IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 3, 2020, accepted September 21, 2020, date of publication September 29, 2020,
date of current version October 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3027512

Comparison on Inverse-Free Method and Psuedoinverse Method for Fault-Tolerant Planning of Redundant Manipulator

JIAWEI LUO[1,3], KENE LI[2], (Member, IEEE), HUI YANG[1], (Member, IEEE), AND JIN YANG[2]
[1]School of Electrical and Automation Engineering, East China Jiaotong University, Nanchang 330013, China
[2]School of Electrical and Information Engineering, Guangxi University of Science and Technology, Liuzhou 545006, China
[3]Jiangxi Vocational College of Industry and Engineering, Pingxiang 337055, China

Corresponding author: Kene Li (likene@163.com)

This work was supported in part by the National Natural Science Foundation (NNSF) of China under Grant 61663003, in part by the Ph.D. Start-Up Fundation of Guangxi University of Science and Technology under Grant 12Z05, and in part by the Science and Technology Research Project in Jiangxi Education Department under Grant GJJ191527.

ABSTRACT Motivated by the importance of fault-tolerant control for robot movement, a fault-tolerant motion planning scheme is proposed to eliminate the joint velocity jump of the redundant manipulator at the time of failure, and enhance the motion stability of the manipulator fault-tolerant operation. The joint velocity jump is eliminated by replacing the degradation scheme at the moment of joint failure, and a neural dynamic method is introduced to eliminate the position error of the end-effector of the manipulator, and the analysis method based on pseudoinverse and inverse-free is used to plan the motion of the manipulator in real time. The simulation results of the redundant manipulator based on the planar four-link indicate that the fault-tolerant scheme can not only realize the manipulator fault-tolerant control without joint velocity jump, but also effectively guarantee the manipulator operation precision.

INDEX TERMS Redundant manipulator, fault-tolerant, joint velocity jump, pseudoinverse, inverse-free.

I. INTRODUCTION

With the development of robot technology, the manipulator is widely used in industrial engineering field with strict requirements for its safety and accuracy [1]–[4]. Due to the complexity of the working environment of the manipulator and the long working time, the manipulator joint failure may occur during the execution of the task, and for the non-redundant robot arm, the joint failure will probably directly lead to the task not being successfully completed. This is not optimistic in practical applications of the manipulator. In order to grant the manipulator more operational flexibility during the movement, so the research of the redundant manipulator has raised more and more attention from experts and scholars. In robotics, redundancy refers to excess degrees of freedom [3]–[5].

An important feature of the redundant manipulator is the fault-tolerant performance that is not available in non-redundant manipulators. When the joint of the manipulator fails, the fault-tolerant solution can be designed to enable the manipulator to continue the desired task [1], [5]–[7]. The redundancy resolution is a crucial issue during fault tolerant operations for redundant manipulator. Because the pseudo-inverse method is simple in structure, easy to understand, and capable of real-time calculation, it is a relatively common method currently used for redundancy analysis [1], [8]–[11]. However, the pseudo-inverse method has to calculate the computationally expensive inverse (specifically, pseudoinverse) of Jacobian matrix, which brings a lot of defects and limitations, and also consumes lots of time. Considering that, to drastically avoid the Jacobian inversion and also to obtain the accurate solution of the time-varying joint velocity jump problem during fault-tolerant control for redundant robot manipulators, it is necessary to introduce other inverse-kinematics schemes through applying the related dynamic methods. Thus based on gradient neural dynamics, another inverse-free scheme is proposed at the joint-velocity level to solve the joint velocity jump problem during the manipulator fault-tolerant control.

When a sudden joint failure occurs in the redundant manipulator, the faulty joint velocity will be abruptly changed to zero at the isolated and locked moment. To maintain the

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng H. Zhu.

desired end-effector motion task, the velocity of the healthy joint will be redistributed. Then the joint velocity of the degraded manipulator may be different from that before the fault, causing the joint velocity jump [12], [13]. The joint velocity jump can cause mechanical and electrical shock to the joint motor and may even damage the manipulator. In addition, The joint velocity will make the redundant manipulator end-effector chatter, which will directly affect the completion of the subsequent tasks and the control accuracy of the manipulator [13]. That is, the joint velocity jump will cause the end-effector to deviate from the desired trajectory, and may even lead to the task failure. Therefore, in order to ensure that the manipulator can steadily and accurately complete the task, it is of great significance to carry out research on the joint velocity jump during fault-tolerant control for manipulator.

The robotic scholars have done a lot of research work on this issue. J. Zhao gave out a definition to the joint velocity jump in the process of the redundant robot fault-tolerant operation, and proposed some fault-tolerant algorithms that can achieve the minimum joint velocity jump [7], [13]–[15]. Hamid Abdi used matrix perturbation to model the fault joints of a mechanical arm, minimizing sudden change in the end-effector velocity [16]. Q. Jia converted the problem of sudden velocity change caused by joint failure into the optimization problem of finding the best compensation vector of joint velocity [17]. However, these methods can only reduce the joint velocity jump, but can not completely eliminate it. This article presents two fault-tolerant planning schemes without joint velocity jump based on pseudoinverse-type and inverse-free type methods, and gives the compare-son on the two methods.

Based on the above analysis, this article proposes a fault-tolerant scheme with no joint velocity jump for the problem of joint failure during the movement of the robot arm. At the moment of manipulator joint failure, the robotic arm is controlled to continue to perform the desired task by converting the original scheme into a degradation scheme. At the same time, the initial joint velocity of the degradation scheme is defined as the joint velocity of the original scheme failure moment, and the redundancy analysis method based on pseudo-inverse and no inverse is used to plan the motion of the manipulator in real time. The position error existing in the redundant end-effector is eliminated by using an error-eliminating method based on a neuro-dynamic method [18], [19]. The corresponding mathematical description of the fault-tolerant scheme is followed; and the effectiveness and accuracy of the manipulator fault-tolerant scheme is proved by simulation experiments.

