Neural-Network-Based Control of Wheeled Mobile Manipulators with Unknown Kinematic Models

Ning Tan
School of Computer Science and Engineering
Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education
Sun Yat-sen University
Guangzhou, China
tann5@mail.sysu.edu.cn

Zelei Zhu
School of Computer Science and Engineering
Sun Yat-sen University
Guangzhou, China
726672495@qq.com

Peng Yu
School of Computer Science and Engineering
Sun Yat-sen University
Guangzhou, China
yupeng6@mail2.sysu.edu.cn

2020 International Symposium on Autonomous Systems (ISAS) | 978-1-6654-1882-9/20/$31.00 ©2020 IEEE | DOI: 10.1109/ISAS49493.2020.9378850

***Abstract*—As we all known, tracking control is always a basic and important issue in robot industry. In the past, the method based on the pseudo-inverse of Jacobian matrix is a conventional solution which has two major limitations: one is the position error accumulation phenomenon; the other is the requirement for the model information of robots. Therefore, in this paper, we propose a neural-network-based control (NNBC) scheme for wheeled mobile manipulators. The NNBC method overcomes the two limitations and is proved feasible to control the track of stationary robot manipulators. However, in robot industry, mobile manipulators controlling is an unavoidable topic. It is necessary to confirm the NNBC method is workable for mobile manipulators. Thus in this paper, we are going to discuss how to control the track of wheeled mobile manipulators using the NNBC method. In details, we will control the joints of the stationary manipulator and the wheels of the mobile platform to ensure the end-effector make a correct track without knowing the information of the parameters and structure of wheeled mobile manipulators. By integrating the stationary manipulator and the mobile platform into a system, the wheels and the manipulator can cooperate correctly to complete the task of the end-effector. Finally, some simulations will be performed to prove that the NNBC method is workable for wheeled mobile manipulators.**

***Index Terms*—tracking control, neural-network-based control, wheeled mobile manipulators, inverse kinematics**

I. Introduction

It is known that inverse kinematics and forward kinematics are typically two kinds of kinematics of robots [1]–[5]. The forward kinematics problem focuses on how to map the state of robots in the actuation space into the position of the end-effector in the task space. The solution for the problem of forward kinematics is a natural simple one, which is easy to find. However, the problem of inverse kinematics is not a piece of cake due to the singularities and the nonlinearities. However, in robot industry what we usually control are the joints but not directly the end-effector position. Thus, the inverse kinematics is so important for tracking control of industrial robots.

The purpose of tracking control of robot manipulators is to control the end-effector of manipulators to move along the desired path in the task space [6]–[12]. Actually a variety of applications of robot manipulators are realized by tracking control, such as parts assembly [13], painting [14], welding [15], and so on. The problem of model-free tracking control for mobile manipulators is a difficult and key matter in robot search. In the past, all the information of the structure and parameters of mobile manipulators needs to be known to control the track of the end-effector. A conventional solution is termed as Jacobian-matrix-pseudo-inverse (JMPI) method, which needs to know the information of robot model and usually arises the accumulation of position error due to its open-loop feature. As we known, the pseudo-inverse of Jacobian matrix needs to be calculated continuously in the JMPI method and the matrix is depended on the information of the robot parameters and structure, such as the length of links, the specific type of the robot, and the Denavit-Hartenberg (DH) parameters [16], [17]. Although the kinematic model of robots are known, the effects resulting from the long time operation of robot, such as fatigue, friction, wearing, etc., will lead to the changes of actual model parameters, resulting in the difference between the nominal parameter value and the actual parameter value. Moreover, different kinds of robots usually have different kinematics parameters, resulting that the JMPI method is not portable. When it comes to a long-term task, the position error accumulation phenomenon always makes the JMPI method not so workable for a practical application.

