based on the recurrent neural network model, the following constrained L_1 sparse optimization is proposed:

$$\begin{aligned} &\min \ \|u\|_1 \\ &\text{s.t. } Ju = -k(r - r_d) + \dot{r}_d \\ &\quad u_- \le u \le u_+. \end{aligned} \tag{29}$$

To make the optimization problem solved by the primal–dual neural network well, we propose a new equivalent optimization formulation as follows:

$$\begin{aligned} &\min \ \sum_{i=1}^{n} \beta_i \\ &\text{s.t. } Ju + k(r - r_d) - \dot{r}_d = 0 \\ &\quad -\beta_i \le u_i \le \beta_i \\ &\quad u_- \le u \le u_+ \end{aligned} \tag{30}$$

where β_i denotes the newly added variable to be optimized.

The optimization problem above can be further equivalently rewritten as the following optimization formulation:

$$\begin{aligned} &\min \ h^{\mathrm{T}}\beta \\ &\text{s.t. } Ju + k(r - r_d) - \dot{r}_d = 0 \\ &\quad -\beta \le u \le \beta \\ &\quad u_- \le u \le u_+ \end{aligned} \tag{31}$$

where $\beta = [\beta_1, \beta_2, \ldots, \beta_n]^{\mathrm{T}} \in R^n$ denote the variable vector to be optimized. The aforementioned sparse optimization formulation can be rewritten as follows:

$$\begin{aligned} &\min \ h^{\mathrm{T}}\beta \\ &\text{s.t. } Ju + k(r - r_d) - \dot{r}_d = 0 \\ &\quad u \in \overline{\Omega} \end{aligned} \tag{32}$$

where we define the new solution set $\overline{\Omega} = \Omega \bigcap \{u_- \le u \le u_+\} \bigcap \{-\beta \le u \le \beta\}$. Optimization formulation (32) is ready to solve the L_1-norm-based sparse optimization problem with saturation considered, through additionally introducing the variable β to restrict the control input u simultaneously.

In order to solve the optimization problem (32) by constructing primal–dual neural networks, we have to define the following Lagrange function:

$$L = h^{\mathrm{T}}\beta + \lambda^{\mathrm{T}}(Ju + k(r - r_d) - \dot{r}_d).$$

Differentiate the Lagrange function above with respect to the unknown variables u, β, and λ, we would get the following equations:

$$\begin{cases} \dfrac{\partial L}{\partial u} = J^{\mathrm{T}}\lambda \\ \dfrac{\partial L}{\partial \beta} = h \\ \dfrac{\partial L}{\partial \lambda} = Ju + k(r - r_d) - \dot{r}_d. \end{cases} \tag{33}$$

According to the aforementioned design procedure of the primal–dual neural network for solving optimization problems, we can have the following new primal–dual neural network solver for the sparse optimization of saturated control input:

$$\begin{cases} \epsilon \begin{bmatrix} \dot{u} \\ \dot{\beta} \end{bmatrix} = -\begin{bmatrix} u \\ \beta \end{bmatrix} + P_{\overline{\Omega}}\left\{ \begin{bmatrix} u \\ \beta \end{bmatrix} - \begin{bmatrix} \dfrac{\partial L}{\partial u} \\ \dfrac{\partial L}{\partial \beta} \end{bmatrix} \right\} \\ \epsilon\dot{\lambda} = Ju - k(r - r_d) + \dot{r}_d \end{cases} \tag{34}$$

where the solution set cone $\overline{\Omega} = \bigcup_{i=1}^{n} \overline{\Omega}_i$ with $\overline{\Omega}_i = \begin{bmatrix} u_i \\ \beta_i \end{bmatrix}$ with boundaries $u_{-,i} \le u_i \le u_{+,i}$ and $-\beta_i \le u_i \le \beta_i$. For simplicity, we can let $u_{-,i} + u_{+,i} = 0$. The number of neurons of such network is $2n + 3$.

For the properties of the newly proposed linear piecewise projection function $P_{\overline{\Omega}}(\cdot)$ with the new divided solution set $\overline{\Omega}$, we have

$$P_{\overline{\Omega}}\left(\begin{bmatrix} u \\ \beta \end{bmatrix}\right) = \bigcup_{i=1}^{n} P_{\overline{\Omega}_i}\left(\begin{bmatrix} u_i \\ \beta_i \end{bmatrix}\right) \tag{35}$$

and its subparts can be expanded as follows:

$$P_{\overline{\Omega}_i}\left(\begin{bmatrix} u_i \\ \beta_i \end{bmatrix}\right) = \begin{cases} \begin{bmatrix} u_i \\ \beta_i \end{bmatrix}, & |u_i| \le u_{+,i} \ \text{ and } |u_i| \le \beta_i \\ \begin{bmatrix} u_{+,i} \\ \beta_i \end{bmatrix}, & u_i \ge u_{+,i} \ \text{ and } \beta_i \ge u_{+,i} \\ \begin{bmatrix} -u_{+,i} \\ \beta_i \end{bmatrix}, & u_i \le -u_{+,i} \ \text{ and } \beta_i \ge u_{+,i} \\ \begin{bmatrix} u_{+,i} \\ u_{+,i} \end{bmatrix}, & \alpha_i \le u_{+,i} \ \text{ and } u_i \ge -\beta_i + 2u_{+,i} \\ \begin{bmatrix} -u_{+,i} \\ u_{+,i} \end{bmatrix}, & \alpha_i \le u_{+,i} \ \text{ and } u_i \le \beta_i - 2u_{+,i} \\ \begin{bmatrix} (u_i + \beta_i)/2 \\ (u_i + \beta_i)/2 \end{bmatrix}, & |\beta_i| \le u_i \le -\beta_i + 2u_{+,i} \\ \begin{bmatrix} (u_i - \beta_i)/2 \\ (-u_i + \beta_i)/2 \end{bmatrix}, & \beta_i - 2u_{+,i} \le u_i \le -|\beta_i| \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, & |u_i| \le -\beta_i. \end{cases} \tag{36}$$

The newly proposed linear projection function $P_{\overline{\Omega}}(\cdot)$ considers both the saturation bounds $[u_-, u_+]$ and the optimization variable β_i to form a new combined solution set cone $\overline{\Omega}$ which divides the solution plane into eight subparts. Different from the non-sparse case, we can see that the final number of variables to be optimized doubles in the sparse case to preserve sparsity and saturation for optimization solution.