II. FAULT-TOLERANT CONTROL SCHEME BASED ON PSEUDO-INVERSE METHOD

The relation between the redundant manipulator end-effector position and orientation vector $r(t) \in \mathrm{R}^m$ in Cartesian space and the joint space vector $\theta(t) \in \mathrm{R}^n (\mathrm{n} > \mathrm{m})$ can be expressed as the following forward kinematic equation:

$$r(t) = f(\theta(t)) \tag{1}$$

where $f(\cdot)$ is a differentiable nonlinear mapping with a known structure and parameters for a given manipulator. By differentiating (1) with respect to time t, The inverse-kinematic problem is thus usually solved at the joint-velocity level:

$$\dot{\theta}(t) = J^{+}(\theta)\,\dot{r}(t) \tag{2}$$

where $J^{+} = J^{T}\left(JJ^{T}\right)^{-1} \in \mathrm{R}^{m\times n}$ is the pseudoinverse of Jacobian matrix, $\dot{\theta}(t)$ is the joint velocity vector, $\dot{r}(t)$ is the Cartesian velocity vector.

At any time during the task execution, once the robot manipulator is isolated and locked due to the joint failure, it will degenerate. In order to achieve the execution of the desired end-effector task, and to take into account the joint velocity jump and the end-effector position error during fault-tolerant operation, a degradation scheme can be considered to replace the original scheme (2) with the faulty joint being locked. The movement of the manipulator is to achieve the purpose of joint fault tolerance. That is to say, the joint velocity at the time of failure is used as the initial joint velocity of the degraded scheme, and the joint velocity at the time of joint fault tolerance does not jump. According to the length of the link of the original manipulator and the joint angle at the moment of failure, the length of the link of the degraded manipulator and the initial joint angle can be solved by the triangular cosine rule. Then, the degradation scheme achieves the fault tolerance of the joint without joint velocity jump and also ensures the accuracy of the task execution. Furthermore, it can be solved by the following expression:

$$\dot{\theta}^{*}(t) = \dot{\theta}(t_s)\,(1-\delta(t)) + \delta(t)\,\dot{\theta}(t) \tag{3}$$

where, $\dot{\theta}(t_s)$ is the joint velocity of the original manipulator (except for the locked joint) at the time of failure. $\dot{\theta}(t) = J^{+}(\theta)\,\dot{r}(t)$ is the joint velocity of the degraded manipulator; and $\delta(t) = 2/(1+\mathrm{e}^{-(t-t_s)}) - 1$ is a real function for smoothing the joint velocity of the manipulator.

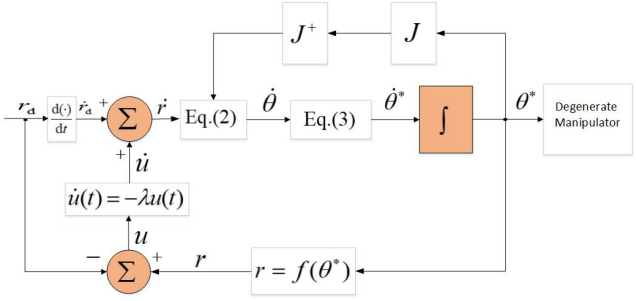


FIGURE 1. Degradation control scheme based on pseudo-inverse.

For the sake of understanding, Fig. 1 shows the control block diagram of the degradation scheme based on pseudo-inverse. It is worth noting that after replacing the original scheme with the degradation scheme, there will be an error

of the end-effector of the robot arm between the actual position and the desired position. In order to improve the task execution accuracy of the redundant robot arm, the real-time deviation is defined as:

$$u(t) = r(t) - r_d(t) \tag{4}$$

where $r_d(t)$ is the desired path of the manipulator's end-effector. In order to eliminate the error of the end-effector, Eq. (4) must be convergent to 0, and thus, we adopt the derivative of $u(t)$ as follows [18], [19]:

$$\dot{u}(t) = \frac{\mathrm{d}u(t)}{\mathrm{d}t} = -\lambda u(t) \tag{5}$$

where, $\dot{u}(t)$ represents the derivative of the vector value deviation between the actual and desired end-arm actuator trajectories over time. $\lambda > 0$ is a parameter that can be set arbitrarily to adjust the error convergence rate. It is worth noting that the redundancy robot can be proved to converge to zero over time during the execution of the task.

Proof: Rearranging (5):

$$\frac{1}{u(t)}\mathrm{d}u(t) = -\lambda \mathrm{d}t \tag{6}$$

Simultaneous integration of both sides of (6):

$$u(t) = C\exp(-\lambda t) \tag{7}$$

Assuming that at some point t_s in the task duration, the end-effector position deviates from the desired trajectory, the resulting position error is:

$$u(t_s) = C\exp(-\lambda t_s) \tag{8}$$

We can obtain $C = u(t_s)\exp(\lambda t_s)$, and substituting it into (7) gives:

$$u(t) = u(t_s)\exp(-\lambda(t - t_s)) \tag{9}$$

Then, we can get the derivative of (9):

$$\dot{u}(t) = -\lambda u(t_s)\exp(-\lambda(t - t_s)) \tag{10}$$

Seen from equation (10), when $t \to \infty$, the derivative $\dot{u}(t)$ converges to zero with time. The proof is completed.

Furthermore, the task of the end-effector can be designed as

$$\dot{r}(t) = \dot{r}_d(t) + \dot{u}(t) \tag{11}$$

where, $\dot{r}(t)$ represents the optimized velocity of the end-effector, $\dot{r}_d(t) = \mathrm{d}r_d(t)/\mathrm{d}t$ represents the desired Cartesian velocity of the end-effector. Then, by using the error elimination methods (4)-(11) in the original schemes (1)-(2) and the degradation schemes (2)-(3), the task execution acc-uracy of the robot end-effector can be improved.

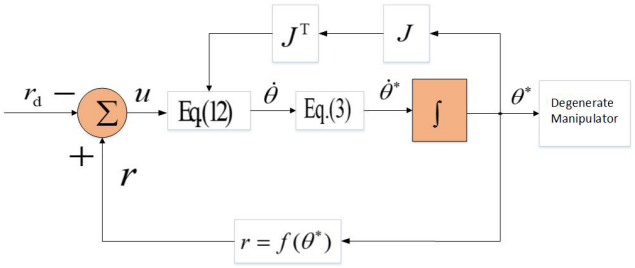


FIGURE 2. Block diagram based on the inverse-free degradation scheme.

III. FAULT-TOLERANT BASED ON INVERSE-FREE METHOD

Considering the degradation schemes (2)-(3) based on the pseudoinverse (including the error elimination method, the same below) involves the inversion of the matrix, and the mechanical Cartesian velocity needs to be redesigned again to eliminate the position error of the end-effector because of the joint failure. Therefore, the computation is more complicated. In this section, the fault-tolerant control of the manipulator is realized by designing the degradation scheme based on the inverse-free method instead of the original scheme (1)-(2) (including the error elimination method, the same below), which is different from the pseudoinverse resolving method.