For the sake of solving tracking control problem of wheeled mobile manipulators, neural networks have been substantiated to be viable tools due to their computational efficacy and hardware implementation. For instance, Xiao el al. [18] achieved the kinematic control of wheeled mobile robots by exploiting Zhang neural networks which possess exponential

This work was supported by the Key-Area Research and Development Program of Guangzhou (No. 202007030004), the Open Project of Shenzhen Institute of Artificial Intelligence and Robotics for Society (No. AC01202005006), the Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University (No. ICT20040), the Open Research Fund of Engineering Research Center of Software/Hardware Co-design Technology and Application, Ministry of Education (East China Normal University), the Fundamental Research Funds (No. 19lgpy221) for the Central Universities, and the CCF-Tencent Open Fund (No. IAGR20190112).

Corresponding author: Ning Tan

convergence. A novel recurrent neural network with finite-time convergence depicted in [19] was designed to solve a nonstationary Lyapunov equation in real time and applied to the inverse kinematics resolution of a wheeled mobile manipulator. However, both the methods and the JMPI method are model-based methods, which require the information of kinematic model of robots and suffer from the above-mentioned limitations. To overcome the limitations of traditional model-based methods, a Jacobian-matrix-adaption (JMA) scheme is presented in [20] based on zeroing dynamics to solve the tracking control problem of stationary manipulators without the concrete information of the structure and parameters of robots. The zeroing dynamics approach forces all the elements of a predefined error function to converge to zero. It can be regarded as a system and methodology to solve multifarious time-varying problems, which include the tracking control of robots [20]–[22]. By using the zeroing dynamics, the JMA method became the first solution to tackle the unknown kinematics of manipulators. However, the JMA method has only been applied to stationary manipulators, which motivates us to develop a neural-network-based control (NNBC) method for wheeled mobile manipulators with unknown kinematic models.

The remainder of this paper is organized in four sections. In section II, we will introduce the mobile manipulator and formulate the forward kinematics. In section III, the control system will be designed via zeroing neural networks. In section IV, we will perform simulations to verify that the proposed NNBC method is feasible for the control of mobile manipulators. Finally, section V will conclude this paper.

II. Problem Formulation

Compared with stationary manipulators, the difference of controlling wheeled mobile manipulators is that we need to control joints and driving wheels simultaneously. What the system needs to do is to coordinate the wheels and the joints to complete end-effector tasks. The geometric illustration of a mobile manipulator is depicted in Fig. 1 and Table I illustrates the notations used in Fig. 1 [18]. In this paper, only the position of the end-effector is considered without considering the orientation.

It is easy to establish an equation between the joints state and the position of the end-effector in the world coordinate frame for a stationary manipulator. So we could ignore the mobile platform at first, and just establish the following equation in the base coordinate frame (i.e., coordinate frame C in Fig. 1(a)):

$$f(\theta) = r_b \tag{1}$$

where $\theta \in \mathbb{R}^n$ is joint angles and $r_b \in \mathbb{R}^m$ denotes the position of the end-effector. $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ represents a nonlinear mapping function determined by the kinematics of the manipulator.

However, we are now controlling a mobile manipulator, so we have to think about integrated kinematics of the mobile manipulator. To begin with, we need to map the position

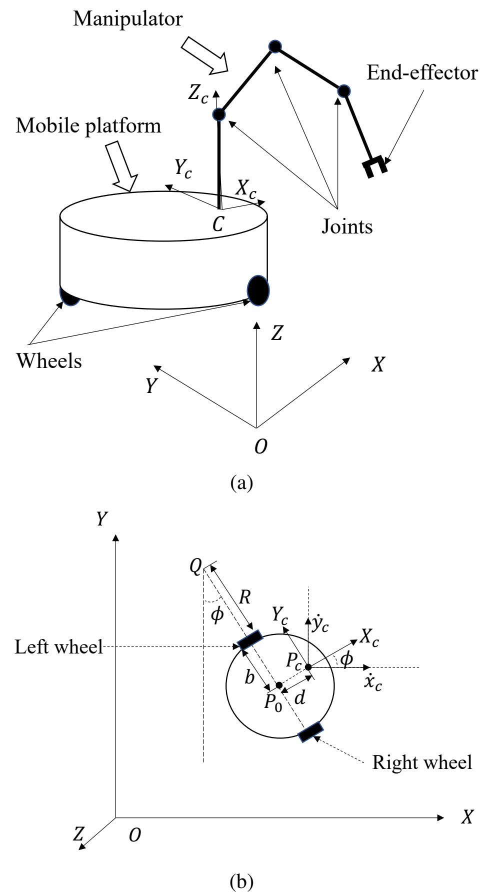


Fig. 1. The geometric illustration of a wheeled mobile manipulator. (a) The schematic diagram of the mobile manipulator. (b) Top view geometry of the platform.