VI. Results

In this section, the manipulator control results based on the proposed autoencoder controller in non-sparse and sparse cases are presented. The normal, noisy, and parameter uncertainty situations are considered to validate the efficiency and

the robustness of the authoencoder controllers for manipulator control.

A. Simulation Setup and Parameter Configuration

For kinematic control of the manipulator system for tracking the desired paths, according to the aforementioned theoretical results on the autoencoder controller based on the recurrent neural network model, the following three representative evaluation scenarios are investigated: 1) evaluation of tracking control performance in normal situation; 2) evaluation of robustness of the tracking control performance against additive noise; and 3) evaluation of robustness of the tracking control performance against parameter uncertainty.

For the first evaluation scenario, the following four cases are considered: 1) non-sparse solution without saturation control input; 2) non-sparse solution with saturation control input; 3) sparse solution without saturation control input; and 4) sparse solution with saturation control input. The forward kinematic modeling of the manipulator system is established by using the kinematic chains based on the D–H parameter table in [3]. The reference paths of the end-effector of the manipulator are, respectively, planned as a circular path with its radius being 0.15 m and a square path with its side length being 0.20 m. For each of the cases, both circular type and square type path tracking tasks are performed. The convergence parameter ϵ of the primal–dual neural network solver for the proposed autoencoder controller is set as 0.001, and the scaling parameter of the autoencoder controller is set as $k = 10$. The initial joint angles $\theta(0)$ of the manipulator are randomly generated within range $[-\pi/2, \pi/2]$. For cases 1) and 3), the saturation bounds are set as very large values [>200 (rad/s)] to indicate the magnitudes of the control input unlimited. For case 2), the saturation bounds are set as $[-0.2, 0.2]$ (rad/s), and for case 4), the configuration of the saturation bounds for the control input is shown in Table II. ODE-45 solver with time interval 0.1 s for updating the dynamic equations of the proposed method is used.

B. Normal Situation

In the normal situation, as no noise and parameter uncertainty appear in the kinematic modeling and control of the manipulator, all the control inputs in the recurrent neural network-based autoencoder are not contaminated by noises, and the parameters in the plant for control are known or configured precisely.

1) Non-Sparse Solution Without Saturation Control Input: For the first case, the autoencoder controller is obtained in a non-sparse encode and decode process based on the recurrent neural network model, and the corresponding optimization solver is formulated in an L_2-norm paradigm. Therefore, the non-sparse optimization solver (26) is adopted to compute the unsaturated control input u. Figs. 4 and 5 show the unsaturated control inputs u solved by the non-sparse optimization solver for a circular path and a square path tracking, respectively. One can observe that, the control input u have oscillations to some extent. With such unsaturation control input and non-sparse autoencoder process, the tracking control performance of the end-effector of the manipulator with position errors is shown by Fig. 6. As seen from Fig. 6, the end-effector of the manipulator can track the circular and square paths well as expected, and the position errors $[e_x, e_y, e_z]$ synthesized by the unsaturated non-sparse autoencoder controller can reach to around 8×10^{-4} m. All of these results first demonstrate the efficiency of the proposed non-sparse autoencoder controller based on the dynamic recurrent neural network for kinematic control of the manipulator system with unsaturated control input.

TABLE II

SATURATION BOUND $[u_{-,i}, u_{+,i}]$ THAT THE SPARSE AUTOENCODER CONTROLLER WITH INPUT SATURATION HAS TO FALL INTO FOR KINEMATIC CONTROL OF THE MANIPULATOR SYSTEM

Control input u_i	Circle path case	Square path case
u_1	$[-0.2, 0.2]$	$[-0.1, 0.1]$
u_2	$[-0.2, 0.2]$	$[-0.1, 0.1]$
u_3	$[-0.2, 0.2]$	$[-0.1, 0.1]$
u_4	$[-0.2, 0.2]$	$[-0.1, 0.1]$
u_5	$[-0.1, 0.1]$	$[-0.1, 0.1]$
u_6	$[-0.1, 0.1]$	$[-0.1, 0.1]$
u_7	$[-0.05, 0.05]$	$[-0.05, 0.05]$

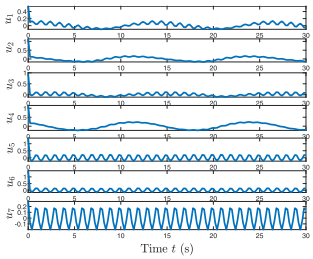

Fig. 4. Control input u without saturation for control of the manipulator system to track the circular path in non-sparse autoencoder case.

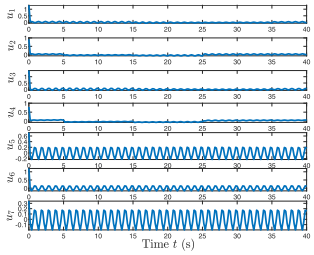

Fig. 5. Control input u without saturation for control of the manipulator system to track the square path in non-sparse autoencoder case.

2) Non-Sparse Solution With Saturation Control Input: Next, we investigate the non-sparse autoencoder controller for kinematic control of the manipulator system with input saturation. In this case, the autoencoder controller is obtained in a non-sparse encode and decode process based on the recurrent neural network model, and the corresponding optimization solver is also formulated in the L_2-norm paradigm. Therefore, the non-sparse optimization solver (28) is adopted to compute the saturated control input u, and the saturation boundary is set as $[u_{-,i}, u_{+,i}] = [-0.2, 0.2]$ (rad/s). Figs. 7 and 8 show the unsaturated control inputs u solved by the non-sparse optimization solver for circular path and square path tracking respectively. One can obviously observe that, the control input u have less oscillations, as compared with the

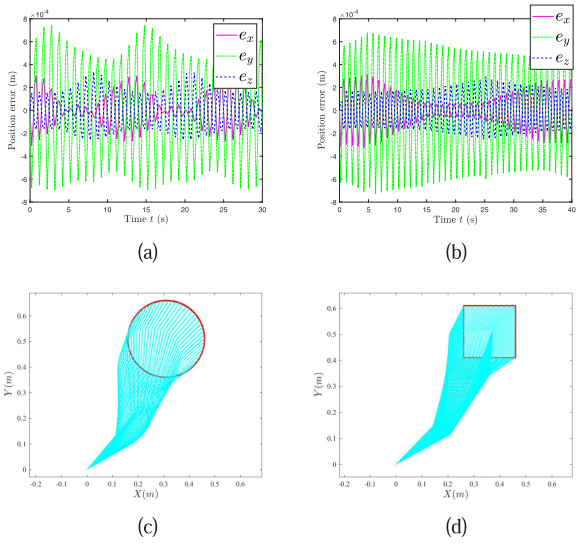

Fig. 6. Tracking performance and the position errors for the manipulator system to track the circular and square paths by the non-sparse autoencoder controller without input saturation. (a) Circle. (b) Square. (c) Circle. (d) Square.