Firstly, based on gradient dynamics, we define a scalar-valued norm-based energy function:

$$E = \|f(\theta(t)) - r_d(t)\|_2^2/2$$

Secondly, an inverse-free scheme can be designed to evolve along the gradient of $E1(\dot{\theta}, \theta, t)$, i.e., $\partial\varepsilon/\partial\dot{\theta}$, until the minimum point is reached. In view of $\partial J(\dot{\theta})/\partial\dot{\theta} = 0$, the joint velocity of degenerate manipulator is obtained by:

$$\dot{\theta}(t) = -\eta J^{\mathrm{T}} u(t) \tag{12}$$

where η is generally a positive-definite parameter that is used to scale the convergence rate of the scheme. $J^{\mathrm{T}} \in \mathrm{R}^{n\times m}$ denotes the transposition of the Jacobian matrix. $r(t) \in \mathrm{R}^m$ is the Cartesian position vector of the end-effector. To keep the differential equation (12) well conditioned, it is necessary to keep η well conditioned; e.g., the eigenvalues of η are in the same scale.

Then, the inverse-free degradation scheme can be solved by equation (3). In particular, the time-varying function $\delta(t)$ here is designed as $\delta(t) := 1/(1+\mathrm{e}^{(-\beta(t-t_s-((T-t_s)/2)))})$, where the parameter $\beta > 0$ is set arbitrarily and T is the task dur-ation. The block diagram based on the inverse-free degradation scheme is shown in Figure 2. It is worth noting that there is no complicated pseudoinverse operation, and the joint velocity of the degraded manipulator is also calcu-lated, as well as position error of the end-effector is eliminated at the same time.

IV. NUMERCAL SIMULATION

In this section, we use the planar four-link redundant manipulator as the simulation model, then we conduct the simulations using the degradation scheme based on pseudo-inverse

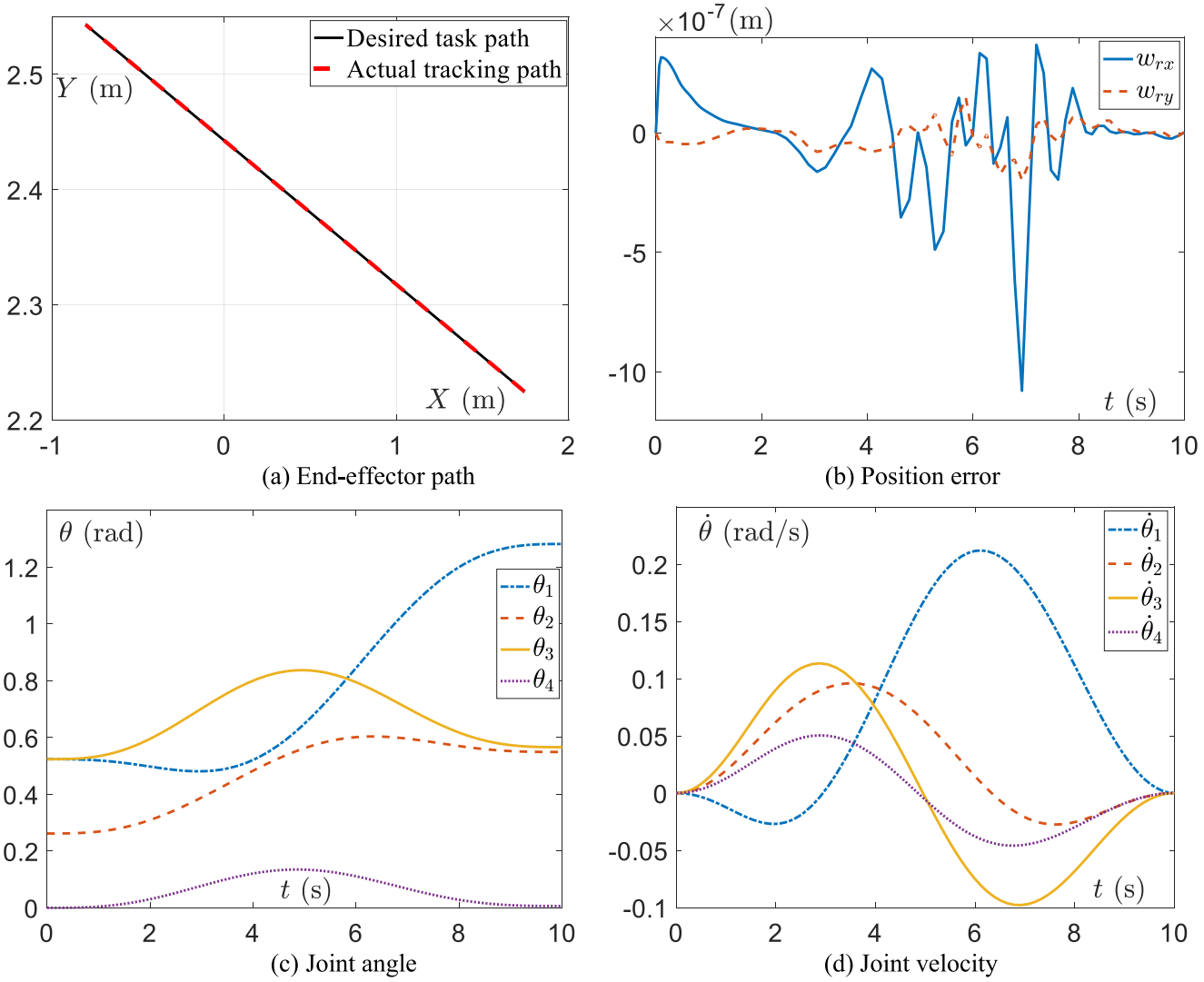


FIGURE 3. Simulation results of original scheme straight line trajectory tracking without joint failure.