TABLE I
The Descriptions of Symbols Used in This Paper.

Notation	Description
P_0	The point located at the center of the mobile platform.
P_c	The point located at the base of the manipulator with coordinate being $[x_c, y_c, z_c]$.
p_c	$p_c = [x_c, y_c]^T$.
d	The distance between P_0 to point P_c.
b	The distance between P_0 and the two wheels.
r	The radius of the two wheels.
ϕ	The deflection angle of the platform.
$\dot{\varphi}_l, \dot{\varphi}_r$	The angular velocities of the two driving wheels.
C	The coordinate frame located at P_c.
O	The world coordinate frame.

coordinate in the frame C into the position coordinate in the world coordinate system O based on a specific transformation matrix. Then, in the world coordinate frame, we can establish a integrated kinematics equation as follows for the mobile manipulator

$$r_w = \begin{bmatrix} \cos\phi & -\sin\phi & 0 & x_c \\ \sin\phi & \cos\phi & 0 & y_c \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} f(\theta) \\ 1 \end{bmatrix}. \tag{2}$$

The x_c, y_c are the value of the x-coordinate and y-coordinate of p_c. As we said above, we are required to control the wheels, so it is better to write (2) as:

$$r_w = \begin{bmatrix} x_c \\ y_c \\ 0 \end{bmatrix} + g(\theta, \phi) \tag{3}$$

and $g(\cdot) : \mathbb{R}^{n+1} \to \mathbb{R}^m$is a nonlinear mapping determined by the structure and parameters of the mobile manipulator. At the velocity level, the integrated kinematics model of the mobile manipulator can be obtained by differentiating (3) as follows:

$$\dot{r}_w = \begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix} + J(\theta, \phi) \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix}. \tag{4}$$

Note that p_c is not the same as P_c in Table I. Because we know that z_c is always 0, so we use p_c to represent only the $[x_c, y_c]^T$ and the ϕ is not something that our system could control directly. So it is not a good idea to compute the value of it. What we can directly change is the velocity of two wheels. By some simple calculation, the relationship among $\dot{\phi}$, $\dot{\varphi}_l$ and $\dot{\varphi}_r$ can be written as:

$$\dot{\phi} = \frac{r}{2b}(\dot{\varphi}_r - \dot{\varphi}_l). \tag{5}$$

With (5), we can transform (4) into the following form:

$$\dot{r}_w = \begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix} + J(\theta, \varphi_l, \varphi_r) \begin{bmatrix} \dot{\theta} \\ \dot{\varphi}_l \\ \dot{\varphi}_r \end{bmatrix}. \tag{6}$$

Thus, in the world coordinate frame, we could get the position coordinate of the end-effector according to the state of joints and wheels.

III. Control System Design

In this part, a NNBC method based on zeroing neural networks for the control of wheeled mobile manipulators will be developed.

A. Inverse Kinematics

With the aid of zeroing neural networks, to monitor the tracking process, a vector-valued error function needs to be defined to characterize the deviation between the actual trajectory of the end-effector and the desired path at the same time:

$$e = r_d - r_a \tag{7}$$

where r_d denotes the position which user defines or desires and r_a denotes the actual position of end-effector with respect to the world coordinate frame. As described in [22], the design formula of zeroing neural network has the following form

$$\dot{e} = -\gamma e \tag{8}$$

where positive constant γ is a design parameter of the neural network. Considering (6) and substituting the error function (7) into (8) , we have

$$\dot{r}_d - \begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix} - J \begin{bmatrix} \dot{\theta} \\ \dot{\varphi}_l \\ \dot{\varphi}_r \end{bmatrix} = -\gamma(r_d - r_a). \tag{9}$$