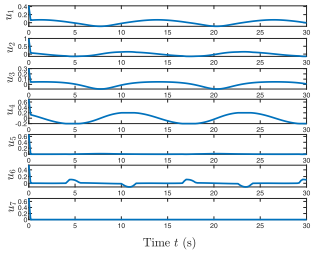

Fig. 7. Control input u with saturation for control of the manipulator system to track the circular path in the non-sparse autoencoder case.

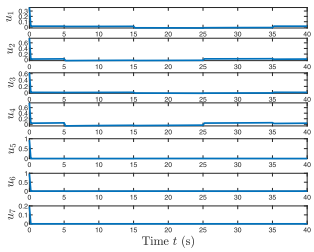

Fig. 8. Control input u with saturation for control of the manipulator system to track the circular path in the non-sparse autoencoder case.

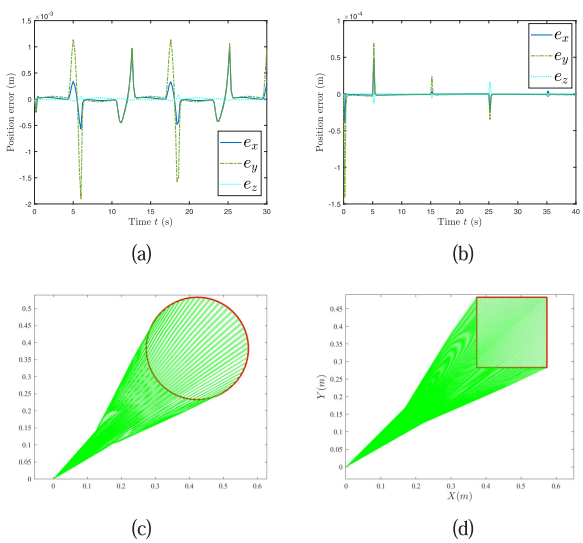

Fig. 9. Tracking performance and the position errors for the manipulator system to track the circular and square paths by the non-sparse autoencoder controller with input saturation. (a) Circle. (b) Square. (c) Circle. (d) Square.

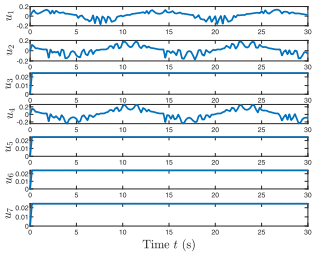

Fig. 10. Control input u without saturation for control of the manipulator system to track the circular path in sparse autoencoder case.

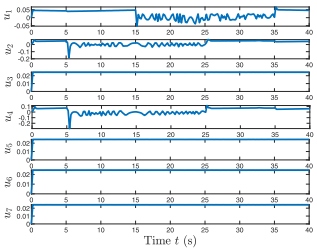

Fig. 11. Control input u without saturation for control of the manipulator system to track the square path in the sparse autoencoder case.

case of non-sparse solution without saturation control input, especially for the square path tracking task. It indicates that the non-sparse autoencoder controller with saturated control input may have more mild transient process. In the non-sparse autoencoder controller with input saturation case, the tracking control performance of the end-effector of the manipulator with position errors is shown by Fig. 9. As observed from Fig. 9, the end-effector of the manipulator can track the circular and square paths well as expected, and the position errors $[e_x, e_y, e_z]$ synthesized by the unsaturated non-sparse autoencoder controller can reach to around 10×10^{-4} m or 1.5×10^{-4} m. All of these results demonstrate the efficiency of the proposed non-sparse autoencoder controller based on the dynamic recurrent neural network for kinematic control of the manipulator system with saturated control input.

3) Sparse Solution Without Saturation Control Input: After validate the non-sparse autoencoder controller for manipulator control, now we investigate the sparse autoencoder controller without input saturation for kinematic control of the manipulator system. In this case, the autoencoder controller without input saturation is obtained in a sparse encode and decode process based on the recurrent neural network model, and the corresponding optimization solver is formulated as an L_1-norm based paradigm rather than an L_2-norm based paradigm. Therefore, the constrained sparse optimization solver (21) is adopted to compute the unsaturated control input u, and the corresponding saturation boundary is configured as Section VI-A. Figs. 10 and 11 show the unsaturated control inputs u solved by the sparse optimization solver for circular path and square path tracking tasks, respectively. Evidently,

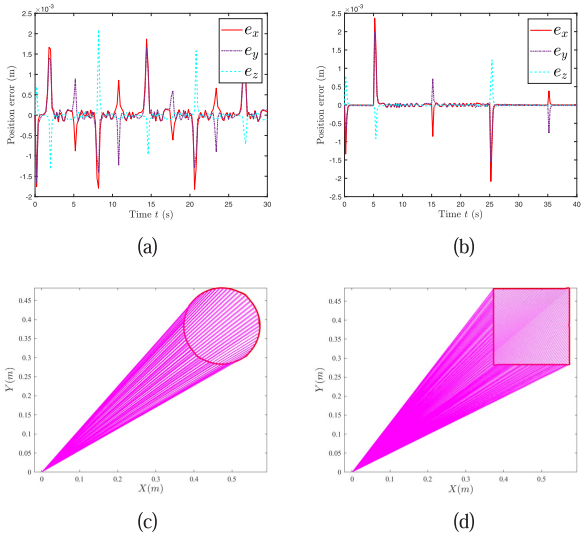

Fig. 12. Tracking performance and the position errors for the manipulator system to track the circular and square paths by the sparse autoencoder controller without input saturation. (a) Circle. (b) Square. (c) Circle. (d) Square.