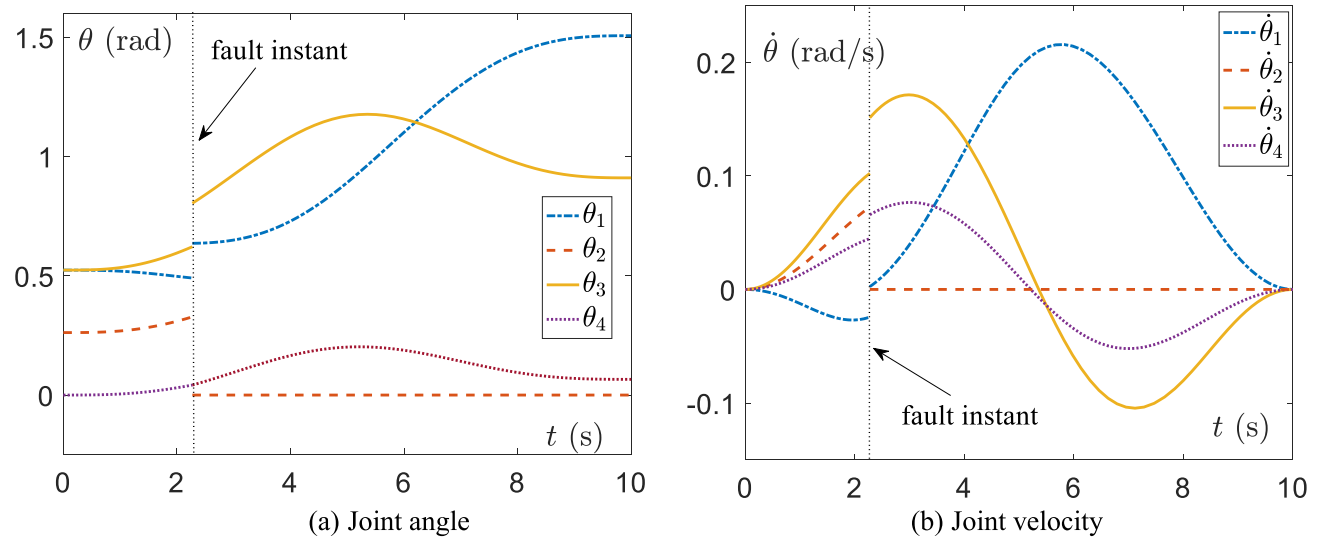


FIGURE 4. Simulation results of velocity jump elimination when the 2nd joint failure occurs.

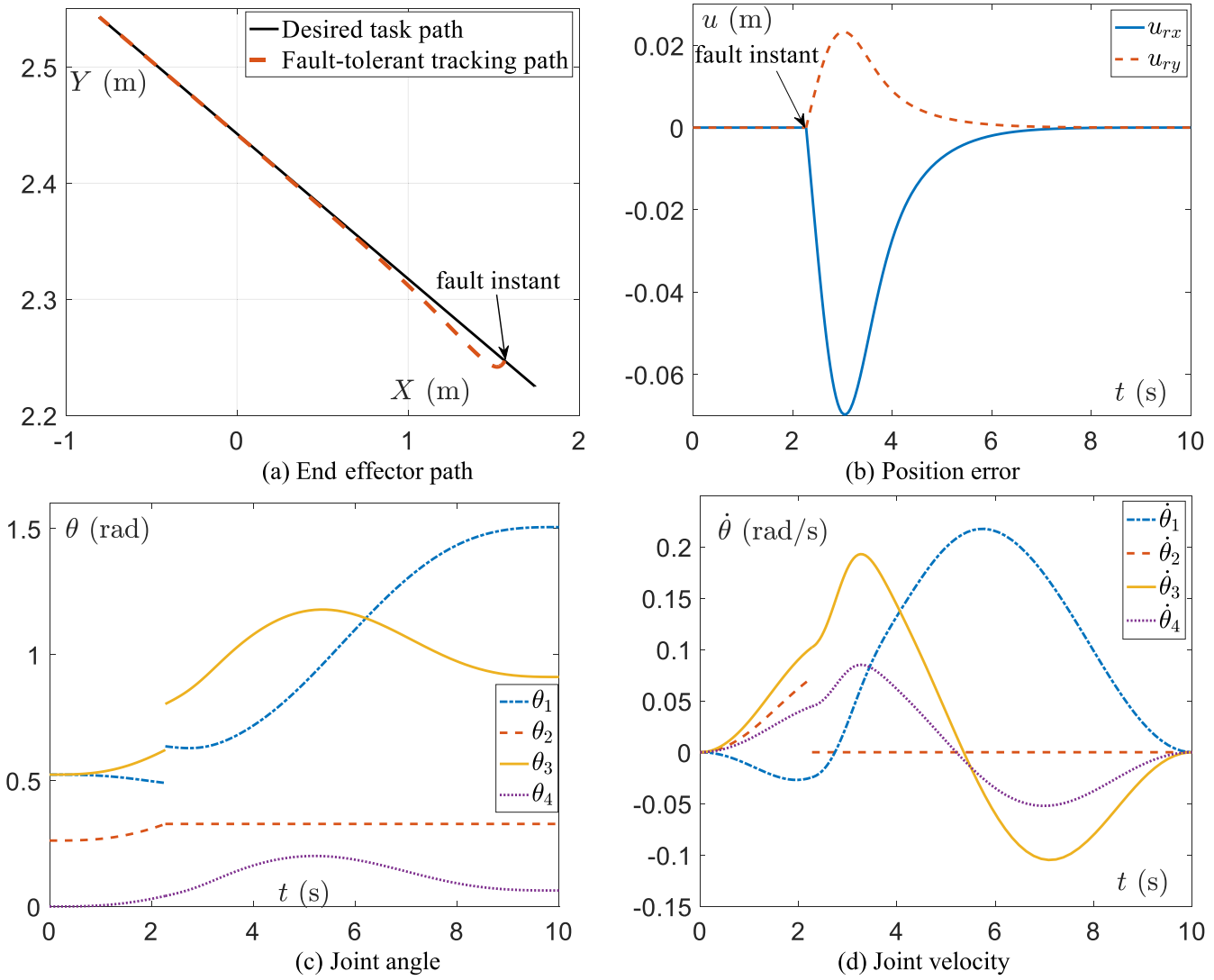


FIGURE 5. Simulation results of straight line trajectory tracking based on pseudo-inverse degradation when the 2nd joint failure occurs.

and inverse-free methods respectively when the joint failure makes the original scheme unfeasible.

The task is a straight line from the start point $p(0) = (1.74229368285670, 2.22479641649612)$ m to $p(10) = (-0.804185406613624, 2.54310630267991)$ m. The task duration is $T = 10$ s, the initial joint angle is $[\pi/6, \pi/12, \pi/6, 0]^{\mathrm{T}}$ rad, and the length of the link is $[1,0.8,0.7,0.5]^{\mathrm{T}}$ m. We adopt the ODE 15s of Matlab to conduct the simulation.