Now the control input signal u at the velocity level can be expressed explicitly as:

$$u = \begin{bmatrix} \dot{\theta} \\ \dot{\varphi}_l \\ \dot{\varphi}_r \end{bmatrix} = J^{\dagger}(\dot{r}_d - \begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix} + \gamma(r_d - r_a)). \tag{10}$$

Obviously, the proposed control system (10) is closed-loop since $\begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix}$ and r_a are feedback information during the tracking process which are measured by sensors in the task space and can eliminate the position error accumulation phenomenon. It is worth noting that the above process of solving the inverse kinematics problem is feasible provided that the kinematic model, i.e., the Jacobian matrix J of the mobile manipulator, is known. However, the objective of this work is to control the mobile manipulator without knowing its kinematics, which means J is unknown. To this end, we estimate the unknown Jacobian matrix via another neural network instead of computing the Jacobian matrix based on the kinematic model directly. Thus, control (10) should be rewritten as:

$$u = \begin{bmatrix} \dot{\theta} \\ \dot{\varphi}_l \\ \dot{\varphi}_r \end{bmatrix} = \hat{J}^{\dagger}(\dot{r}_d - \begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix} + \gamma(r_d - r_a)) \tag{11}$$

with $\hat{J}^{\dagger} \in \mathbb{R}^{n\times m}$ being the pseudo-inverse of the unknown Jacobian matrix which needs to be estimated.

B. Jacobian Matrix Update

In this part, we will employ the zeroing neural network again to estimate the unknown Jacobian matrix to achieve the real-time control of the mobile manipulator. In the tracking process, the robot system is assumed to be at a nonsingular configuration.

Firstly, a vector-valued error function is defined as follows inspired by (6)

$$\epsilon = \dot{r}_a - \begin{bmatrix} \dot{p}_c \\ 0 \end{bmatrix} - \hat{J} \begin{bmatrix} \dot{\theta} \\ \dot{\varphi}_l \\ \dot{\varphi}_r \end{bmatrix} \tag{12}$$

where $\epsilon \in \mathbb{R}^m$. The design formula of zeroing neural network is applied as

$$\dot{\epsilon} = -\mu\epsilon \tag{13}$$

with positive constant μ being another design parameter of the neural network. Substituting the error function into the design formula results the following dynamic differential equation

$$\ddot{r}_a - \begin{bmatrix}\ddot{p}_c\\0\end{bmatrix} - \dot{\hat{J}}\begin{bmatrix}\dot{\theta}\\\dot{\varphi}_l\\\dot{\varphi}_r\end{bmatrix} - \hat{J}\begin{bmatrix}\ddot{\theta}\\\ddot{\varphi}_l\\\ddot{\varphi}_r\end{bmatrix} = -\mu(\dot{r}_a - \begin{bmatrix}\dot{p}_c\\0\end{bmatrix} - \hat{J}\begin{bmatrix}\dot{\theta}\\\dot{\varphi}_l\\\dot{\varphi}_r\end{bmatrix}) \tag{14}$$

which is further transformed as

$$\dot{\hat{J}} = (\ddot{r}_a - \begin{bmatrix}\ddot{p}_c\\0\end{bmatrix} - \hat{J}\begin{bmatrix}\ddot{\theta}\\\ddot{\varphi}_l\\\ddot{\varphi}_r\end{bmatrix} + \mu(\dot{r}_a - \begin{bmatrix}\dot{p}_c\\0\end{bmatrix} - \hat{J}\begin{bmatrix}\dot{\theta}\\\dot{\varphi}_l\\\dot{\varphi}_r\end{bmatrix}))\begin{bmatrix}\dot{\theta}\\\dot{\varphi}_l\\\dot{\varphi}_r\end{bmatrix}^{\dagger} \tag{15}$$

where $\ddot{r}_a$ is the acceleration of the end-effector, $\ddot{p}_c$ is the acceleration of the base of the manipulator, $\ddot{\theta}$, $\ddot{\varphi}_l$ and $\ddot{\varphi}_r$ are the accelerations of joints and the two wheels respectively, and the notation $\dagger$ denotes the pseudo-inverse of a vector. Finally, the Jacobian matrix of the mobile manipulator is estimated based on (15).