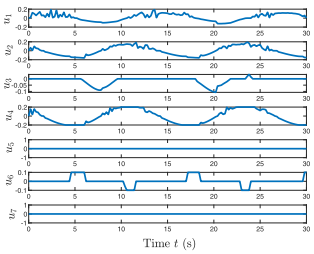

Fig. 13. Control input u with saturation for control of the manipulator system to track the circular path in the sparse autoencoder case.

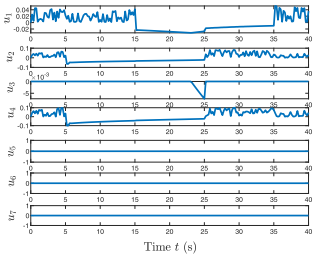

Fig. 14. Control input u with saturation for control of the manipulator system to track the circular path in sparse autoencoder case.

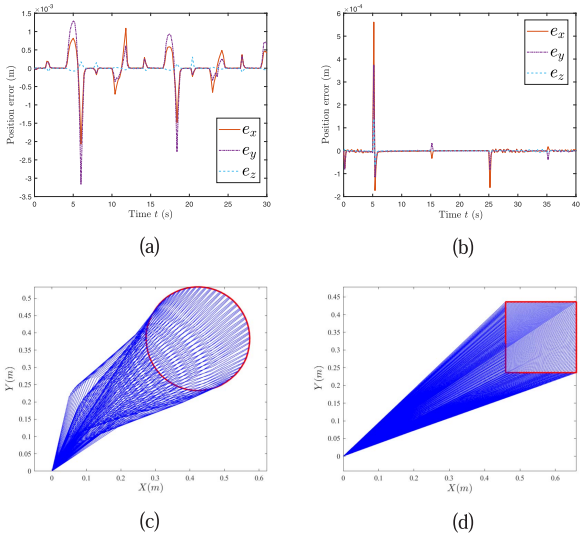

Fig. 15. Tracking performance and the position errors for the manipulator system to track the circular and square paths by the sparse autoencoder controller with input saturation. (a) Circle. (b) Square. (c) Circle. (d) Square.

one can observe that the control input u can be within rather small amplitude range, as compared with the case of non-sparse controller solution without saturation control input which is addressed in Section VI-B. It shows that the sparse autoencoder controller with unsaturated control input might have more mild control input even no saturation bounds are configured in the controller design. For the sparse autoencoder controller without input saturation, additionally, the tracking control performance of the end-effector of the manipulator with position errors is shown by Fig. 12. Observed from Fig. 12, the end-effector of the manipulator can track the circular and square paths promisingly as expected, and the position errors $[e_x, e_y, e_z]$ synthesized by the unsaturated sparse autoencoder controller can reach around 2.5×10^{-3} m. All the tracking control performance results demonstrate the efficiency of the proposed sparse autoencoder controller without input saturation based on the dynamic recurrent neural network for kinematic control of the manipulator system.

4) Sparse Solution With Saturation Control Input: Here, we investigate the sparse autoencoder controller for kinematic control of the manipulator system with input saturation considered. In this case, the autoencoder controller is solved by a sparse encode and decode process based on the recurrent neural network model, and the corresponding constrained sparse optimization solver is formulated in an L_1-norm paradigm with the saturation boundary on the control input u. Therefore, the sparse optimization solver (34) is applied to obtain the saturated control input u, and the corresponding saturation boundary is configured as shown in Table II. Figs. 13 and 14 show the saturated autoencoder control inputs u solved by the sparse optimization solver for circular path and square path tracking, respectively. One can evidently observe that the control input u can be within the saturation boundary provided in Table II. All these results demonstrate that the sparse autoencoder controller with input saturation considered can make control input bounded as expected. In such a sparse autoencoder controller with input saturation, the tracking control performance of the end-effector of the manipulator with position errors is further shown by Fig. 15. Seen from Fig. 15, the end-effector of the manipulator can track the circular and square paths well as expected, and the position errors $[e_x, e_y, e_z]$ synthesized by the saturated sparse autoencoder controller can reach around 3.5×10^{-3} and 6×10^{-4} m. All of these performance results demonstrate the efficiency of the proposed sparse autoencoder controller based on the dynamic recurrent neural network for kinematic control of the manipulator system with saturated control input considered.

Table III summarizes the average position error performances of the four aforementioned autoencoder controllers for the manipulator system. One can see from Table III that the four autoencoder controllers can make the end-effector of the

TABLE III
AVERAGE POSITION ERRORS OF THE AFOREMENTIONED FOUR AUTOENCODER CONTROLLERS FOR THE MANIPULATOR SYSTEM IN NORMAL SITUATION

Autoencoder controller	Circle path (m)	Square path (m)
Non-sparse without saturation	6.04×10^{-3}	6.84×10^{-3}
Non-sparse with saturation	5.21×10^{-3}	4.63×10^{-4}
Sparse without saturation	7.71×10^{-3}	5.31×10^{-3}
Sparse with saturation	7.14×10^{-3}	7.58×10^{-4}

manipulator achieve satisfying accuracy around 10^{-3} m with the parameters configured as aforementioned. These results also demonstrate the efficiency of the proposed autoencoder controllers for manipulation control with input saturation considered, either in non-sparse case or in sparse case.

C. Evaluation of Robustness Against Additive Noise

After verifying the proposed autoencoder controllers for kinematic control of the manipulator system in above normal situations, we evaluate the robustness of the proposed sparse autoencoder controller against additive noises. The autoencoder controller may be disturbed by the noises and have unexpected effects on the manipulator control, thus novel noise suppression should be incorporated to enhance the robustness [40], [44]. The saturation bounds are still set as $[-0.2, 0.2]$, the noise added to the controller is set as the harmonic noise with the form being $A\sin(2\pi ft + \phi)$, where A denotes the noise amplitude, f denotes the noise frequency, and ϕ denotes the noise phase. In the additional simulation setup for the additive noise to the sparse autoencoder controller, the noise frequency is set as $f = 1000$, and we consider three scenarios, that is, case 1: Noise 1 $= \sin(2\pi ft + 2.3)$; case 2: Noise 2 $= 20\sin(2\pi ft + 2.3)$; case 3: Noise 3 $= 200\sin(2\pi ft + 2.3)$. All the amplitudes of these three types of noises exceed the configured saturation boundary. We evaluate the position tracking errors $e = (e_x^2 + e_y^2 + e_z^2)^{1/2}$ of the end-effector of the manipulator system. We compare the position errors e without and with saturation control input for the same kinematic control tasks such as circular path tracking and square path tracking of the end-effector.

