IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 27, 2020, accepted December 3, 2020, date of publication December 9, 2020,
date of current version December 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043190

Analysis and Verification on the Equivalence Between Jerk-Level and Acceleration-Level Schemes Applied to Manipulators Controlling

LI HE[1,2], DAN SU[1,2], MEI LIU[1,2], AND ZHIGUAN HUANG[3]

[1]School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China
[2]Key Laboratory of Medical Imaging, Lanzhou University Second Hospital, Lanzhou 730000, China
[3]Guangdong Provincial Engineering Technology Research Center for Sports Assistive Device, Guangzhou Sport University, Guangzhou 510000, China

Corresponding authors: Mei Liu (mliu@lzu.edu.cn) and Zhiguan Huang (zhiguan1980@163.com)

This work was supported in part by the Gansu Province Key Laboratory of Medical Imaging Fund Project under Grant 18JR2RA028, in part by the Research and Development Foundation of Nanchong, China, under Grant 20YFZJ0018, in part by the Guangzhou Sport University Innovation and Strengthen Project under Grant 5200080589, in part by the Ministry of Education Industry-Academic Cooperation Collaborative Education Program of China under Grant 201901007048, in part by the Fundamental Research Funds for the Central Universities under Grant lzujbky-2019-89, and in part by the Lanzhou Talent Innovation and Entrepreneurship Project of Lanzhou Science and Technology Bureau under Grant 2020-RC-34.

ABSTRACT Two different-level schemes are researched in this article for achieving the kinematic control of redundant manipulators, one of which is exploited at the acceleration level, and the other is at the jerk level. Firstly, they are both reconstructed as a standard quadratic programming problem with different parameter definitions and addressed by a gradient neural network (GNN) method. Secondly, from the perspective of the GNN algorithm, a theoretical interpretation of the intrinsic equivalence between the acceleration-level scheme and jerk-level scheme is performed. Further, simulations on the manipulator synthesized by the two schemes aided with the GNN method tracking two different trajectories are conducted. Finally, comparisons of relevant joint data (i.e., joint angles, joint velocities, and joint accelerations) are presented to substantiate the equivalence between the two schemes, and simulative experiments are carried out at the same time.

INDEX TERMS Jerk-level schemes, acceleration-level schemes, gradient neural network.

I. INTRODUCTION

For a long time, robots have been a concern for many researchers, and their development has led to significant revolutions in many fields [1]–[3], such as the automated production that has already emerged in the industrial field and has broken the limits of pure manual production [4]–[7]. Besides, a particular class of robots called redundant manipulators possesses more degrees-of-freedom than needed to satisfy some extra performance indicators while finishing the primary task [8]–[10]. Redundant manipulators are mainly applied to accomplish tasks that are difficult for human beings or time-consuming, thus reducing unpredictable problems caused by operators in precision and durability [11], [12].

For robots with structure unknown, a data-driven approach is usually adopted to control their motion [13]. For manipulators with the structure known, their kinematics control can be treated as an issue of dealing with inverse kinematic problems [14], [15]. Since the degree of freedom of redundant manipulators is certainly higher than the working space dimension, the kinematic equation is usually underdetermined. That is to say, in many cases, numerous solutions fit the equation. Redundancy analysis is a method to find the optimal solution that meets the performance index from all these solutions [16], [17]. According to the different joint data involved in the performance index, these schemes can be roughly divided into the following four categories: velocity-level schemes [18]–[20], acceleration-level schemes [21]–[28], jerk-level schemes [29], [30], [32], and multi-level schemes [33], [34]. Regarding acceleration-level schemes, there are some representative schemes such as the minimum acceleration norm (MAN) [21], cyclic motion generation (CMG) [22], [23], infinity-norm acceleration minimization scheme [24], and approaches based on their extensions [25]–[27].

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

Although they are all acceleration-based methods, their performance is significantly different. For example, the CMG and CMG-based solutions focus on remedying the joint drift [22], [27], while the MAN scheme focuses more on minimizing the acceleration norm [28]. Beyond that, compared with schemes at the velocity and acceleration levels, there is little research on the jerk-level schemes. They are of critical significance for the robot motion control because the jerk is related to the motion stationarity and structural vibration of a robot [29]–[32]. For instance, in [32], Chen *et al*. investigate a humanized jerk-level scheme for cyclic motion planning and control, which is more humanized than other existing strategies because of its constraint on joint jerks.

In recent years, neural networks, as an extremely effective computational tool, have been widely applied to control [35]–[37], optimization [38]–[40], model prediction [41], [42], and other areas [43]–[46]. Various practical problems, especially for dynamic problems, can be addressed by neural networks [47]–[49]. For example, Cheng *et al*. study a recurrent neural network to solve non-smooth convex optimization problems with the convex inequality and linear equality constraints, and apply it to deal with the genetic regulatory networks [49]. Generally, schemes about manipulators control as stated above are addressed via numerical algorithms [50]–[52] or neural-network-based approaches [53]–[55]. The pseudoinverse based approach is a traditional method for manipulators controlling. Because it requires much time and computational memory to perform pseudoinverse operations, its computing efficiency is not high [51], [52]. Therefore, researchers normally adopt neural-network-based algorithms that possess higher computational performance to control robots [53]–[58]. In [57], a finite-time convergent neural network is studied and used for generating joint data to drive the manipulator to move along the predefined path. To formulate the optimal multi-robot systems, Wang *et al*. research a recurrent neural network to find the optimal formation possessing the minimum distance to the initial multi-robot-system formation, which could also handle ranges of the orientation, scale, and admissible of the formation [58].

Furthermore, it is worth pointing out that although a large number of redundancy analysis schemes have emerged for the motion planning of manipulators in recent years, as mentioned previously, discussions on the relationship between schemes at different levels are still lacking. Moreover, several current discussions on different-level schemes concentrate on the relationship between schemes at the velocity and acceleration levels without consideration of jerk-level schemes [59]–[62]. For instance, Cai *et al*. elucidate the equivalence between the velocity-level scheme and acceleration-level scheme in terms of both theory and simulation [61]. Besides, in [62], an acceleration-level minimum kinetic energy (MKE) scheme is investigated, which is proved to be equivalent to the MKE scheme at the velocity level, and their simulation results are also consistent. Given that there is a rare exploration of the relationship between acceleration-level schemes and jerk-level schemes, in this article, an acceleration-level scheme and a jerk-level scheme are proposed, and the intrinsic equivalence between them is explained and proved through a gradient neural network (GNN).

The rest of the paper is divided into the following sections. In Section II, an acceleration-level and jerk-level scheme are proposed, and both reconstructed as QP problems. Section III exploits a GNN method to address resultant QP problems and interprets the instinct consistency between the two different-level schemes based on the GNN algorithm. Simulations are carried out in Section IV to show their consistency, and the last section concludes the paper. Before ending the current section, the main contributions of the paper are as follows.

1) The article proposes an acceleration-level scheme and a jerk-level scheme for redundant manipulators controlling, and theoretically discusses the instinct equivalence between them. It is worth pointing out that most of the studies focus on the optimization schemes for the motion control of redundant manipulators, and there are few studies on the relationship between schemes based at different levels. For the first time, this article illustrates and proves the equivalence relation between them.
2) The two different-level schemes are rewritten as QP forms through transformation. Then, by derivation, the resultant QP problem is converted into a zero finding issue of linear equations, and addressed by a proposed GNN algorithm.
3) Simulations on the redundant manipulator tracking two different trajectories synthesized by the acceleration-level scheme and jerk-level scheme aided with the GNN algorithm are performed. By comparing relevant joint angles, velocities, and accelerations, their equivalence is substantiated.

II. DIFFERENT-LEVEL SCHEMES

Before exploiting the internal relation between acceleration-level scheme and jerk-level scheme, two different-level schemes are discussed and reconstructed as two standard QP problems.