A. PSEUDO-INVERSE SIMULATION

This section mainly conducts the simulation analysis on the degradation schemes (2)-(3) based on the pseudo-inverse method. Among them, the parameter λ is set to $\lambda(t) = 2t$, and the corresponding simulation results that can be seen in the Figure 3-7.

For the comparison purpose, we first analyze the original scheme (1)-(2) without joint failure, and Fig. 3 shows the simulation results without joint failure. The end-effector actual and desired path are respectively indicated by a red dotted line and a solid black line in Fig. 3(a). Fig. 3(b) shows the position error of the end-effector in the X and Y directions, respectively. The transients of the joint variables synthesized are given in Figures 3(c) and 3(d). As can be seen from Fig. 3(a) and Fig. 3(b), the robot arm traces the desired path well, and the position error at the end of the task is $\left(1.24 \times 10^{-9}, 4.57 \times 10^{-9}\right)$ m, indicating the effectiveness of the error elimination methods (4)-(11). In addition, in Fig. 3(c) and Fig. 3(d), it can be seen that the joint angle and the joint velocity curve of the manipulator are smooth during the whole motion duration, and the first joint has made the most contribution to the execution of the task (e.g., the joint angle and joint velocity vary the most). From the simulation results and the above analysis, the original schemes (1)-(2) based on the pseudo-inverse method are effective for the trajectory tracking task of the manipulator arm.

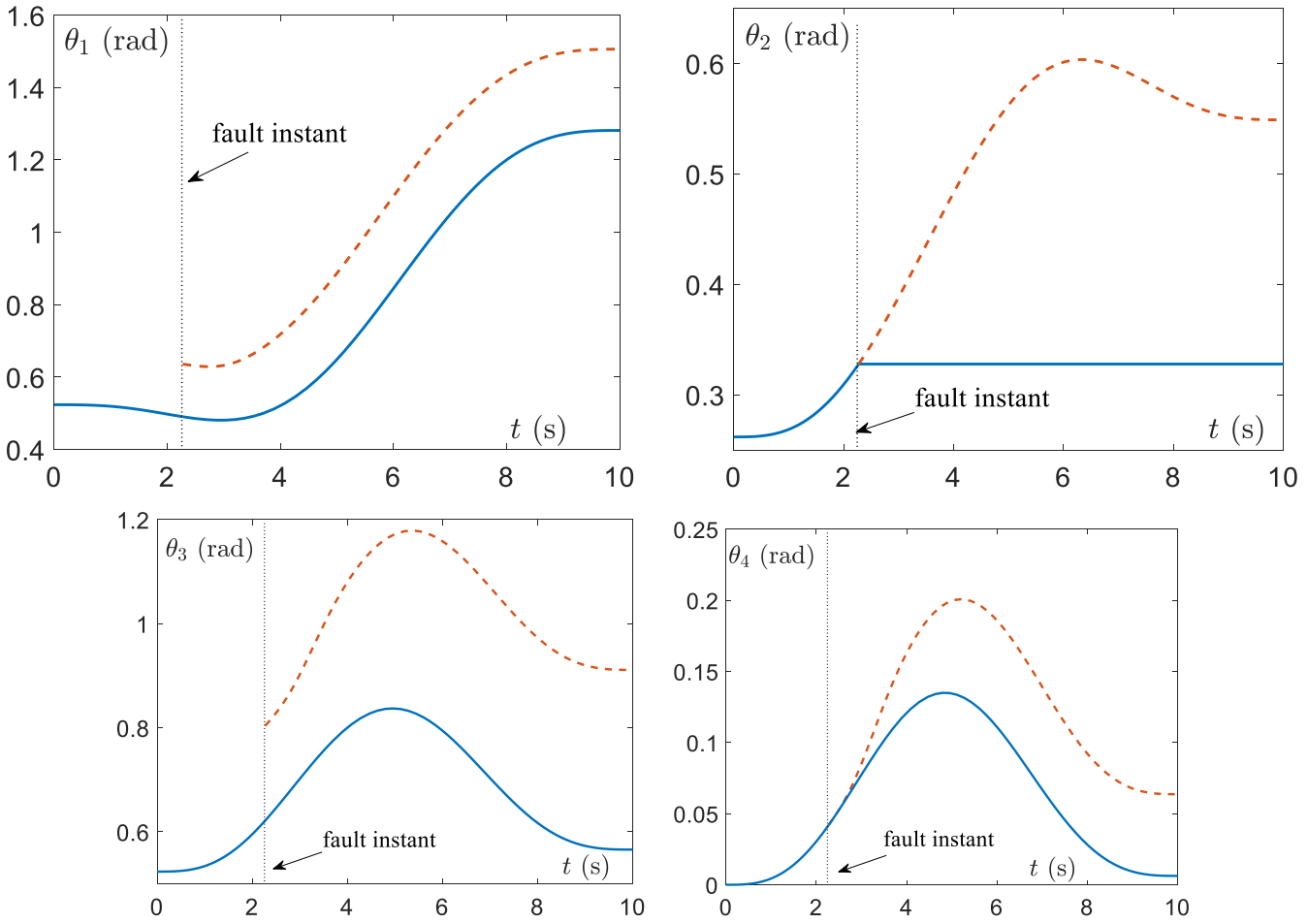


FIGURE 6. Simulation results of joint angle based on pseudo-inverse degradation scheme.

For further comparison, we first locked the 2nd joint during the execution of the task, but did not consider the joint velocity jump in the fault-tolerant processing for analysis and discussion. If the time of failure of Joint-2 of the redundant manipulator is $t = 2.727$ s. The 4-link manipulator is degenerated to a 3-link manipulator after the failure, then the fault-tolerant plan for the degraded manipulator can be calculated by the triangular cosine theorem. At this task simulation, the length of the link is- $[1.776175707294101, 0.7, 0.5]^{\mathrm{T}}$ m, and initial joint angle is [0.635959579947438, 0.803939056723034, $0.042001265750934]^{\mathrm{T}}$ rad. The corresponding simulation results are shown in Figure 4. Fig. 4(a) and Fig. 4(b) respectively describe the changes in joint angle and joint velocity in the case where the 2nd joint is failure and locked, and the vertical dotted line in the figure indicates the time in that the failure occurred. It can be seen from Fig. 4(a) that after the failure of the 2nd joint is locked, the initial joint angle of the 1st and 2nd joints of the degradation scheme (without considering the joint velocity jump) is different from the corresponding joint angles of the original scheme at the failure moment. Therefore, the joint angle curve after the replacement scheme deviates from the curve before the fault. Since the problem of joint velocity jump is not considered in the fault-tolerant scheme, the joint speed of each joint in Fig. 4(b) produces a large jump in the fault-tolerant operation. In particular, the maximum joint speeds of 3rd and 4th joints are 0.171 rad/s and 0.077 rad/s respectively, and are much larger than the joints without failure. In practice, the robot arm may even be accidentally damaged. Therefore, in this case, it is very important to design and apply a fault-tolerant solution that effectively eliminates joint velocity jumps.