1) Circular Path: Fig. 16 shows the position errors of the end-effector of the manipulator system for circular path tracking with the aforementioned three noises of different levels synthesized by the proposed sparse autocoder controller with unsaturated control input (i.e., without saturation boundary). From the first subfigure of Fig. 16, we can observe that, when the first type of noise (i.e., case 1) is added into the controller, the position error can be within a range between 10^{-4} and 10^{-2} m, which is rather small when compared with the radius of the circle (i.e., 0.15 m). When the additive noise's amplitude is enlarged by 20 times (i.e., case 2) as shown in the second subfigure of Fig. 16, the corresponding position error can still be within a range between 10^{-4} and 10^{-2} m synthesized by the proposed sparse autoencoder controller without input saturation, which indicates that the proposed sparse autoencoder controller may have tolerance ability of amplified noise. Moreover, when the additive noise's amplitude is enlarged by 200 times (i.e., case 3) as shown in the third subfigure of Fig. 16, the position error rises between 10^{-3} and 10^{-2} m but it is still within a rather small range, which further seals the robustness of the proposed sparse autoencoder controller with unsaturated control input for kinematic control of manipulator system for the circular path tracking task as well.

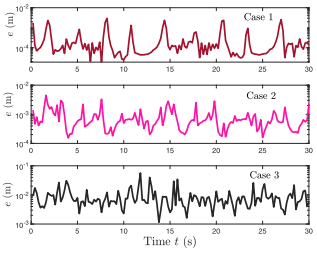

Fig. 16. Position error in sparse case without saturation input with different level of noises for circular path tracking.

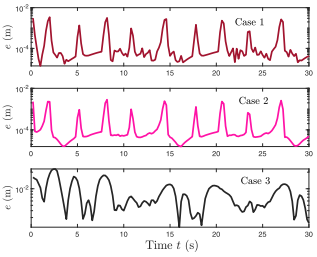

Fig. 17. Position error in sparse case with saturation input with different level of noise for circular path tracking.

Fig. 17 shows the position errors of the end-effector of the manipulator system for circular path tracking with the aforementioned three noises of different levels synthesized by the proposed sparse autocoder controller with saturated control input. From the first subfigure of Fig. 17, we can observe that when the first type of noise (i.e., case 1) is added into the controller, the position error can be within a range between 10^{-4} and 10^{-2} m, which is rather small when compared with the radius of the circle (i.e., 0.15 m). When the additive noise's amplitude is enlarged by 20 times (i.e., case 2) as shown in the second subfigure of Fig. 17, the corresponding position error can still be around a range between 10^{-4} and 10^{-2} m synthesized by the proposed sparse autoencoder controller, which indicates that the proposed sparse autoencoder controller may have tolerance ability of the amplified noise. Moreover, when the additive noise's amplitude is enlarged by 200 times (i.e., case 3) as shown in the third subfigure of Fig. 16, the position error rises around 10^{-2} m but it is still within a rather small range, which further seals the robustness of the proposed sparse autoencoder controller with saturated control input for kinematic control of manipulator system for the circular path tracking task.

2) Square Path: Fig. 18 shows the position errors of the end-effector of the manipulator system for square path tracking with the aforementioned three noises of different levels synthesized by the proposed sparse autocoder controller with unsaturated control input (i.e., without saturation boundary). From the first subfigure of Fig. 18, we can observe that when the first type of noise (i.e., case 1) is added into the controller, the position error can be within a range between 10^{-4} and 10^{-2} m, which is rather small when compared with the radius

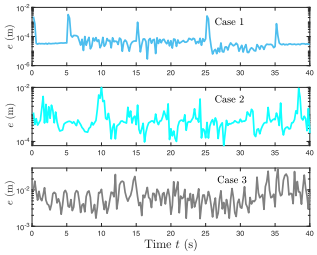

Fig. 18. Position error in sparse case without saturation input with different level of noises for square path tracking.

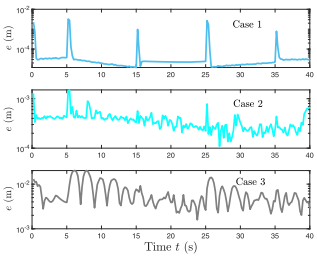

Fig. 19. Position error in sparse case with saturation input with different level of noises for square path tracking.

of the circle (i.e., 0.15 m). When the additive noise's amplitude is enlarged by 20 times (i.e., case 2) as shown in the second subfigure of Fig. 18, the corresponding position error can still be within a range between 10^{-4} and 10^{-2} m synthesized by the proposed sparse autoencoder controller, which indicates that the proposed sparse autoencoder controller may have tolerance ability of the amplified noise. Moreover, when the additive noise's amplitude is enlarged by 200 times (i.e., case 3) as shown in the third subfigure of Fig. 18, the position error rises between 10^{-3} and 10^{-2} m but it is still within a rather small range, which further seals the robustness of the proposed sparse autoencoder controller with unsaturated control input for kinematic control of manipulator system for the square path tracking task.

Fig. 19 shows the position errors of the end-effector of the manipulator system for square path tracking with the aforementioned three noises of different levels synthesized by the proposed sparse autocoder controller with saturated control input. From the first subfigure of Fig. 19, we can observe that when the first type of noise (i.e., case 1) is added into the controller, the position error can be within a range between 10^{-4} and 10^{-2} m, which is rather small when compared with the radius of the circle (i.e., 0.15 m). When the additive noise's amplitude is enlarged by 20 times (i.e., case 2) as shown in the second subfigure of Fig. 19, the corresponding position error can still be around a range between 10^{-4} and 10^{-2} m synthesized by the proposed sparse autoencoder controller, which indicates that the proposed sparse autoencoder controller may have tolerance ability of amplified noise. Moreover, when the additive noise's amplitude is enlarged by 200 times (i.e., case 3) as shown in the third subfigure of Fig. 18, the position error rises around 10^{-2} m but it is still within a rather small range, which further seals the robustness of the proposed sparse autoencoder controller with saturated control input for kinematic control of manipulator system for the square path tracking task.