A. PREPARATORY WORK

Kinematic equations are the basis of analyzing redundant manipulators controlling from the perspective of kinematics. Therefore, it is necessary to describe and expand the redundant manipulator's kinematic equation before introducing the redundancy solution method.

Firstly, the joint angle of the manipulator is defined as $\vartheta = [\vartheta_1, \vartheta_2, \cdots, \vartheta_n]^{\mathrm{T}} \in \mathbb{R}^n$ at time t, where $^{\mathrm{T}}$ is the transpose and n represents the number of joints. Then, the position of the end-effector is assumed as $r_{\mathrm{p}} \in \mathbb{R}^m$, which can be described as a mapping of joint angles $r_{\mathrm{p}} = g(\vartheta)$. Further, extending it to the velocity level leads to

$$\dot{r}_{\mathrm{p}} = J\dot{\vartheta}, \tag{1}$$

where $\dot{r}_{\mathrm{p}} = \partial r_{\mathrm{p}}/\partial t \in \mathbb{R}^m$ is the velocity of the end-effector of the redundant manipulator; $\dot{\vartheta} = \partial\vartheta/\partial t \in \mathbb{R}^n$ is the joint velocity; $J = \partial g(\vartheta)/\partial\vartheta \in \mathbb{R}^{m\times n}$ denotes the Jacobian matrix of $g(\vartheta)$. The time derivative of (1) is written as

$$\ddot{r}_{\rm p} = J\ddot{\vartheta} + \dot{J}\dot{\vartheta}, \tag{2}$$

where $\dot{J} = \partial J/\partial t \in \mathbb{R}^{m\times n}$; $\ddot{r}_{\rm p} = \partial\dot{r}_{\rm p}/\partial t \in \mathbb{R}^m$ is the acceleration of the end-effector; $\ddot{\vartheta} = \partial\dot{\vartheta}/\partial t \in \mathbb{R}^n$ is the joint acceleration. Taking the time derivative of (2) obtains

$$\dddot{r}_{\rm p} = J\dddot{\vartheta} + 2\dot{J}\ddot{\vartheta} + \ddot{J}\dot{\vartheta}, \tag{3}$$

where $\ddot{J} = \partial\dot{J}/\partial t \in \mathbb{R}^{m\times n}$; $\dddot{r}_{\rm p} = \partial\ddot{r}_{\rm p}/\partial t \in \mathbb{R}^m$ is the jerk of the end-effector; $\dddot{\vartheta} = \partial\ddot{\vartheta}/\partial t \in \mathbb{R}^n$ is the joint jerk.

Achieving the kinematic control of manipulators means calculating joint data that can drive a manipulator to move along the predefined trajectory or satisfy the kinematic equations. Theoretically speaking, addressing kinematic equations above can acquire these data, such as joint angles and joint velocities. A direct solving method is a pseudo-inverse based approach which could be described as $\ddot{\vartheta} = J^+(\ddot{r}_{\rm p} - \dot{J}\dot{\vartheta}) + (I - J^+J)s$, or $\dddot{\vartheta} = J^+(\dddot{r}_{\rm p} - 2\dot{J}\ddot{\vartheta} - \ddot{J}\dot{\vartheta}) + (I - J^+J)s$, where J^+ is the pseudoinverse of the Jacobian matrix calculated via $J^+ = J^{\rm T}(JJ^{\rm T})^{-1}$; I stands for the identity matrix; s represents the gradient of a specific index; superscript $^{-1}$ denotes a matrix inverse operator. Although this method is feasible, it is a waste of time because of the large amount of inverse operation.

B. ACCELERATION-LEVEL SCHEME

Firstly, the redundancy analysis problem is studied from the acceleration level, and an acceleration-level scheme is investigated. This scheme optimizes the minimum acceleration norm while completing the trajectory tracking task, and it can be constructed as follows:

$$\begin{aligned} &\min \ \frac{1}{2}\|\ddot{\vartheta}\|_2^2, \\ &\text{s.t. } J\ddot{\vartheta} = \ddot{r}_{\rm d} - \dot{J}\dot{\vartheta} + \beta_1, \\ &\qquad \text{with } \beta_1 = z_1(\dot{r}_{\rm d} - J\dot{\vartheta}) + z_2(r_{\rm d} - r_{\rm p}) \in \mathbb{R}^m, \end{aligned} \tag{4}$$

where β_1 is the feedback to revise position errors and velocities errors; $r_{\rm d} \in \mathbb{R}^m$, $\dot{r}_{\rm d} \in \mathbb{R}^m$, and $\ddot{r}_{\rm d} \in \mathbb{R}^m$ are the desired position, velocity, and acceleration of the end-effector; the equation constraint is transformed from (2); $\|\cdot\|_2$ is the Euclidean norm of a vector; $z_1 > 0$ and $z_2 > 0$ are the feedback coefficients of velocity and position error of the end-effector, separately. Besides, the above optimization scheme can be converted into the following QP problem:

$$\begin{aligned} &\min \ \frac{1}{2}\chi^{\rm T}\chi, \\ &\text{s.t. } A\chi = b, \end{aligned} \tag{5}$$

where $\chi = \ddot{\vartheta} \in \mathbb{R}^n$; $A = J \in \mathbb{R}^{m\times n}$; $b = \ddot{r}_{\rm d} - \dot{J}\dot{\vartheta} + \beta_1 \in \mathbb{R}^m$. Different from the pseudo-invert-based method mentioned in the previous part, describing and solving the control problem of the manipulator from the perspective of optimization can minimize a specific index to meet the requirement of some engineering practices while executing the motion task.

C. JERK-LEVEL SCHEME

Moreover, a jerk-level scheme is exploited and analyzed, of which the objective function and constraint are depicted as follows:

$$\begin{aligned} &\min \ \frac{1}{2}\|\dddot{\vartheta} + \mu\ddot{\vartheta}\|_2^2, \\ &\text{s.t. } J\dddot{\vartheta} = \dddot{r}_{\rm d} - 2\dot{J}\ddot{\vartheta} - \ddot{J}\dot{\vartheta} + \beta_2, \\ &\qquad \text{with } \beta_2 = z_3(\ddot{r}_{\rm d} - \dot{J}\dot{\vartheta} - J\ddot{\vartheta}) + z_4(\dot{r}_{\rm d} - J\dot{\vartheta}) + z_5(r_{\rm d} - r_{\rm p}), \end{aligned} \tag{6}$$

where $\beta_2 \in \mathbb{R}^m$ is the feedback to revise position error, velocities errors, and acceleration errors; $z_3 > 0$ and $z_4 > 0$, and $z_5 > 0$ are the feedback coefficients of acceleration, velocity, and position error of the end-effector, separately; $\mu > 0$ is an auxiliary variable, and $\dddot{r}_{\rm d} \in \mathbb{R}^n$ represents the desired jerk of the end-effector. Besides, extend the objective function to get

$$\frac{1}{2}\|\dddot{\vartheta} + \mu\ddot{\vartheta}\|_2^2 = \frac{1}{2}\dddot{\vartheta}^{\rm T}\dddot{\vartheta} + \mu\ddot{\vartheta}^{\rm T}\dddot{\vartheta} + \frac{1}{2}\mu^2\ddot{\vartheta}^{\rm T}\ddot{\vartheta}.$$

Note that $\mu^2\ddot{\vartheta}^{\rm T}\ddot{\vartheta}$ is a constant from the jerk level, and can be ignored in the objective function because this term has no effects to the result. Therefore, to some extent, the minimum index can be replayed by $\dddot{\vartheta}^{\rm T}\dddot{\vartheta}/2 + \mu\ddot{\vartheta}^{\rm T}\dddot{\vartheta}$ and the scheme is reformulated as the following QP problem:

$$\begin{aligned} &\min \ \frac{1}{2}\chi^{\rm T}\chi + \varrho^{\rm T}\chi, \\ &\text{s.t. } A\chi = b, \end{aligned} \tag{7}$$

where $\chi = \dddot{\vartheta} \in \mathbb{R}^n$; $\varrho = \mu\ddot{\vartheta} \in \mathbb{R}^n$; $A = J \in \mathbb{R}^{m\times n}$; $b = \dddot{r}_{\rm d} - 2\dot{J}\ddot{\vartheta} - \ddot{J}\dot{\vartheta} + \beta_2 \in \mathbb{R}^m$. Compared with the acceleration-level scheme (5), the jerk-level scheme (7) adds the variable $\varrho^{\rm T}\chi$ and defines variables different from (5) such as the vector χ. Generally speaking, the jerk level control may bear some errors due to its supersensitivity. Accordingly, feedback mechanisms, e.g., the position error feedback and velocity error feedback, are designed in the jerk-level control to compensate for these errors and make the results more accurate.