As mentioned above, this article proposes a fault-tolerant analytic scheme based on pseudo-inverse (2)-(3). This scheme replaces the original scheme with the degradation method to achieve non-jumping of joint velocity. The corresponding simulation results are shown in the figure. 5 - Figure 7. It can be seen from Fig. 5(a) and Fig. 5(b) that the end of the arm deviates from the desired trajectory for a short period of time after the failure, and then the end trajectory quickly returns to the desired trajectory, and the position error at the end point is $(1.40742 \times 10^{-6}, 4.28595 \times 10^{-6})$ m. In addition, it can be seen from Fig. 5(a) and Fig. 5(b) that although the joint angle inevitably deviates from

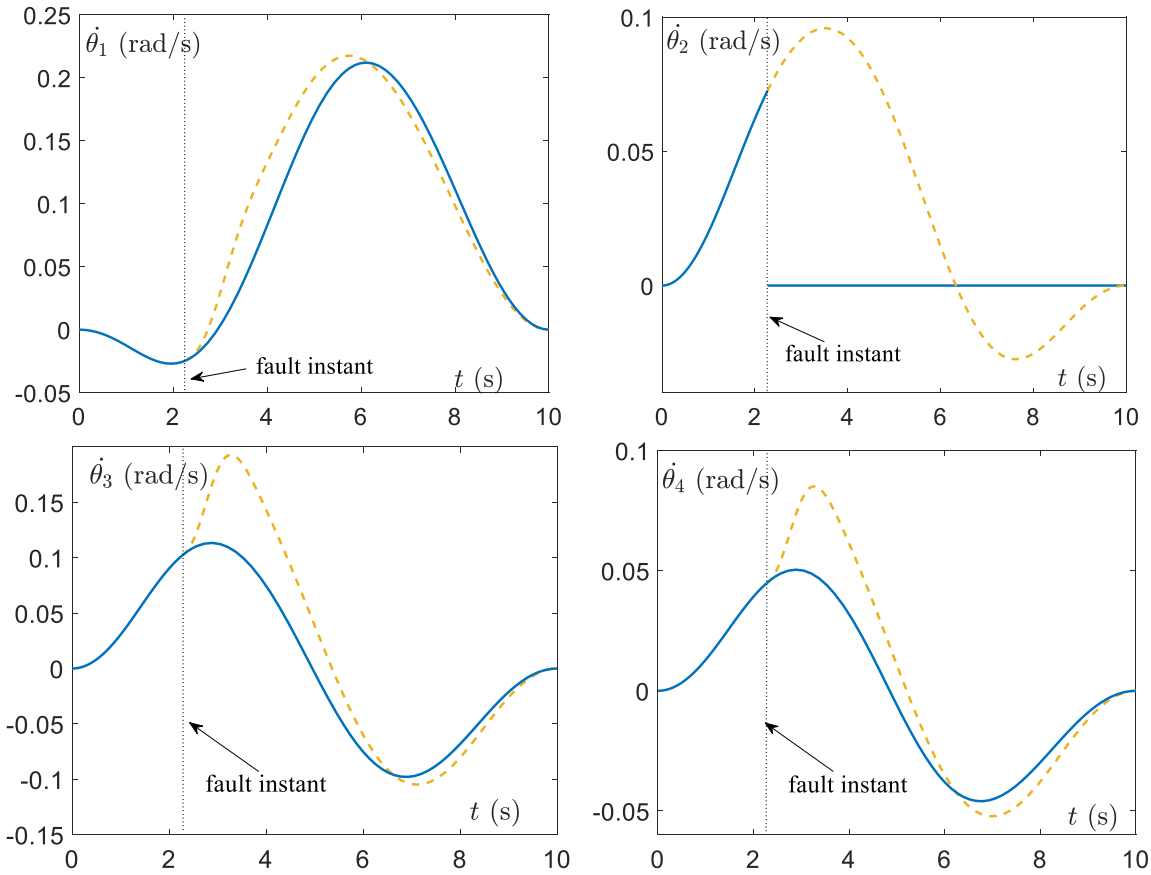


FIGURE 7. Joint velocity simulation result based on pseudo-inverse degradation scheme.

the previous trajectory after the failure, the joint velocity of the manipulator does not cause any sudden change, which is in line with our degradation scheme design. In particular, for the sake of contrast, Figures 6 and 7 show the change in joint angle and joint velocity of each joint before and after fault tolerance. As can be seen in Fig. 6, the arm joint is degraded due to the failure of the manipulator, and the failure posture of the second joint remains. The joint angle does not change, and the joint of the second joint is compensated by the first and third joints. It can be seen from Fig. 7 that the velocity of the 2nd joint is mainly compensated by the 3rd and 4th joints after the 2nd joint is locked, and the change of the joint velocity is a continuous and relatively smooth. In addition, after the mission is completed, the joint velocity of all joints of the robot arm returns to 0, which is in line with practical applications. These simulation results and analysis demonstrate the effectiveness of the pseudo-inverse method based degradation schemes (2)-(3) on joint fault tolerance and the elimination of joint velocity jumps. Compared with the position error at the end of the original scheme task, and the accuracy of the task execution of the robot arm needs to be improved.

B. INVERSE-FREE SIMULATION

In order to discuss the effectiveness of the degradation scheme based on the inverse-free method (Eq. (12), Eq. (3)), simulation is also done in this section. Among them, the parameters are $\eta = 1 \times 10^4$, $\beta = 2.5$. The corresponding sim-ulation results can be seen in Figure 8 and Table 1.