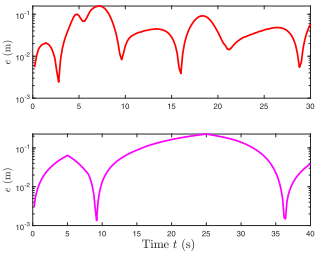

Fig. 20. Position errors in sparse case without input saturation against constant Jacobian uncertainties for both circular and square path tracking. Top: circular path tracking; bottom: square path tracking.

TABLE IV
D–H PARAMETERS OF THE MANIPULATOR CROSS-ROTATIONAL JOINTS

Link	a_i (m)	α_i (rad)	θ_i (rad)
1	l_1	$\pi/2$	θ_1
2	l_2	0	θ_2
3	l_3	$\pi/2$	θ_3
4	l_4	0	θ_4
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	l_5	$\pi/2$	θ_{n-1}
n	l_6	0	θ_n

From these results on the position errors with different additive noises to the sparse autoencoder controller for kinematic control of the manipulator without and with input saturation, one can summarize that even if the additive noise is increased with very large amplitude, the tracking control accuracy can still be guaranteed in an acceptable level. Such performances demonstrate the robustness of the proposed sparse autoencoder controller against additive noises in different amplitude levels.

D. Evaluation of Robustness Against Parameter Uncertainty

After evaluating robustness against additive noise for the proposed sparse autoencoder controller, now we evaluate the effect of the parameter uncertainty to the tracking error accuracy, as parameter uncertainty is quite common when implementing realistic applications [41], [45], [46]. For the kinematic control of the manipulator system, the main parameter uncertainty exists in the Jacobian matrix J in the velocity kinematics equation. We consider the following two cases: 1) constant parameter uncertainty and 2) time-varying parameter uncertainty, and evaluate the position error performance under these two cases.

1) Constant Parameter Uncertainty: First, we evaluate the control performance with constant parameter uncertainty for the sparse autoencoder controller without and with saturation involved, that is, the new Jacobian matrix in the velocity kinematics model becomes $J' = J + \Delta J$, where J is the original Jacobian matrix constructed by the mechanism of the manipulator by D–H parameter table and ΔJ is the randomly generated constant-valued matrix without input from the joint angle variable. For the unsaturated and saturated sparse autoencoder controllers, the tracking targets are set as the circular path and the square path for both controllers. Figs. 20 and 21 show the position error performance under the constant uncertainty in Jacobian matrix, and one can see that for both circular and square paths tracking, the position errors $e = (e_x^2 + e_y^2 + e_z^2)^{1/2}$ can be within a quite small range.

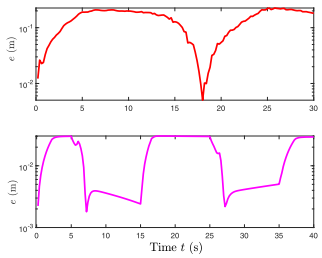

Fig. 21. Position errors in sparse case with input saturation against constant Jacobian uncertainties for both circular and square path tracking. Top: circular path tracking; bottom: square path tracking.

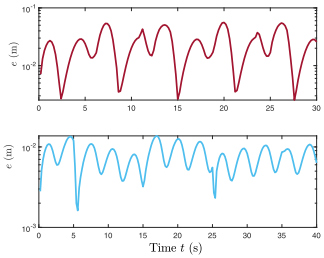

Fig. 22. Position error in sparse case without input saturation against time-varying Jacobian uncertainties for both circular and square path tracking. Top: circular path tracking; bottom: square path tracking.

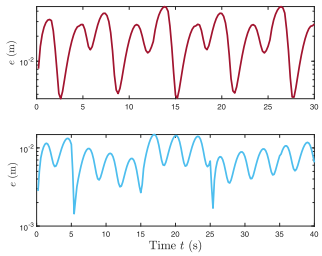

Fig. 23. Position error in sparse case with input saturation against time-varying Jacobian uncertainties for both circular and square path tracking. Top: circular path tracking; bottom: square path tracking.

Specifically, in the unsaturated input case, the position errors can reach 10^{-1} and 10^{-3} m for both tracking tasks. Moreover, in the saturated input case, the position errors can reach 10^{-1} and 10^{-2} m for circular path tracking and 10^{-1} and 10^{-3} for square path tracking. These position errors are small when compared with the shape size of the tracked paths. All of these demonstrate the robustness of the proposed sparse autoencoder controller without and with input saturation against constant parameter uncertainty.

2) Time-Varying Parameter Uncertainty: Second, we evaluate the control performance with time-varying parameter uncertainty for the sparse autoencoder controller without and with input saturation involved, that is, the new Jacobian matrix in the velocity kinematics model becomes $J' = J + \Delta J(t)$ with time-varying parameter disturbance $\Delta J(t)$, where J is the original Jacobian matrix constructed by the mechanism of the manipulator by D–H parameter table and $\Delta J(t)$ is the time-varying/time-dependent matrix and defined as $\Xi(1 + 0.5 \sin\ 2t)$ with parameter Ξ being a randomly generated matrix. For the unsaturated and saturated sparse autoencoder controllers, the tracking targets are set as the circular path and the square path for both controllers as well. Figs. 22 and 23 show the position error performance under the time-varying uncertainty in the Jacobian matrix, and one can observe that,

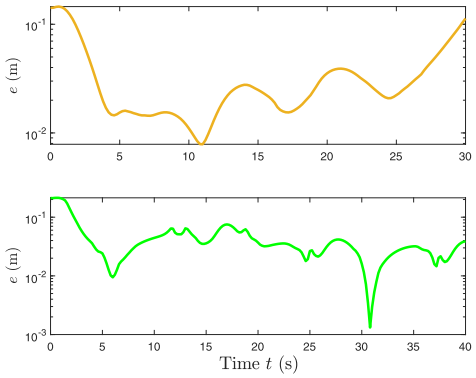

Fig. 24. Position error in sparse case with input saturation against noises and Jacobian uncertainties for both circular and square path tracking in the manipulator with cross-rotational joints. Top: circular path tracking; bottom: square path tracking.