III. SOLUTION AND ANALYSIS

After introducing two different-level schemes mentioned in the previous section, we exploit a neural-network-based algorithm to address the two schemes. Besides, on the basis of the algorithm, an analysis is conducted on the intrinsic consistency of acceleration-level scheme (4) and jerk-level scheme (6).

A. GNN ALGORITHM

To facilitate comparison and analysis, two different-level schemes (4) and (6) need to be solved via an identical algorithm and then are rewritten as a unified form as follows:

$$\begin{aligned} &\min \ \frac{1}{2}\chi^{\rm T}\chi + \delta^{\rm T}\chi, \\ &\text{s.t. } A\chi = b. \end{aligned} \tag{8}$$

It should be noted that although two schemes (4) and (6) are transformed into a standard QP form, specific variables represent different meanings. For instance, for the acceleration-level scheme, variable δ is equal to zero, while it is not zero in the other one.

Based on this uniform form, a Lagrangian function is obtained and defined as $Y(\chi,\rho) = \chi^{\mathrm{T}}\chi/2 + \delta^{\mathrm{T}}\chi + \rho^{\mathrm{T}}(A\chi - b)$, where $\rho \in \mathbb{R}^m$ is the Lagrange multiplier, and the QP problem is converted into a Lagrangian function minimum value problem. Further, it is equivalent to solving the following equations:

$$\begin{aligned}\frac{\partial Y}{\partial \chi} &= \chi + \delta + A^{\mathrm{T}}\rho = \mathbf{0},\\ \frac{\partial Y}{\partial \rho} &= A\chi - b = \mathbf{0},\end{aligned} \tag{9}$$

where $\mathbf{0}$ denotes a vector composed of zeros. Therefore, solutions of two different-level schemes (4) and (6) are transformed into the zero finding problem of linear equations (9) that are redefined as a compact form:

$$Cy = d, \tag{10}$$

where

$$C = \begin{bmatrix} I & A^{\mathrm{T}} \\ A & 0 \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)},$$
$$y = \begin{bmatrix} \chi \\ \rho \end{bmatrix} \in \mathbb{R}^{n+m},\ d = \begin{bmatrix} -\delta \\ b \end{bmatrix} \in \mathbb{R}^{n+m}.$$

That is to say, to obtain the optimal solution of schemes (4) and (6), it is pre-requisite to find the solution of linear system (10). For solving linear system (10), a GNN algorithm is proposed and constructed as follows.

First, an energy cost function is defined as

$$E(y) = \frac{1}{2}\|Cy - d\|_2^2, \tag{11}$$

and when $E(y)$ reaches its minimum, its solution satisfies linear equation (10). Then, the core of the GNN algorithm is to make (11) to reduce in the direction of its negative gradient that is calculated through

$$\frac{\partial E(y)}{\partial y} = \frac{1}{2}\frac{\partial \|Cy - d\|_2^2}{\partial y} = C^{\mathrm{T}}(Cy - d). \tag{12}$$

Based on [63], the GNN algorithm is constructed as follows:

$$\frac{\partial y}{\partial t} = -\eta C^{\mathrm{T}}\Gamma(Cy - d), \tag{13}$$

where $\eta > 0$ is a proportion coefficient to control the convergence rate of GNN algorithm (13), and $\Gamma(\cdot)$ is the activation function. Routinely, activation functions are divided into linear activation functions and nonlinear activation functions. Typical linear activation functions include

$$\Gamma(x_i) = x_i, \tag{14}$$

with x_i representing the ith element of the vector x, and usual nonlinear activation functions contain the power-sigmoid activation function described as

$$\Gamma(x_i) = \begin{cases} x_i^k & x_i \leq -1 \text{ or } x_i \geq 1 \\ \dfrac{(1+\exp(-\varsigma))(1-\exp(-\varsigma x_i))}{(1-\exp(-\varsigma))(1+\exp(-\varsigma x_i))} & -1 < x_i < 1, \end{cases} \tag{15}$$

where $k \geq 3$ and $\varsigma \geq 2$ are two design parameters, and k should be an odd number.

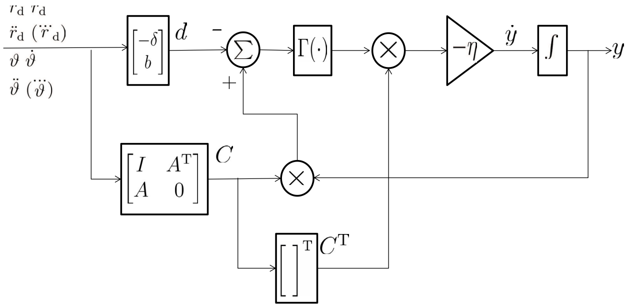

r_{d} $\dot{r}_{\mathrm{d}}$
$\ddot{r}_{\mathrm{d}}$ ($\dddot{r}_{\mathrm{d}}$)
ϑ $\dot{\vartheta}$
$\ddot{\vartheta}$ ($\dddot{\vartheta}$)
$\begin{bmatrix} -\delta \\ b \end{bmatrix}$
d
$-$
$+$
Σ
$\Gamma(\cdot)$
$\times$
$-\eta$
$\dot{y}$
$\int$
y
$\begin{bmatrix} I & A^{\mathrm{T}} \\ A & 0 \end{bmatrix}$
C
$\times$
$[\ ]^{\mathrm{T}}$
C^{T}

FIGURE 1. Architecture of GNN algorithm (13) for addressing acceleration-level scheme (4) and jerk-level scheme (6).

The structure of the GNN algorithm (13) for solving acceleration-level scheme (4) and jerk-level scheme (6) is depicted in Fig. 1. To some extent, such a gradient-based strategy can be deemed as a proportional feedback controller to drive the system to converge to the theoretical solution. Compared with the matrix-inverse-based algorithm, GNN algorithm (13) is more suitable for dealing with the online control of manipulators. For problems (4) and (6), if matrix J is full rank, the two methods usually obtain the same solution. When the matrix is not full of rank, the result via computing the pseudoinverse may be large, i.e., $\ddot{\vartheta}$ could be large, thus leading to a discontinuity phenomenon in joint velocities. However, the GNN (13) can obtain an approximate solution that satisfies the objective function and is continuous, even if the solution space is small or zero. Besides, GNN (13) provides a possibility for solving the problems (4) and (6) that may contain other objective functions and inequality constraints [64], which are difficult for the matrix-inverse-based method to handle.

B. ANALYSIS ON DIFFERENT-LEVEL SCHEMES

The ensuing content aims at explaining the intrinsic consistency of acceleration-level scheme (4) and jerk-level scheme (6) through the proposed GNN algorithm (13).