Figure 8 shows the simulation results of the linear trajectory tracking based on the inverse-free degradation scheme of the redundant manipulator in the second joint failure. It can be seen from Fig. 8(a) and Fig. 8(b) that although the end of the arm deviates from the desired trajectory after the replacement due to the failure of the second joint, it still returns to the desired path very quickly. And it has a good tracking accuracy. In addition, Fig. 8(c) and Fig. 8(d) are joint angles and joint velocity curves of the manipulator when the second joint is error-free and subjected to fault tolerance treatment, respectively. The thicker curve in the figure represents the simulation result of fault-tolerant processing, and the thinner curve shows the simulation result of joint failure. In the Figure 8 (c) and Figure 8 (d), similar to the simulation results of the pseudo-inverse method, the joint angle changes more smoothly, and there is non-jumping in the joint velocity when the joint is fault-tolerant, and mainly consists of 3 joints. And the 4 joints compensate the joint velocity for the locked 2nd joint. In addition, in order to better compare our proposed deg-radation scheme with the pseudo-inverse method, the end position error of the manipulator based on two different analytical methods after the replacement scheme is given

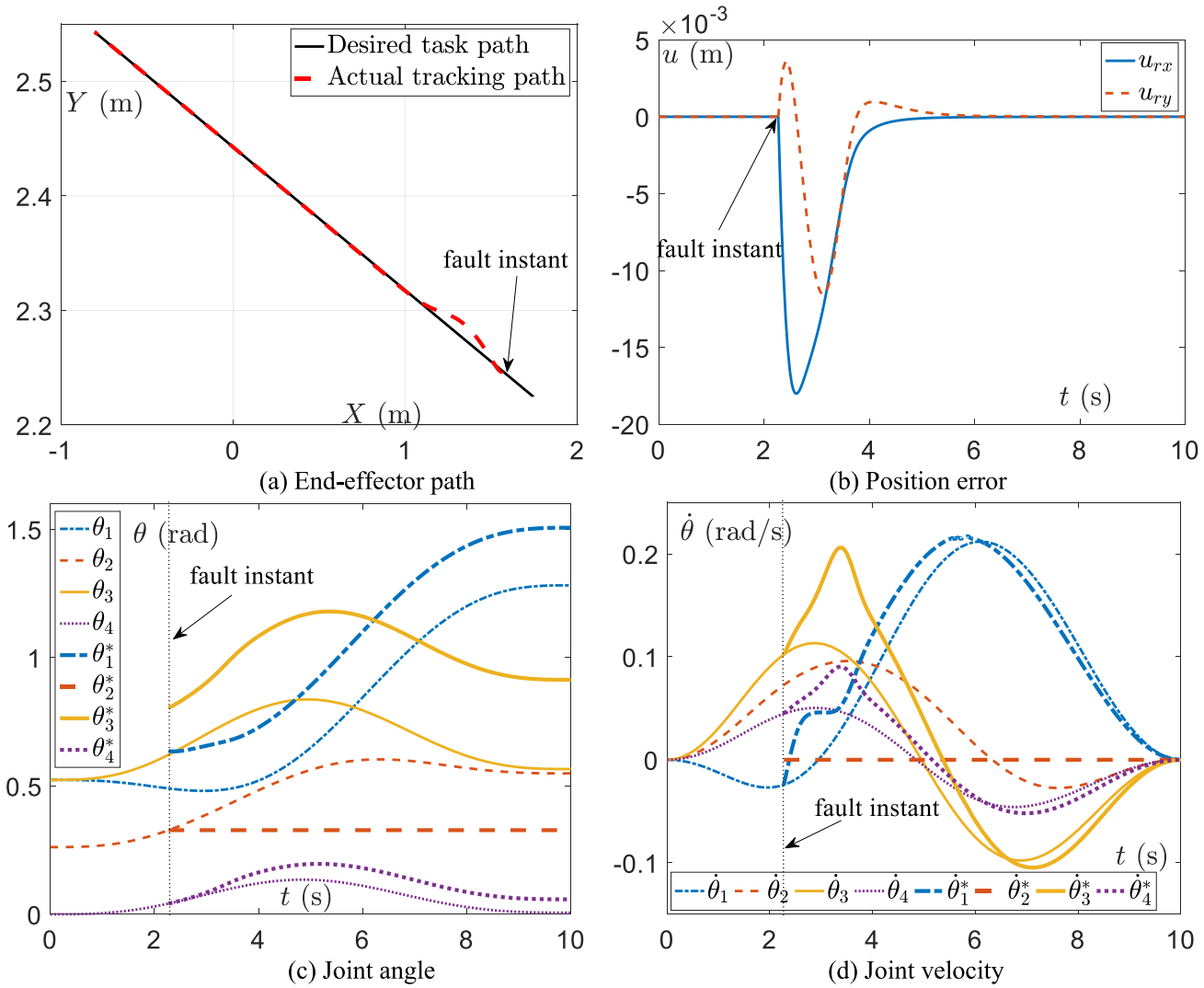


FIGURE 8. Linear trajectory tracking results based on the inverse-free degradation scheme for the 2nd joint failure.

TABLE 1. Position error of the end in the X and Y directions when the pseudo-inverse scheme and the inverse-free scheme track the linear trajectory after the 2nd joint failure.

t/s	PI SCHEME $u(t)$/m	INVERSE-FREE SCHEME $u(t)$/m
2.27	$(1.77240\times10^{-10}, 6.75448\times10^{-9})$	$(1.77240\times10^{-10}, 6.75448\times10^{-9})$
3.72	$(-0.03887, 0.01266)$	$(-0.00232, -4.57085\times10^{-4})$
5.05	$(-0.00650, 0.00225)$	$(-9.39816\times10^{-5}, 2.92581\times10^{-4})$
7.02	$(-1.19489\times10^{-4}, 1.794196\times10^{-4})$	$(-1.19489\times10^{-5}, 2.44097\times10^{-5})$
10	$(1.40742\times10^{-6}, 4.28595\times10^{-6})$	$(-3.27059\times10^{-10}, 8.78700\times10^{-10})$

in Table 2. From the data in Table 2, it can be found that the inverse-free degradation scheme enables manipulator end-effector to return to the desired path more quickly after the joint failure. And the $(-3.27059 \times 10^{-10}, 8.78700 \times 10^{-10})$ m position error at the end of the task is smaller than the position error of the original scheme. This shows that the program has better working accuracy. The above simulation analysis results show that in the case of manipulator joint failure, replacing the original scheme with the inverse-free degradation scheme can not only enhance the fault tolerance

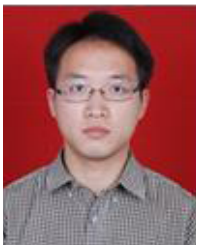
of the algorithm, but also make up for the short-comings of high computational complexity and insufficient task execution precision. It also illustrates the effectiveness and accuracy of the degradation inverse-free scheme.