TABLE V

AVERAGE POSITION ERRORS OF THE SPARSE AUTOENCODER CONTROLLER WITHOUT AND WITH INPUT SATURATION FOR CONTROL OF THE MANIPULATOR SYSTEM AGAINST ADDITIVE NOISE AND PARAMETER UNCERTAINTY

Noise/Uncertainty	Without saturation (m)	With saturation (m)
Noise 1	6.12×10^{-3}	6.51×10^{-3}
Noise 2	1.49×10^{-3}	5.48×10^{-3}
Noise 3	8.16×10^{-2}	9.74×10^{-2}
Constant type	7.05×10^{-2}	7.02×10^{-2}
Time-varying type	1.43×10^{-2}	1.03×10^{-2}

for both circular and square path tracking tasks, the position errors $e = (e_x^2 + e_y^2 + e_z^2)^{1/2}$ can be still within a rather small range. In more detail, in the unsaturated input case, the position errors can reach 10^{-1} and 10^{-2} m for circular tracking and 10^{-2} and 10^{-3} m for square tracking. In the saturated input case, the position errors can reach 10^{-2} m for circular path tracking and 10^{-2} and 10^{-3} m for square path tracking. Similarly, these position errors are small when compared with the size of the tracked paths. All of these demonstrate the robustness of the proposed sparse autoencoder controller without and with input saturation against time-varying parameter uncertainty.

3) Validation on More Complex Manipulator With Cross-Rotational Joints: We verify the proposed method on more complex manipulator systems in the noisy and parameter-uncertain scenario, and the manipulator system's D–H parameters are shown in Table IV. The proposed sparse autoencoder controller with input saturation is adopted for the kinematic control tasks. Fig. 24 shows the position error e for the more complex manipulator system with $n = 8$ under aforementioned noises and Jacobian uncertainties, and we could see that the position error of the manipulator can still keep the level around 10^{-1}–10^{-2} m. This validates the efficiency and robustness of the proposed autoencoder controller for manipulator control.

To summarize, Table V further presents the average position errors by the sparse autoencoder controller without and with input saturation for control of the manipulator system against additive noise and parameter uncertainty. Clearly seen from the table, even the additive noise's amplitude is increased largely or the constant and time-varying parameter uncertainty exist along the entire control process, the position errors can be

rather small up to around 10^{-2} m. All these results demonstrate the robustness of the proposed sparse autoencoder controller for manipulator control from different aspects.

VII. Conclusion

In this work, an autoencoder framework based on the recurrent neural network model for kinematic control of manipulators is proposed, and both non-sparse and sparse autoencoder controllers are further developed. Both input unsaturation and saturation cases are considered to be incorporated into the proposed non-sparse and sparse autoencoder controllers. Theoretical analysis and extensive simulations demonstrate that the proposed sparse autoencoder controller with input saturation can make the end-effector of the manipulator system track the desired path efficiently. Additional performance evaluation against the additive noise and parameter uncertainty substantiate robustness of the proposed sparse autoencoder manipulator controller.

References

[1] Y. Shi, B. Qiu, D. Chen, J. Li, and Y. Zhang, "Proposing and validation of a new four-point finite-difference formula with manipulator application," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1323–1333, Apr. 2018.

[2] Z. Zhang, Y. Lin, S. Li, Y. Li, Z. Yu, and Y. Luo, "Tricriteria optimization-coordination motion of dual-redundant-robot manipulators for complex path planning," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1345–1357, Jul. 2018.

[3] Z. Li, C. Li, S. Li, and X. Cao, "A fault-tolerant method for motion planning of industrial redundant manipulator," *IEEE Trans. Ind. Informat.*, vol. 16, no. 12, pp. 7469–7478, Dec. 2020.

[4] R.-J. Wai and Z.-W. Yang, "Adaptive fuzzy neural network control design via a T–S fuzzy model for a robot manipulator including actuator dynamics," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 38, no. 5, pp. 1326–1346, Oct. 2008.

[5] S. Li, J. He, Y. Li, and M. U. Rafique, "Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 415–426, Feb. 2017.

[6] Y. Zhang, S. Chen, S. Li, and Z. Zhang, "Adaptive projection neural network for kinematic control of redundant manipulators with unknown physical parameters," *IEEE Trans. Ind. Electron.*, vol. 65, no. 6, pp. 4909–4920, Jun. 2017.

[7] Y. Zhang, S. Li, J. Zou, and A. H. Khan, "A passivity-based approach for kinematic control of manipulators with constraints," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3029–3038, May 2020.

[8] S. Li, Y. Zhang, and L. Jin, "Kinematic control of redundant manipulators using neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2243–2254, Oct. 2017.

[9] A. A. Maciejewski, "Kinetic limitations on the use of redundancy in robotic manipulators," *IEEE Trans. Robot. Autom.*, vol. 7, no. 2, pp. 205–210, Apr. 1991.

[10] S. Ma, "A new formulation technique for local torque optimization of redundant manipulators," *IEEE Trans. Ind. Electron.*, vol. 43, no. 4, pp. 462–468, Aug. 1996.

[11] Y. Fu and T. Chai, "Neural-network-based nonlinear adaptive dynamical decoupling control," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 921–925, May 2007.

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

[13] S. Li, Z. Shao, and Y. Guan, "A dynamic neural network approach for efficient control of manipulators," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 5, pp. 932–941, May 2019.

[14] W. He, Y. Sun, Z. Yan, C. Yang, Z. Li, and O. Kaynak, "Disturbance observer-based neural network control of cooperative multiple manipulators with input saturation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 5, pp. 1735–1746, May 2020.

[15] Z. Li, Y. Xia, D. Wang, D.-H. Zhai, C.-Y. Su, and X. Zhao, "Neural network-based control of networked trilateral teleoperation with geometrically unknown constraints," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1051–1064, May 2016.

[16] C. Yang, Y. Jiang, W. He, J. Na, Z. Li, and B. Xu, "Adaptive parameter estimation and control design for robot manipulators with finite-time convergence," *IEEE Trans. Ind. Electron.*, vol. 65, no. 10, pp. 8112–8123, Oct. 2018.

[17] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 785–792, Aug. 2011.

[18] Y. Zhang, S. S. Ge, and T. H. Lee, "A unified quadratic-programming-based dynamical system approach to joint torque optimization of physically constrained redundant manipulators," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 5, pp. 2126–2132, Oct. 2004.

[19] D. Chen and Y. Zhang, "A hybrid multi-objective scheme applied to redundant robot manipulators," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 3, pp. 1337–1350, Jul. 2017.

[20] Z. Li, Y. Xia, C.-Y. Su, J. Deng, J. Fu, and W. He, "Missile guidance law based on robust model predictive control using neural-network optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 8, pp. 1803–1809, Aug. 2015.