IV. Simulation

In our simulation, we fix the PUMA 560 manipulator on a mobile platform with two wheels and the resultant mobile manipulator is required to follow a desired path in the three-dimensional task space. In this tracking control task, the mobile manipulator will serve as a functional redundant manipulator. In fact, any functional redundant manipulator is an overactuated robot in the inverse-kinematics resolution, which means the number of actuated joints is more than the number of the output degrees-of-freedom. Although the solution to the inverse-kinematics problem of the redundant manipulator is not unique, the state of the redundant manipulator will only converge to one of the feasible solutions by using the proposed NNBC method. It is worth noting that the proposed NNBC method is not feasible for underactuated robots because of the lack of the redundancy.

In this section, the simulation based on a wheeled mobile PUMA 560 manipulator will be shown for proving the proposed NNBC method is feasible for mobile manipulators. In our simulation, $T_d = 20$ s is selected as the duration of the tracking task. $\theta(0) = [0, \pi/4, 0, 2\pi/3, -\pi/4, 0]^T$ rad is chosen as the initial state of joint angles. Besides, the initial states of the two wheels of the mobile platform are set as $\varphi_l = \varphi_r = 0$ rad. In addition, the initial value of the Jacobian matrix is given as follows to solve the differential equation

$$\hat{J}(0) = \begin{bmatrix} -0.007 & -0.689 & -0.384 & -0.111 & -0.064 & 0 & 0.340 & 0.122 \\ -0.178 & 0 & 0 & -0.091 & -0.157 & 0 & 0.032 & 0.016 \\ 0 & -0.178 & -0.483 & -0.111 & 0.192 & 0 & 0 & 0 \end{bmatrix}$$

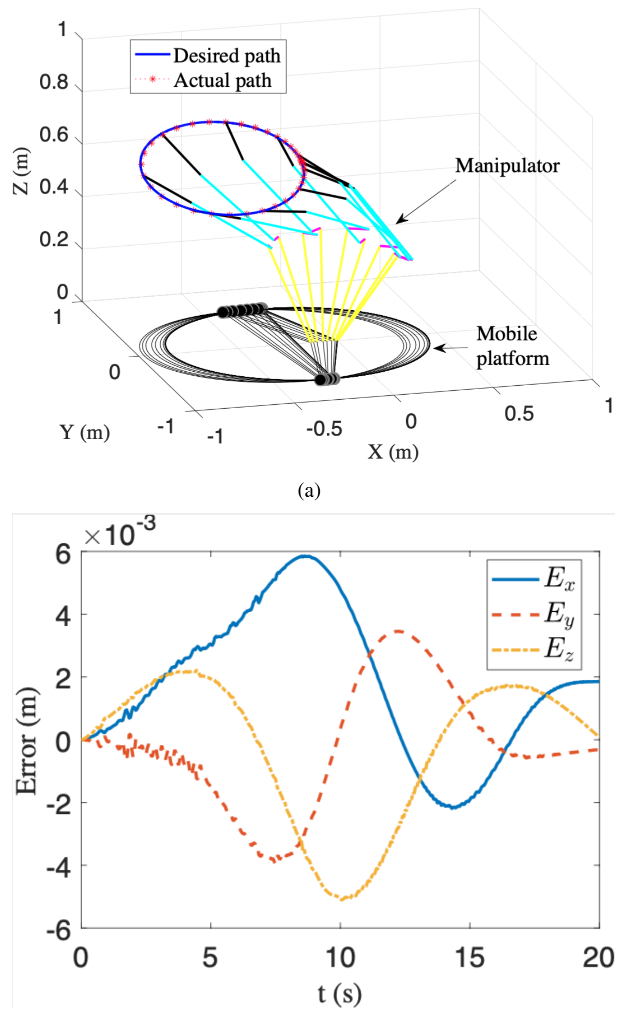
and the design parameters of neural networks are respectively set to be $\gamma = 10$, $\mu = 10$.