V. CONCLUSION

Based on the pseudo-inverse and inverse-free redundancy analysis methods, this article proposes a fault-tolerant motion planning scheme that can be used for eliminating the joint velocity jump of the manipulator. By using the joint velocity of the manipulator at the time of failure as the initial joint velocity of the degradation scheme to plan the motion of the degraded manipulator, the joint velocity jump elimination of the manipulator is achieved. The planar 4-link redundant manipulator is used as the simulation model, and two different analytical schemes are simulated in the case of the 2nd joint failure. The simulation results illustrate the validity and accuracy of the proposed fault-tolerant scheme, and compared with the pseudoinverse method, the degradation scheme based on the inverse-free method has the advantages of lower computational complexity and higher precision. The proposed method may also be applied to the collaborative control of multi-redundant manipulators, which will be the future research work.

REFERENCES

[1] M. Yazdani, R. S. Novin, M. T. Masouleh, M. B. Menhaj, and H. Abdi, "An experimental study on the failure tolerant control of a redundant planar serial manipulator via pseudo-inverse approach," in *Proc. 3rd RSI Int. Conf. Robot. Mechatronics (ICROM)*, Tehran, Iran, Oct. 2015, pp. 365–370.
[2] L. Jin, B. Liao, M. Liu, L. Xiao, D. Guo, and X. Yan, "Different-level simultaneous minimization scheme for fault tolerance of redundant manipulator aided with discrete-time recurrent neural network," *Frontiers Neurorobotics*, vol. 11, pp. 1–7, Sep. 2017.
[3] R. Tinos, M. H. Terra, and M. Bergerman, "Fault tolerance in cooperative manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, USA, May 2002, pp. 470–475.
[4] D. Guo, Q. Feng, and J. Cai, "Acceleration-level obstacle-avoidance scheme for motion planning of redundant robot manipulators," *IEEE Access*, vol. 7, pp. 183040–183048, Dec. 2019.
[5] K. Li and Y. Zhang, "Fault-tolerant motion planning and control of redundant manipulator," *Control Eng. Pract.*, vol. 20, no. 3, pp. 282–292, Mar. 2012.
[6] R. G. Roberts, S. A. Siddiqui, and A. A. Maciejewski, "Designing equally fault-tolerant configurations for kinematically redundant manipulators," in *Proc. 41st Southeastern Symp. Syst. Theory*, Tullahoma, TN, USA, Mar. 2009, pp. 335–339.
[7] J. Zhao and Q. Li, "On the joint velocity jump for redundant robots in the presence of locked-joint failures," in *Proc. Chin. Condiment*, 2008, pp. 109–117.
[8] S. Huang, Y. Peng, W. Wei, and J. Xiang, "Clamping weighted least-norm method for the manipulator kinematic control with constraints," *Int. J. Control*, vol. 89, no. 11, pp. 2240–2249, Nov. 2016.
[9] 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.
[10] J. Wan, H. Wu, R. Ma, and L. Zhang, "A study on avoiding joint limits for inverse kinematics of redundant manipulators using improved clamping weighted least-norm method," *J. Mech. Sci. Technol.*, vol. 32, no. 3, pp. 1367–1378, Mar. 2018.
[11] M. Rouhani and S. Ebrahimabadi, "Inverse kinematics of a 7-DOF redundant robot manipulator using the active set approach under joint physical limits," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 25, no. 5, pp. 3920–3931, 2017.
[12] H. Abdi and S. Nahavandi, "Joint velocity redistribution for fault tolerant manipulators," in *Proc. IEEE Conf. Robot., Autom. Mechatronics*, Singapore, Jun. 2010, pp. 492–497.
[13] Z. Jing, Y. Yanbin, and Y. Xuebin, "On the sudden change in joint velocity during fault tolerant operations for spatial coordinating redundant manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Beijing, China, Oct. 2006, pp. 3885–3890.
[14] Z. Jing, L. Yu, and J. Yi, "On path errors of redundant robots in fault tolerant operations for locked joint failures," in *Proc. IEEE Int. Conf. Adv. Robot.*, Beijing, China, Jun. 2009, pp. 1–6.
[15] Z. Jing and F. Cheng, "On the joint velocity jump during fault tolerant operations for manipulators with multiple degrees of redundancy," *Mechanism Mach. Theory*, vol. 44, no. 6, pp. 1201–1210, Jun. 2009.
[16] H. Abdi, S. Nahavandi, Y. Frayman, and A. A. Maciejewski, "Optimal mapping of joint faults into healthy joint velocity space for fault-tolerant redundant manipulators," *Robotica*, vol. 30, no. 4, pp. 635–648, Jul. 2012.
[17] Q. Jia, T. Li, G. Chen, H. Sun, and J. Zhang, "Velocity jump reduction for manipulator with single joint failure," in *Proc. Int. Conf. Multisensor Fusion Inf. Integr. Intell. Syst. (MFI)*, Beijing, China, Sep. 2014, pp. 1–6.
[18] L. Xiao, Z. Zhang, Z. Zhang, W. Li, and S. Li, "Design, verification and robotic application of a novel recurrent neural network for computing dynamic Sylvester equation," *Neural Netw.*, vol. 105, pp. 185–196, Sep. 2018.
[19] 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, Oct. 2018.



JIAWEI LUO received the B.S. degree in electronic information engineering from Nanchang Hangkong University, Nanchang, China, in 2011, and the M.S. degree in detection technology and automatic equipment from the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China, in 2015. He is currently pursuing the Ph.D. degree in control science and engineering with the School of Electrical and Automation Engineering, East China Jiaotong University, Nanchang. His research interests include robotics and intelligent control systems.

KENE LI (Member, IEEE) received the Ph.D. degree from Sun Yat-sen University, Guangzhou, China, in 2011. He has been an Associate Professor with the School of Electrical and Information Engineering, Guangxi University of Science and Technology, since 2012. From May 2017 to May 2018, he worked as a Visiting Scholar with the Department of Mechanical, Industrial and Systems Engineering, University of Rhode Island, USA. He works on robotics and intelligent control.

HUI YANG (Member, IEEE) received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2004. He is currently a Full Professor and the Vice President of the East China Jiaotong University, Nanchang, China. His main research interests include autonomous driving of high-speed rail-way, robotics, and intelligent control.

JIN YANG received the B.S. degree in measurement and control technology and instrumentation and the M.S. degree from the School of Electrical and Information Engineering, Guangxi University of Science and Technology (GXUST), Liuzhou, China, in 2019. Her current research interests include robotics and control algorithm.