First, let us review the definition of acceleration-level scheme (4), and it is evident that the optimization objective in (4) is the minimum of $\|\ddot{\vartheta}\|_2^2/2$. Accordingly, it can be defined as the scalar-valued norm-based energy function in the GNN algorithm:

$$E(\ddot{\vartheta}) = \frac{1}{2}\|\ddot{\vartheta}\|_2^2. \tag{16}$$

Referring to the derivation procedure of GNN algorithm (13), as long as energy function (16) changes in the direction of the negative gradient, its minimum value can be acquired.

Then, calculating the gradient of function (16) leads to

$$\frac{\partial E(\ddot{\vartheta})}{\partial \ddot{\vartheta}} = \frac{1}{2}\frac{\partial \|\ddot{\vartheta}\|_2^2}{\partial \ddot{\vartheta}} = \ddot{\vartheta}.$$

To minimize $\|\ddot{\vartheta}\|_2^2/2$, a gradient-based approach is constructed as $\dddot{\vartheta} = -\mu\ddot{\vartheta}$, and is further written as

$$\dddot{\vartheta} + \mu\ddot{\vartheta} = \mathbf{0}, \tag{17}$$

where μ is used to adjust the convergence rate of (17). As long as μ is sufficiently large, equation (17) can be satisfied, and the performance index in (4) is minimized as well. However, it is well known that the requirement of (17) can only be implemented theoretically owing to the trajectory requirement of the manipulator. In general, minimizing $\|\dddot{\vartheta} + \mu\ddot{\vartheta}\|_2^2$ can be chosen as an alternative method to the minimization of $\dddot{\vartheta} + \mu\ddot{\vartheta}$. Hence, the performance index in (4) is physically equivalent to the index $\|\dddot{\vartheta} + \mu\ddot{\vartheta}\|_2^2/2$, and the index at the acceleration level is converted into a jerk-level index. Besides, reviewing jerk-level scheme (6) mentioned in Section II-C, it defines $\|\dddot{\vartheta} + \mu\ddot{\vartheta}\|_2^2/2$ as the minimum index. Then, according to the above analysis, it is concluded readily that theoretical solutions of acceleration-level scheme (4) and jerk-level scheme (6) are equivalent. Besides, the solution of jerk-level scheme (4) is able to converge exponentially to the theoretical solution of acceleration-level scheme (6). The parameter μ controls the exponential convergence speed, and a larger μ means a faster convergence. For example, when $\mu = 60$, $\ddot{\vartheta}$ converges to 5% of the initial value in 0.05 s, and reduces to 0.25% of the initial value in 0.1 s. When time is infinite, the solution of jerk-level scheme (6) becomes the same as the theoretical solution to the acceleration-level one. Nevertheless, time can not be infinite, so there is always an error between the solutions to the two schemes. Since the error converges exponentially and is generally tiny, the two different-level schemes can be considered equivalent, to a certain extent. In practice, the choice between the two schemes depends on the actual situation. If joint jerks drive a manipulator, the jerk-based method is more appropriate than the other one. If joint accelerations drive the manipulator, the acceleration-based solution could be the first choice.

IV. SIMULATION

In view of theoretical explanations, simulations on redundant manipulator tracking different trajectories are performed to substantiate that acceleration-level scheme (4) and jerk-level scheme (6) are indeed equivalent.

First, the redundant manipulator PUMA 560 with $m = 3$ and $n = 6$ is used to verify the equivalence between two schemes (4) and (6) through tracking a circular path with the aid of GNN algorithm (13). In the simulation, the three-dimensional position control of the end effector is only considered, and thus, the PUMA 560 is served as a redundant robot for this particular task. Besides, its schematic diagram, kinematic model, and Jacobian matrix are supplemented in Appendix. Beyond that, relevant settings are as follows: the trajectory is a circle with a radius of 0.15 m; the initial joint angles are defined as $\vartheta(0) = [-\pi/4, 0, 0, \pi/6, \pi/8, -\pi/4]^{\mathrm{T}}$ rad; $\eta = 10^6$; feedback coefficient are defined as $z_1 = z_2 = z_3 = z_4 = z_5 = 2$; $\mu = 60$; the activation function is designed as linear activation function (14) mentioned in Section III-A.

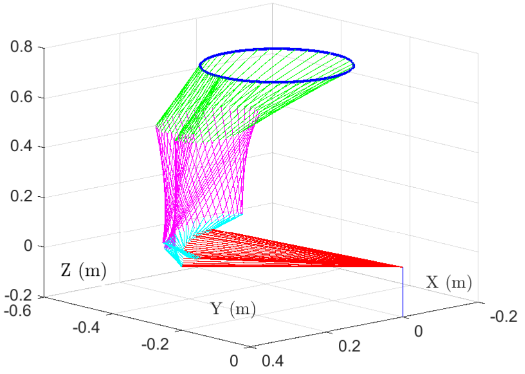


FIGURE 2. Motion trajectories of the manipulator tracking a circular path with the blue line representing the actual end-effector trajectory, and other lines depicting motion process of the individual links.

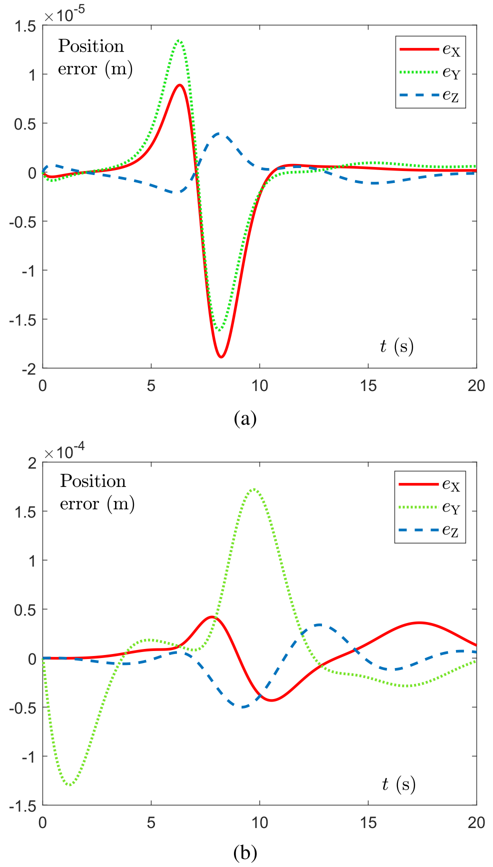
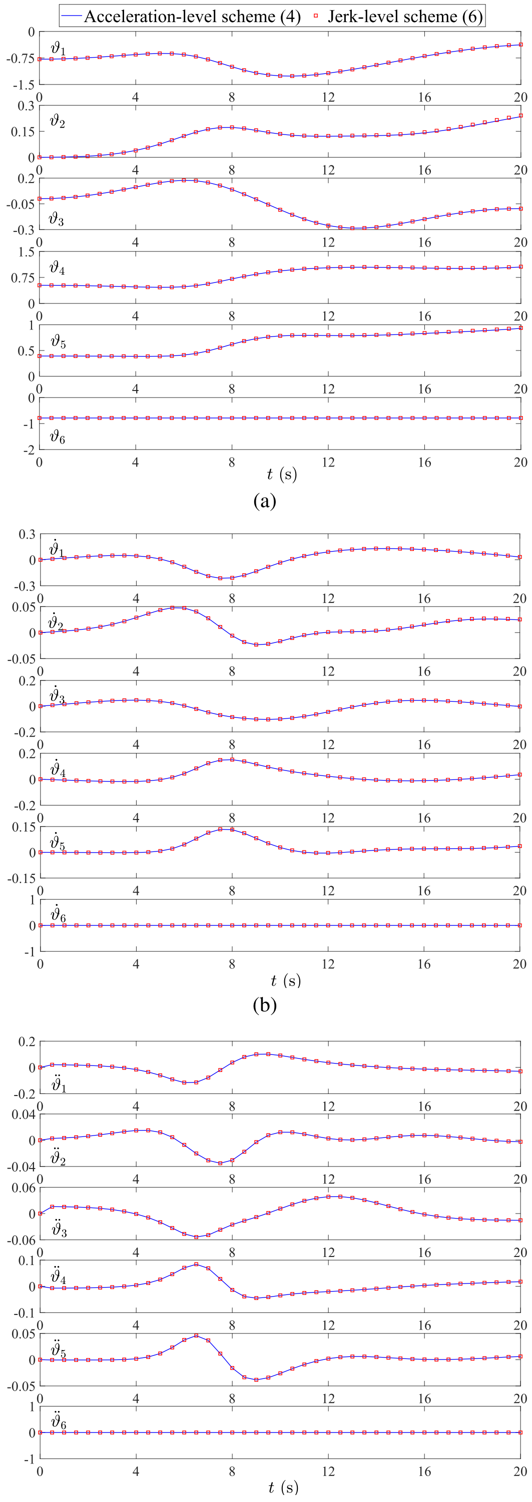


FIGURE 3. Position errors of the manipulator aided with acceleration-level scheme (4) and jerk-level scheme (6) solved by GNN algorithm (13). (a) Acceleration-level scheme (4). (b) Jerk-level scheme (6).