[21] D. Chen, S. Li, W. Li, and Q. Wu, "A multi-level simultaneous minimization scheme applied to jerk-bounded redundant robot manipulators," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 1, pp. 463–474, Jan. 2020.

[22] Z. Zhang, L.-D. Kong, and L. Zheng, "Power-type varying-parameter RNN for solving TVQP problems: Design, analysis, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2419–2433, Aug. 2019.

[23] W. He, Z. Yan, Y. Sun, Y. Ou, and C. Sun, "Neural-learning-based control for a constrained robotic manipulator with flexible joints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5993–6003, Dec. 2018.

[24] B. Liao, Y. Zhang, and L. Jin, "Taylor $O(h^3)$ discretization of ZNN models for dynamic equality-constrained quadratic programming with application to manipulators," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 225–237, Feb. 2016.

[25] C. Yang, G. Peng, Y. Li, R. Cui, L. Cheng, and Z. Li, "Neural networks enhanced adaptive admittance control of optimized robot–environment interaction," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2568–2579, Jul. 2019.

[26] D. Chen and Y. Zhang, "Robust zeroing neural-dynamics and its time-varying disturbances suppression model applied to mobile robot manipulators," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 4385–4397, Sep. 2018.

[27] Y. Sun, B. Xue, G. G. Yen, and M. Zhang, "A particle swarm optimization-based flexible convolutional autoencoder for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2295–2309, Aug. 2019.

[28] E. Hosseini-Asl, J. M. Zurada, and O. Nasraoui, "Deep learning of part-based representation of data using sparse autoencoders with nonnegativity constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2486–2498, Dec. 2016.

[29] X. Li, R. Zhang, Q. Wang, and H. Zhang, "Autoencoder constrained clustering with adaptive neighbors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 443–449, Jan. 2020.

[30] M. Kachuee, S. Darabi, B. Moatamed, and M. Sarrafzadeh, "Dynamic feature acquisition using denoising autoencoders," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 8, pp. 2252–2262, Aug. 2019.

[31] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, "Convolutional sparse autoencoders for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3289–3294, Jul. 2018.

[32] W. Xu and Y. Tan, "Semisupervised text classification by variational autoencoder," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 1, pp. 295–308, Jan. 2020.

[33] P. Ge, C.-X. Ren, D.-Q. Dai, J. Feng, and S. Yan, "Dual adversarial autoencoders for clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1417–1424, Apr. 2020.

[34] Y. Zhang, D. Guo, and Z. Li, "Common nature of learning between back-propagation and Hopfield-type neural networks for generalized matrix inversion with simplified models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 4, pp. 579–592, Apr. 2013.

[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[36] X. B. Gao, "Exponential stability of globally projected dynamic systems," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 426–431, Mar. 2003.
[37] X. B. Gao and L.-Z. Liao, "A neural network for monotone variational inequalities with linear constraints," *Phys. Lett. A*, vol. 307, nos. 2–3, pp. 118–128, Jan. 2003.
[38] D. Chen, Y. Zhang, and S. Li, "Tracking control of robot manipulators with unknown models: A jacobian-matrix-adaption method," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3044–3053, Jul. 2018.
[39] D. Guo and Y. Zhang, "A new inequality-based obstacle-avoidance mvn scheme and its application to redundant robot manipulators," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 6, pp. 1326–1340, Nov. 2012.
[40] S. Li, M. Zhou, and X. Luo, "Modified primal-dual neural networks for motion control of redundant manipulators with dynamic rejection of harmonic noises," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4791–4801, Oct. 2018.
[41] J. Na, Y. Li, Y. Huang, G. Gao, and Q. Chen, "Output feedback control of uncertain hydraulic servo systems," *IEEE Trans. Ind. Electron.*, vol. 67, no. 1, pp. 490–500, Jan. 2020.
[42] J. Na, B. Wang, G. Li, S. Zhan, and W. He, "Nonlinear constrained optimal control of wave energy converters with adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, vol. 66, no. 10, pp. 7904–7915, Oct. 2019.
[43] G. Lai, Z. Liu, Y. Zhang, C. L. P. Chen, S. Xie, and Y. Liu, "Fuzzy adaptive inverse compensation method to tracking control of uncertain nonlinear systems with generalized actuator dead zone," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 1, pp. 191–204, Feb. 2017.
[44] L. Jin, Y. Zhang, S. Li, and Y. Zhang, "Noise-tolerant ZNN models for solving time-varying zero-finding problems: A control-theoretic approach," *IEEE Trans. Autom. Control*, vol. 62, no. 2, pp. 992–997, Feb. 2017.
[45] Y. Zhang, S. Li, and X. Liu, "Adaptive near-optimal control of uncertain systems with application to underactuated surface vessels," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1204–1218, Jul. 2018.
[46] Q. Liu, M. Liu, Q. Jin, and Y. Liu, "Design of DOB-based control system in the presence of uncertain delays for low-order processes," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 2, pp. 558–565, Mar. 2020.

Zhan Li received the B.S. and M.S. degrees from Sun Yat-sen University, Guangzhou, China, in 2009 and 2011, respectively, and the Ph.D. degree from Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM)-INRIA, University of Montpellier, Montpellier, France, in 2014.

He is currently a Senior Lecturer with the Department of Computer Science, Swansea University, Swansea, U.K. His current research interests include intelligent control of medical/service robotics and biosignal processing.

Dr. Li serves as an Editorial Board Member for *PLOS One* and a Guest Editor for *Frontiers in Neurorobotics* and *Frontiers in Neuroscience*.

Shuai Li (Senior Member, IEEE) received the B.E. degree in precision mechanical engineering from Hefei University of Technology, Hefei, China, in 2005, the M.E. degree in automatic control engineering from the University of Science and Technology of China, Hefei, in 2008, and the Ph.D. degree in electrical and computer engineering from Stevens Institute of Technology, Hoboken, NJ, USA, in 2014.

He is currently an Associate Professor (Reader) at Swansea University, Swansea, U.K., leading the Robotic Laboratory, conducting research on robot manipulation and impedance control, multi-robot coordination, distributed control, intelligent optimization and control, and legged robots.

Dr. Li was the General Co-Chair of 2018 International Conference on Advanced Robotics and Intelligent Control. He is the Founding Editor-in-Chief of *International Journal of Robotics and Control*.