The initial value of the Jacobian matrix $\hat{J}(0)$ should be set to be a rough value from engineering experience without loss of generality. When choosing the values of design parameters, the requirement is to satisfy that $\gamma > 0$ and $\mu > 0$. However, the values of design parameters can be set as large as possible to guarantee faster convergence under the premise of hardware computing power permitting.

Now we are going to steer the end-effector to track a circular path in the task space. Firstly, the expression of the desired path r_d corresponding to different axes at the real-time t is set as

$$\begin{aligned} r_{dX} &= \iota\cos(2\pi\sin^2(0.5\pi t/T_d)) - \iota + x_0 \\ r_{dY} &= \iota\cos(\pi/6)\sin(2\pi\sin^2(0.5\pi t/T_d)) + y_0 \\ r_{dZ} &= \iota\sin(\pi/6)\sin(2\pi\sin^2(0.5\pi t/T_d)) + z_0 \end{aligned}$$

where the geometry parameter of the desired path is set as $\iota = 0.4$ m, and $[x_0, y_0, z_0]$ is the absolute position of the desired path in the task space. The simulation results synthesized by the proposed NNBC scheme when tracking a circular path with the mobile manipulator are illustrated in Fig. 2.



(a)

(b)

Fig. 2. Simulation results synthesized by the proposed NNBC scheme when tracking a circular path with the mobile manipulator. (a) The movement of the mobile manipulator. (b) Tracking errors corresponding to different coordinate axes.

The movement of the joints and the mobile platform in the task space can be found in Fig. 2(a) during the tracking task.

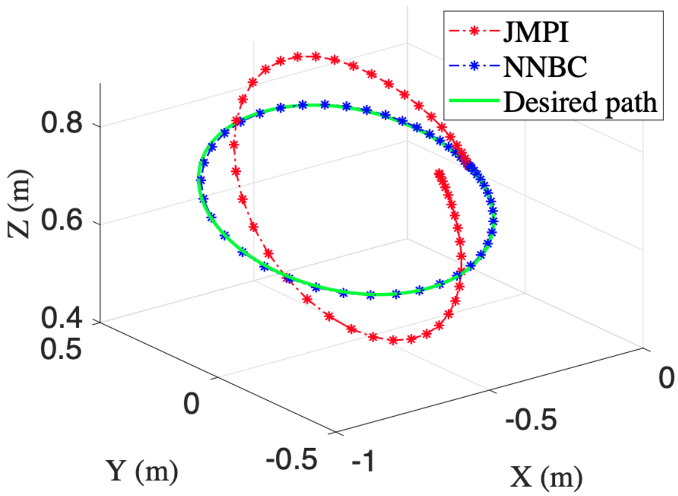


Fig. 3. Comparison between the proposed NNBC scheme and conventional JMPI scheme in the tracking task of mobile manipulators.

The actual trajectory is a precise circle from the figure and it is easy to find that the joints and the two wheels are moving harmoniously during the task execution. The tracking errors $[E_x, E_y, E_z]$ corresponding to different axes are depicted in Fig. 2(b). The scale of the desired path is 0.8 m $\times 0.8$ m $\times 0.2$ m while we could find that the maximum tracking error is not more than 0.006 m, which could be a strong evidence that the NNBC method is feasible for a wheeled mobile manipulator like this. So it could be said that the result is acceptable within the allowable error range. Finally, to verify the merits of the NNBC scheme, we consider the JMPI method as a comparison. As depicted in Fig. 3, we can find that the actual path synthesized by the proposed NNBC method is close enough to the user-defined path. However, the JMPI method apparently fails to fulfill the tracking task, which reveals the efficacy and superiority of the proposed NNBC method.

V. Conclusion

By employing the proposed NNBC scheme based on zeroing neural networks, the tracking control of the wheeled mobile manipulator has been achieved in this paper. The NNBC method makes the problem of tracking control of mobile manipulators become easier, more explicit and external due to the requirement on only the input-output information. Without knowing the parameters and structure of mobile manipulators, the NNBC method has shown the powerful capability for solving the inverse kinematics problem of wheeled mobile manipulators. As for the future work, we should focus more on how to apply the NNBC method to more kinds of robot manipulators. So it will not cost too much to consider about the information of the parameters and structure of the robot manipulators in the robot industry.