(a)

(b)

(c)

FIGURE 4. **Comparisons on joint angles, velocities, and accelerations of the manipulator aided with acceleration-level scheme (4) and jerk-level scheme (6) solved by GNN algorithm (13) during the task of tracking a circular path. (a) Joint angles. (b) Joint velocities. (c) Joint accelerations.**

Motion trajectories of the manipulator tracking a predefined circular path aided with acceleration-level scheme (4) and jerk-level scheme (6) solved by GNN algorithm (13) are shown in Fig. 2. Since the trajectories of each joint obtained by using the two schemes are similar, only one figure is adopted to describe the changes. Fig. 3(a) and (b) depict the position errors of the end-effector aided with acceleration-level scheme (4) and jerk-level scheme (6) during the task, where e_{X}, e_{Y}, and e_{Z} denote position errors in three dimensions, separately. As visualized in these two figures, the manipulator synthesized by different-level schemes (4) and (6) can both track the predefined path with the maximum error less than 2×10^{-4} m, which suggests the accuracy and effectiveness of the proposed GNN algorithm (13). Besides, the end-effector aided with acceleration-level scheme (4) can accomplish the task in a highly accurate manner.

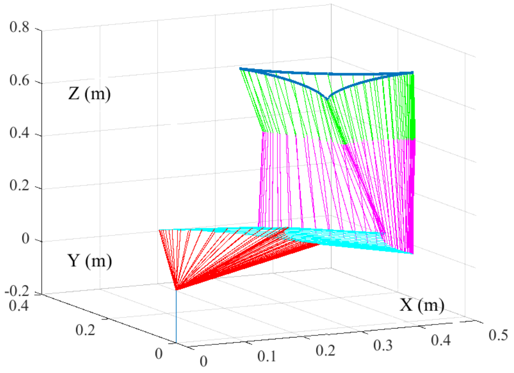


FIGURE 5. **Motion trajectories of the manipulator tracking a tricuspid path with the blue line representing the actual end-effector trajectory, and other lines depicting the motion process of the individual links.**

To elucidate the equivalence between acceleration-level scheme (4) and jerk-level scheme (6), joint angles, velocities, and accelerations generated by schemes (4) and (6) are compared in Fig. 4(a), (b), and (c), respectively. It needs to be supplemented that blue lines represent the transient behaviors of the joint data generated by acceleration-level scheme (4), and the tiny red squares stand for the transient behaviors of the joint data generated by jerk-level scheme (6) in Fig. 4. As observed, these tiny red squares entirely overlap the blue line. In other words, in terms of joint accelerations, velocities, and angles, these results generated via schemes (4) and (6) at different levels are consistent, which substantiates that acceleration-level scheme (4) and jerk-level scheme (6) are indeed equivalent. Furthermore, these comparative results are in accord with the previous theoretical analysis in Section III-B as well.

In addition, a tricuspid path is simulated as well. Other settings are as follows: the initial joint angles are defined as $\vartheta(0) = [\pi/6, 0, 0, \pi/4, 0, 0]^{\mathrm{T}}$ rad; feedback coefficient are defined as $z_1 = z_2 = z_3 = z_4 = z_5 = 2$; $\eta = 10^5$; $\mu = 60$; the activation function is nonlinear activation function (15) appeared in Section III-A. Fig. 5 shows motion trajectories of the redundant manipulator during the task of tracking a tricuspid path. As revealed, the redundant manipulator synthesized by acceleration-level scheme (4) and jerk-level

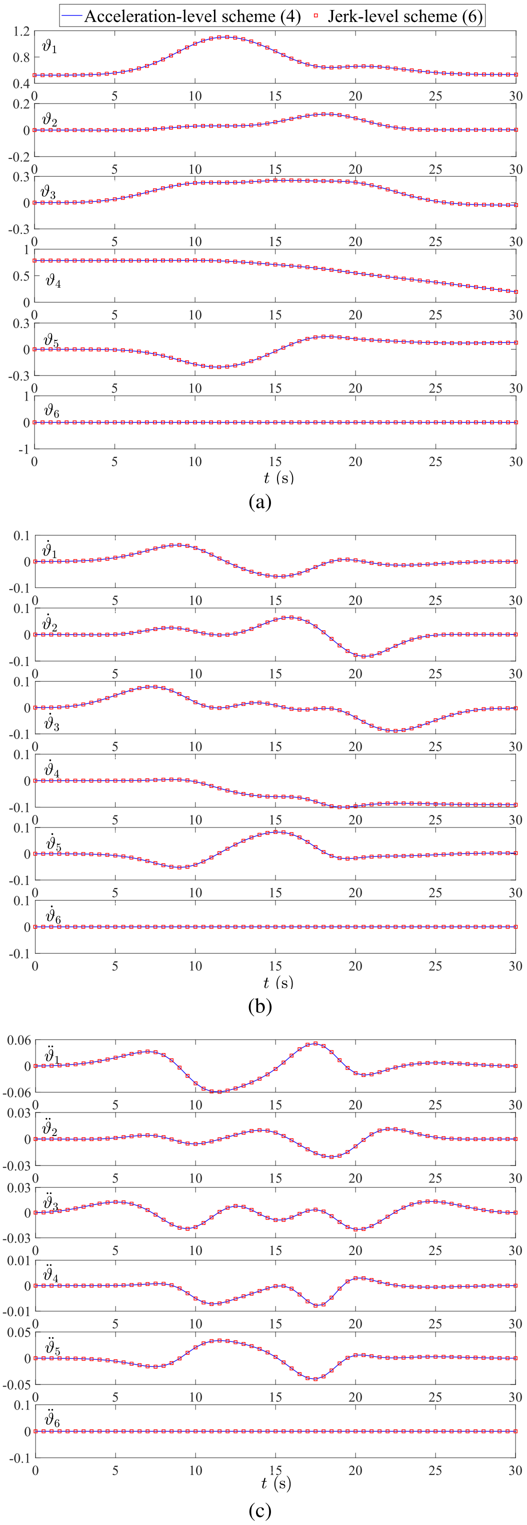


FIGURE 6. Comparisons on joint angles, velocities, and accelerations of the manipulator aided with acceleration-level scheme (4) and jerk-level scheme (6) solved by GNN algorithm (13) during the task of tracing a tricuspid path. (a) Joint angles. (b) Joint velocities. (c) Joint accelerations.

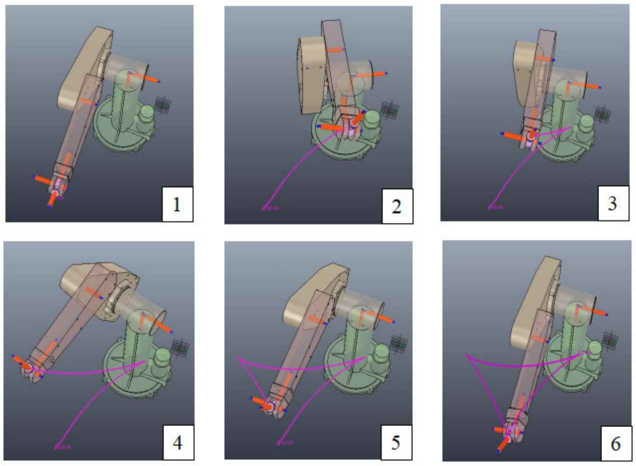


FIGURE 7. Snapshots of PUMA 560 manipulator tracking a tricuspid path synthesized by the GNN model (13) on the CoppeliaSim software.

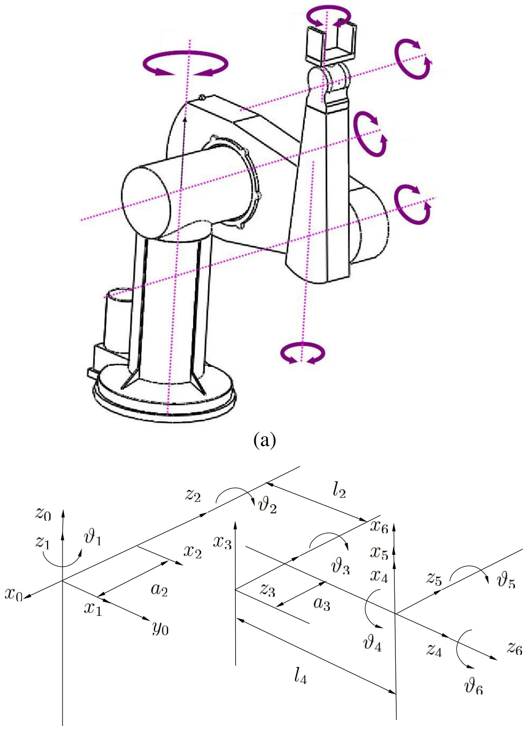
scheme (6) can finish the motion with a tiny error. Besides, Fig. 6 depicts comparisons on joint angles, velocities, and accelerations of the redundant manipulator tracing a tricuspid path aided with acceleration-level scheme (4) and jerk-level scheme (6) solved by GNN algorithm (13). The blue lines and tiny red squares represent the same meanings as indicated in Fig. 4. As observed in Fig. 6, joint angles, velocities, and accelerations generated via schemes (4) and (6) at different levels are consistent, which demonstrates that acceleration-level scheme (4) and jerk-level scheme (6) are equivalent. Beyond that, the simulative experiments on the CoppeliaSim platform is applied to verify the validity of the schemes. Figure 7 displays snapshots of the manipulator PUMA 560 aided with GNN (13) during the motion, in which the task is accomplished.

V. CONCLUSION

This article concentrates on the analysis and verification of the equivalence between different-level schemes applied to redundant manipulators control. Specifically speaking, the acceleration-level scheme and jerk-level scheme have been discussed, and a GNN algorithm has been exploited to address the resultant QP problems converted from them. Then, the intrinsic consistency between the two schemes has been interpreted from a theoretical point of view via the GNN algorithm. To further validate the theoretical analysis, the redundant manipulator has been required to track a circular path and a tricuspid path with the aid of the acceleration-level scheme and jerk-level scheme. The comparisons of joint angles, velocities, and accelerations generated by the two different-level schemes have substantiated the equivalence between the two schemes.

APPENDIX

The appendix aims at supplementing the schematic diagram, kinematic model and Jacobian matrix of the six-link manipulator PUMA 560. Generally, the core of kinematic modelling for a manipulator is constructing the Denavil-Hertenberg (D-H) coordinate which is based on the unity transformation matrix method investigated by Denavil and Hertenberg [65].



(a)

(b)

FIGURE 8. Schematic diagram and D-H coordinate system of PUMA 560. (a) Schematic diagram. (b) D-H coordinate system.

TABLE 1. D-H Parameters of PUMA 560.

Link j	α_{j-1}(rad)	a_{j-1}(m)	l_j(m)	Constraints (rad)
1	0	0	0	$-8\pi/9 \sim 8\pi/9$
2	$-\pi/2$	0	0.15005	$-5\pi/4 \sim \pi/4$
3	0	0.4318	0	$-\pi/4 \sim 5\pi/4$
4	$-\pi/2$	0.0203	0.4318	$-11\pi/18 \sim 17\pi/18$
5	$\pi/2$	0	0	$-5\pi/9 \sim 5\pi/9$
6	$-\pi/2$	0	0	$-8\pi/9 \sim 8\pi/9$

Figure 8(a) and (b) depict the schematic diagram and D-H coordinate system of PUMA 560. In Fig. 8(b), a new coordinate system (including x_j, y_j, and z_j) is created at each joint with the rotation axis as the z-axis. Besides, its linkage parameters are shown in Table 1, in which a_j represents the lengths of jth linkage, α_j denotes twist angle of jth linkage, and l_j is the distance between the two linkages. Facilitated by these linkage parameters and D-H coordinate, the pose transformation matrix between $j-1$th and jth linkages is described as

$$\mathcal{T}_j^{j-1} = \begin{bmatrix} \cos\vartheta_j & -\sin\vartheta_j\cos\alpha_j & \sin\vartheta_j\sin\alpha_j & a_j\cos\vartheta_j \\ \sin\vartheta_j & \cos\vartheta_j\cos\alpha_j & -\cos\vartheta_j\sin\alpha_j & a_j\sin\vartheta_j \\ 0 & \sin\alpha_j & \cos\alpha_j & l_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{18}$$

and the kinematic equation of the PUMA 560 is

$$r_p = g(\vartheta) = \mathcal{T}_1^0\mathcal{T}_2^1\mathcal{T}_3^2\mathcal{T}_4^3\mathcal{T}_5^4\mathcal{T}_6^5. \tag{19}$$

Thus, the Jacobian matrix of PUMA 560 is

$$J = \frac{\partial g(\vartheta)}{\partial \vartheta}. \tag{20}$$

REFERENCES

[1] C. Yang, C. Chen, W. He, R. Cui, and Z. Li, "Robot learning system based on adaptive neural control and dynamic movement primitives," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 3, pp. 777–787, Mar. 2019.

[2] L. Chen, X. Hu, W. Tian, H. Wang, D. Cao, and F.-Y. Wang, "Parallel planning: A new motion planning framework for autonomous driving," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 236–246, Jan. 2019.

[3] 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.

[4] X. Luo, J. Sun, Z. Wang, S. Li, and M. Shang, "Symmetric and nonnegative latent factormodels for undirected, high dimensional and sparse networks in industrial applications," *IEEE Trans. Ind. Informat.*, vol. 13, no. 16, pp. 3098–3107, Dec. 2017.

[5] Y. Huang, J. Na, X. Wu, and G. Gao, "Approximation-free control for vehicle active suspensions with hydraulic actuator," *IEEE Trans. Ind. Electron.*, vol. 65, no. 9, pp. 7258–7267, Sep. 2018.

[6] Q. Li, H. Ju, P. Xiao, F. Chen, and F. Lin, "Optimal trajectory optimization of 7R robot for space maintenance operation," *IEEE Access*, early access, Jul. 20, 2020, doi: 10.1109/ACCESS.2020.3008754.

[7] X. Luo, M. Zhou, S. Li, and M. Shang, "An inherently nonnegative latent factor model for high-dimensional and sparse matrices from industrial applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2011–2022, May 2018.

[8] X. Li, Z. Xu, S. Li, H. Wu, and X. Zhou, "Cooperative kinematic control for multiple redundant manipulators under partially known information using recurrent neural network," *IEEE Access*, vol. 8, pp. 40029–40038, 2020, doi: 10.1109/ACCESS.2020.2974248.