References

[1] M. M. Fateh and S. Khorashadizadeh, "Optimal robust voltage control of electrically driven robot manipulators," *Nonlinear Dynamics*, vol. 70, no. 2, pp. 1445–1458, 2012.

[2] C. T. Chen and H. V. Pham, "Trajectory planning in parallel kinematic manipulators using a constrained multi-objective evolutionary algorithm," *Nonlinear Dynamics*, vol. 67, no. 2, pp. 1669–1681, 2012.

[3] L. Xiao and Y. Zhang, "Acceleration-level repetitive motion planning and its experimental verification on a six-link planar robot manipulator," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 906–914, 2013.

[4] N. Rahmanian-Shahri and I. Troch, "Collision-avoidance for redundant robots through control of the self-motion of the manipulator," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 16, no. 2, pp. 123–149, 1996.

[5] H. Ding and S. P. Chan, "Real-time planning algorithm for obstacle avoidance of redundant robots," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 16, no. 3, pp. 229–243, 1996.

[6] Z. Zhang, A. Beck, and N. Magnenat-Thalmann, "Human-like behavior generation based on head-arms model for robot tracking external targets and body parts," *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1390–1400, 2015.

[7] C. Yang, Y. Jiang, Z. Li, W. He, and C. Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.

[8] G. Hu, N. Gans, N. Fitz-Coy, and W. Dixon, "Adaptive homography-based visual servo tracking control via a quaternion formulation," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 1, pp. 128–135, 2010.

[9] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.

[10] K. Tchoń, A. Ratajczak, and I. Góral, "Lagrangian Jacobian inverse for nonholonomic robotic systems," *Nonlinear Dynamics*, vol. 82, no. 4, pp. 1923–1932, 2015.

[11] Y. J. Liu and S. Tong, "Adaptive NN tracking control of uncertain nonlinear discrete-time systems with nonaffine dead-zone input," *IEEE Transactions on Cybernetics*, vol. 45, no. 3, pp. 497–505, 2015.

[12] H. Gao and T. Chen, "Network-based $\mathcal{H}_\infty$ output tracking control," *IEEE Transactions on Automatic Control*, vol. 53, no. 3, pp. 655–667, 2008.

[13] L. Johannsmeier and S. Haddadin, "A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 41–48, 2017.

[14] Y. M. Zhao, Y. Lin, F. Xi, and S. Guo, "Calibration-based iterative learning control for path tracking of industrial robots," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 5, pp. 2921–2929, 2015.

[15] M. Tarokh and X. Zhang, "Real-time motion tracking of robot manipulators using adaptive genetic algorithms," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 74, no. 3-4, pp. 697–708, 2014.

[16] Y. Zhang, J. Wang, and Y. Xu, "A dual neural network for bi-criteria kinematic control of redundant manipulators," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 6, pp. 923–931, 2002.

[17] S. Li, Y. Zhang, and L. Jin, "Kinematic control of redundant manipulators using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2243–2254, 2017.

[18] L. Xiao and Y. Zhang, "Solving time-varying inverse kinematics problem of wheeled mobile manipulators using Zhang neural network with exponential convergence," *Nonlinear Dynamics*, vol. 76, no. 2, pp. 1543–1559, 2014.

[19] 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 Transactions on Industrial Informatics*, vol. 14, no. 1, pp. 98–105, 2018.

[20] D. Chen, Y. Zhang, and S. Li, "Tracking control of robot manipulators with unknown models: a Jacobian-matrix-adaption method," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3044–3053, 2018.

[21] Y. Zhang, L. Xiao, Z. Xiao, and M. Mao, *Zeroing Dynamics, Gradient Dynamics, and Newton Iterations*. Boca Raton, FL, USA: CRC Press, Inc., 2015.

[22] Y. Zhang and D. Guo, *Zhang functions and various models*. Berlin, Heidelberg: Springer, 2015.