[9] J. Yang, L. Chen, Y. Qi, M. Liu, and C. Cui, "Discrete perturbation-immunity neural network for dynamic constrained redundant robot control," *IEEE Access*, vol. 8, pp. 84490–84500, 2020, doi: 10.1109/ACCESS.2020.2991726.

[10] 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.

[11] J. Zhang, L. Jin, and L. Cheng, "RNN for perturbed manipulability optimization of manipulators based on a distributed scheme: A game-theoretic perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5116–5126, Dec. 2020, doi: 10.1109/TNNLS.2020.2963998.

[12] C. Li, C. Yang, Z. Ju, and A. S. K. Annamalai, "An enhanced teaching interface for a robot using DMP and GMR," *Int. J. Intell. Robot. Appl.*, vol. 2, no. 1, pp. 110–121, Mar. 2018.

[13] J. N. Wong, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, "A data-driven motion prior for continuous-time trajectory estimation on SE(3)," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1429–1436, Apr. 2020.

[14] Z. Li, F. Xu, Q. Feng, J. Cai, and D. Guo, "The application of ZFD formula to kinematic control of redundant robot manipulators with guaranteed motion precision," *IEEE Access*, vol. 6, pp. 64777–64783, 2018.

[15] K. Majd, M. Razeghi-Jahromi, and A. Homaifar, "A stable analytical solution method for car-like robot trajectory tracking and optimization," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 1, pp. 39–47, Jan. 2020.

[16] Z. Xie, L. Jin, X. Du, X. Xiao, H. Li, and S. Li, "On generalized RMP scheme for redundant robot manipulators aided with dynamic neural networks and nonconvex bound constraints," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5172–5181, Sep. 2019.

[17] Z. Xie, L. Jin, X. Luo, Z. Sun, and M. Liu, "RNN for repetitive motion generation of redundant robot manipulators: An orthogonal projection based," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 20, 2020, doi: 10.1109/TNNLS.2020.3028304.

[18] Z. Jia, S. Chen, X. Qu, P. Zhang, and N. Zhong, "Velocity-level tri-criteria optimization scheme for different complex path tracking of redundant manipulators," *IEEE Access*, vol. 7, pp. 64289–64296, 2019.

[19] S. Li, H. Wang, and M. U. Rafique, "A novel recurrent neural network for manipulator control with improved noise tolerance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1908–1918, May 2018.

[20] L. Jin, Y. Zhang, S. Li, and Y. Zhang, "Modified ZNN for time-varying quadratic programming with inherent tolerance to noises and its application to kinematic redundancy resolution of robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 6978–6988, Nov. 2016.

[21] Y. Zhang and Z. Zhang, *Repetitive Motion Planning Control Redundant Robot Manipulators*. New York, NY, USA: Springer-Verlag, 2013.

[22] Z. Xie, L. Jin, X. Luo, S. Li, and X. Xiao, "A data-driven cyclic-motion generation scheme for kinematic control of redundant manipulators," *IEEE Trans. Control Syst. Technol.*, early access, Jan. 15, 2020, doi: 10.1109/TCST.2019.2963017.

[23] Z. Zhang and Y. Zhang, "Acceleration-level cyclic-motion generation of constrained redundant robots tracking different paths," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 42, no. 4, pp. 1257–1269, Aug. 2012.

[24] Y.-N. Zhang, B.-H. Cai, J.-P. Yin, and L. Zhang, "Two/Infinity norm criteria resolution of manipulator redundancy at joint-acceleration level using primal-dual neural network," *Asian J. Control*, vol. 14, no. 4, pp. 1036–1046, Jul. 2012.

[25] D. Guo and Y. Zhang, "Acceleration-level inequality-based MAN scheme for obstacle avoidance of redundant robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 6903–6914, Dec. 2014.

[26] L. Jin, Z. Xie, M. Liu, C. Ke, C. Li, and C. Yang, "Novel Joint-Drift-Free scheme at acceleration level for robotic redundancy resolution with tracking error theoretically eliminated," *IEEE/ASME Trans. Mechatronics*, early access, Jun. 11, 2020, doi: 10.1109/TMECH.2020.3001624.

[27] H. Lu, L. Jin, J. Zhang, Z. Sun, S. Li, and Z. Zhang, "New Joint-Drift-Free scheme aided with projected ZNN for motion generation of redundant robot manipulators perturbed by disturbances," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Dec. 19, 2020, doi: 10.1109/TSMC.2019.2956961.

[28] Z. Jia, X. Qu, S. Chen, N. Zhong, P. Zhang, and F. Ouyang, "Acceleration-level multi-criteria optimization for remedying joint-angle drift of redundant manipulators on complex path tracking," *IEEE Access*, vol. 7, pp. 95716–95724, 2019.

[29] D. Chen and Y. Zhang, "Minimum jerk norm scheme applied to obstacle avoidance of redundant robot arm with jerk bounded and feedback control," *IET Control Theory Appl.*, vol. 10, no. 15, pp. 1896–1903, Oct. 2016.

[30] Y. Zhang, M. Yang, B. Qiu, J. Luo, and H. Tan, "Jerk-level solutions to manipulator inverse kinematics with mathematical equivalence of operations discovered," in *Proc. 12th Int. Conf. Natural Comput., Fuzzy Syst. Knowl. Discovery (ICNC-FSKD)*, Aug. 2016, pp. 2121–2126.

[31] R. R. Nair and L. Behera, "Robust adaptive gain higher order sliding mode observer based control-constrained nonlinear model predictive control for spacecraft formation flying," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 1, pp. 367–381, Jan. 2018.

[32] D. Chen, M. Yang, H. Huang, J. Li, and Y. Zhang, "Computer simulations and comparisons of jerk-level cyclic motion planning and control for CRRM," in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 3668–3673.

[33] 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.

[34] Z.-G. Hou, L. Cheng, and M. Tan, "Multicriteria optimization for coordination of redundant robots using a dual neural network," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 40, no. 4, pp. 1075–1087, Aug. 2010.

[35] Y. Huang, J. Na, X. Wu, X. Liu, and Y. Guo, "Adaptive control of nonlinear uncertain active suspension systems with prescribed performance," *ISA Trans.*, vol. 54, pp. 145–155, Jan. 2015.

[36] L. Cheng, W. Liu, C. Yang, T. Huang, Z.-G. Hou, and M. Tan, "A Neural-Network-Based controller for piezoelectric-actuated Stick–Slip devices," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2598–2607, Mar. 2018.

[37] L. Xiao, S. Li, K. Li, L. Jin, and B. Liao, "Co-design of finite-time convergence and noise suppression: A unified neural model for time varying linear equations with robotic applications," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 12, pp. 5233–5243, Dec. 2020, doi: 10.1109/TSMC.2018.2870489.

[38] N. Liu and S. Qin, "A neurodynamic approach to nonlinear optimization problems with affine equality and convex inequality constraints," *Neural Netw.*, vol. 109, pp. 147–158, Jan. 2019.

[39] S. Qin, X. Yang, X. Xue, and J. Song, "A one-layer recurrent neural network for pseudoconvex optimization problems with equality and inequality constraints," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3063–3074, Oct. 2017.

[40] X. Luo and M. Zhou, "Effects of extended stochastic gradient descent algorithms on improving latent factor-based recommender systems," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 618–624, Apr. 2019.

[41] X. Luo, H. Wu, H. Yuan, and M. Zhou, "Temporal pattern-aware QoS prediction via biased non-negative latent factorization of tensors," *IEEE Trans. Cybern.*, vol. 50, no. 5, pp. 1798–1809, May 2020.

[42] X. Luo, M. Zhou, S. Li, Y. Xia, Z.-H. You, Q. Zhu, and H. Leung, "Incorporation of efficient second-order solvers into latent factor models for accurate prediction of missing QoS data," *IEEE Trans. Cybern.*, vol. 48, no. 4, pp. 1216–1228, Apr. 2018.

[43] P. S. Stanimirovic, I. S. Zivkovic, and Y. Wei, "Recurrent neural network for computing the drazin inverse," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2830–2843, Nov. 2015.

[44] B. Liao, Q. Xiang, and S. Li, "Bounded Z-type neurodynamics with limited-time convergence and noise tolerance for calculating time-dependent Lyapunov equation," *Neurocomputing*, vol. 325, pp. 234–241, Jan. 2019.

[45] X. Luo, M. Zhou, Y. Xia, Q. Zhu, A. Chiheb Ammari, and A. Alabdulwahab, "Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 524–537, Mar. 2016.

[46] P. S. Stanimiroviá and M. D. Petkoviá, "Gradient neural dynamics for solving matrix equations and their applications," *Neurocomputing*, vol. 306, pp. 200–212, Sep. 2018.

[47] N. Zerari, M. Chemachema, and N. Essounbouli, "Neural network based adaptive tracking control for a class of pure feedback nonlinear systems with input saturation," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 278–290, Jan. 2019.

[48] S. Qin and X. Xue, "Dynamical analysis of neural networks of subgradient system," *IEEE Trans. Autom. Control*, vol. 55, no. 10, pp. 2347–2352, Oct. 2010.

[49] L. Cheng, Z.-G. Hou, Y. Lin, M. Tan, W. C. Zhang, and F.-X. Wu, "Recurrent neural network for non-smooth convex optimization problems with application to the identification of genetic regulatory networks," *IEEE Trans. Neural Netw.*, vol. 22, no. 5, pp. 714–726, May 2011.

[50] X. Xiao, L. Wei, D. Fu, J. Yan, and H. Wang, "Noise-suppressing newton algorithm for kinematic control of robots," *IEEE Access*, early access, Aug. 28, 2019, doi: 10.1109/ACCESS.2019.2937686.

[51] H. Zhang, H. Jin, Z. Liu, Y. Liu, Y. Zhu, and J. Zhao, "Real-time kinematic control for redundant manipulators in a time-varying environment: Multiple-dynamic obstacle avoidance and fast tracking of a moving object," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 28–41, Jan. 2020.

[52] B. Liao and W. Liu, "Pseudoinverse-type bi-criteria minimization scheme for redundancy resolution of robot manipulators," *Robotica*, vol. 33, no. 10, pp. 2100–2113, Dec. 2015.

[53] Y. Qi, L. Jin, Y. Wang, L. Xiao, and J. Zhang, "Complex-valued discrete-time neural dynamics for perturbed time-dependent complex quadratic programming with applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3555–3569, Sep. 2020, doi: 10.1109/TNNLS.2019.2944992.

[54] L. Jin, S. Li, and B. Hu, "RNN models for dynamic matrix inversion: A control-theoretical perspective," *IEEE Trans. Ind. Informat.*, vol. 14, no. 1, pp. 189–199, Jan. 2018.

[55] Y. Qi, L. Jin, H. Li, Y. Li, and M. Liu, "Discrete computational neural dynamics models for solving time-dependent Sylvester equations with applications to robotics and MIMO systems," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6231–6241, Oct. 2020, doi: 10.1109/TII.2020.2966544.

[56] L. Jin, J. Yan, X. Du, X. Xiao, and D. Fu, "RNN for solving time-variant generalized Sylvester equation with applications to robots and acoustic source localization," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6359–6369, Oct. 2020, doi: 10.1109/TII.2020.2964817.

[57] L. Xiao, B. Liao, S. Li, Z. Zhang, L. Ding, and L. Jin, "Design and analysis of FTZNN applied to the real-time solution of a nonstationary Lyapunov equation and tracking control of a wheeled mobile manipulator," *IEEE Trans. Ind. Informat.*, vol. 14, no. 1, pp. 98–105, Jan. 2018.

[58] Y. Wang, L. Cheng, Z.-G. Hou, J. Yu, and M. Tan, "Optimal formation of multirobot systems based on a recurrent neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 322–333, Feb. 2016.

[59] Y. Zhang, K. Li, D. Guo, and B. Cai, "Different-level simultaneous resolution of robot redundancy with end-effector path tracked and with joint velocity and acceleration both minimized," in *Proc. 31th Chin. Contr. Conf. (CCC)*, Hefei, China, 2012, pp. 4856–4861.

[60] Y. Zhang, M. Yang, H. Huang, M. Xiao, and H. Hu, "New discrete-solution model for solving future different-level linear inequality and equality with robot manipulator control," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 1975–1984, Apr. 2019.

[61] B. Cai and Y. Zhang, "Different-level redundancy-resolution and its equivalent relationship analysis for robot manipulators using gradient-descent and zhang 's neural-dynamic methods," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3146–3155, Aug. 2012.

[62] D. Guo, K. Zhai, Z. Xiao, H. Tan, and Y. Zhang, "Acceleration-level minimum kinetic energy (MKE) scheme derived via ma equivalence for motion planning of redundant robot manipulators," in *Proc. 7th Int. Symp. Comput. Intell. Des.*, Dec. 2014, pp. 26–30.

[63] Y. Zhang, K. Chen, and H.-Z. Tan, "Performance analysis of gradient neural network exploited for online time-varying matrix inversion," *IEEE Trans. Autom. Control*, vol. 54, no. 8, pp. 1940–1945, Aug. 2009.

[64] L. Jin, J. Zhang, X. Luo, M. Liu, S. Li, L. Xiao, and Z. Yang, "Perturbed manipulability optimization in a distributed network of redundant robots," *IEEE Trans. Ind. Electron.*, early access, Jul. 10, 2020, doi: 10.1109/TIE.2020.3007099.

[65] L. Wu, R. Crawford, and J. Roberts, "An analytic approach to converting POE parameters into D–H parameters for serial-link robots," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2174–2179, Oct. 2017.

LI HE received the B.E. degree from Wuhan University, Wuhan, China, in 2017. She is currently pursuing the M.E. degree in communication and information systems with the School of Information Science and Engineering, Lanzhou University, Lanzhou, China. She currently conducts cooperative research with Key Laboratory Medical Imaging, Lanzhou University Second Hospital, Lanzhou. Her current research interests include robotics, neural networks, intelligent information processing, artificial intelligence, and optimization theory.

DAN SU received the B.E. degree in computer science from Central South University, Changsha, China, in 2014, and the M.E. degree in software engineering from The University of Texas at Austin, Austin, TX, USA, in 2017. She is currently pursuing the Ph.D. degree in artificial intelligence with the School of Information Science and Engineering, Lanzhou University, Lanzhou, China. She currently conducts cooperative research with Key Laboratory Medical Imaging, Lanzhou University Second Hospital, Lanzhou. Her current research interests include neural networks, robotics, and intelligent information processing.

MEI LIU received the B.E. degree in communication engineering from Yantai University, Yantai, China, in 2011, and the M.E. degree in pattern recognition and intelligent system from Sun Yat-sen University, Guangzhou, China, in 2014. She currently conducts research at Lanzhou University, Lanzhou, China, and the University of Chinese Academy of Sciences, Beijing, China. She currently conducts cooperative research with Key Laboratory Medical Imaging, Lanzhou University Second Hospital, Lanzhou. Her current research interests include neural networks, computation, and optimization.

ZHIGUAN HUANG was born in Changning, Hunan, China, in 1980. He received the Ph.D. degree in human kinesiology from South China Normal University, Guangzhou, China, in 2009. He is currently an Associate Professor with the Engineering Research Center for Sport Assistive Device, Guangzhou Sport University. He has published around 30 articles in various journals and conferences. His current research interests include neural networks, sport biomechanics, and sport engineering.