Inverse Kinematics Solution Method Based on Workspace Analysis and Iterative Step Coefficient Adjustment of General Robot

Abstract—**General robots do not meet the Pieper criterion and require swift, precise inverse kinematics solutions for accurate control. Current numerical techniques lack universal applicability and desired convergence. This study proposes a numerical iterative method that combines workspace analysis with iterative step coefficient adjustment. An error function is constructed, and the manipulator's workspace is delineated using Monte Carlo random sampling. The spatial point with the minimum error function is identified, and its joint variables are used as optimal initial inputs. An iterative step coefficient, adjusting in real time based on adjacent iteration steps, ensures rapid convergence within a predefined error threshold. Experimental validation of the proposed method was performed on the manipulator of the anchor drilling robot. The average angle errors along the X, Y, and Z axes were 2.1147°, 2.1426°, and 1.1800°, respectively, while the position errors were approximately 0.001 mm. In comparison with the conventional method, the angle errors were significantly reduced by 61.27%, 50.91%, and 45.65%, respectively. Additionally, the computation time and the average number of iterations were decreased by 17.99% and 20.01%, respectively. Experimental results demonstrate its effectiveness for general robots with different structures, marking a significant advancement in motion control technologies for manipulators.**

Index Terms—**Inverse kinematics, Workspace analysis, Iterative step coefficient, Monte Carlo, General manipulators**

I. Introduction

Robots combine the knowledge of various disciplines, such as machinery [1,2], electronics [3,4], sensors [5,6], computers, and artificial intelligence [7,8], which involve diverse cutting-edge technologies. Robots are widely used in automatic production [9,10], automobile manufacturing [11], logistics [12], medical treatment [13,14] and other industries [15,16]. Therefore, they are extensively studied owing to their significant development prospects. Currently, manipulators, which are a type of robot, are widely used.

An inverse kinematics analysis of the manipulators was performed to calculate the specific values of each joint based on the spatial position of the manipulator end-effector, which forms the basis of robot path planning, navigation, and automatic control. For most industrial manipulators, the geometric structure satisfies the Pieper criterion and the calculation formula for each joint can be obtained through a transformation. Furthermore, all possible inverse solutions can be obtained rapidly and with high accuracy. In contrast, the inverse solution of a general manipulator that does not satisfy Pieper's criterion does not yield the expression of its joint variables; hence, rapidly and accurately obtaining an inverse solution of a general manipulator is challenging. Owing to the increasing application of general manipulators in certain industries, a rapid and effective method must be developed to solve their inverse kinematics.

This work was supported in part by the National Natural Science Foundation of China under Grant 52104166 and 52204176, and in part by China Postdoctoral Science Foundation funded project under Grant No.2022MD723826.

To obtain an inverse kinematics solution for general manipulators, scholars and experts worldwide have conducted comprehensive investigations and achieved notable results. Currently, the most typical methods used to solve the inverse kinematics problems of general manipulators are numerical methods [17], intelligent algorithms, and geometry methods [18].

The process of solving the inverse kinematics of a general manipulator using intelligent algorithms is essentially the transformation of the kinematic equations into an optimal control problem. The solution process involves optimizing the combination of joint variables to ensure that the end-effector can reach the desired position with the expected orientation, which can be viewed as an optimization problem. Intelligent algorithms, under given constraints, search for the optimal solution to achieve this objective, thereby attaining optimal control of the system[19,20]. The intelligent algorithms primarily include particle swarm optimization (PSO) [21], neural networks [22,23], battle royal optimization [24], deep learning [25,26], and firefly algorithms [27]. Because the PSO algorithm has fewer parameters, a simple configuration, and easy control, it is more efficient and commonly used in solving optimization problems than other intelligent algorithms [28,29]. Moreover, the PSO algorithm is time-consuming when computing the inverse solution in a complex multidimensional robot model, and it is easily trapped in the local optimum; thus, it does not provide adequate accuracy. To overcome these limitations, many researchers have improved the PSO algorithm. S.M. proposed a hybrid algorithm, e3GSA [30], which is a hybrid of the gravitational search algorithm (GSA) and PSO, for solving the inverse kinematics problem of an n-link redundant robot manipulator. Ram [31] conceived an optimization problem using a bidirectional PSO method that can reduce the time required to solve the inverse kinematics solution. To enhance the performance of the algorithm in searching for the global optimum, adaptive PSO (APSO) [32], a novel parallel learning PSO (PLPSO) [33] method, and a quantum particle swarm algorithm (QPSO) [34,35] were proposed, and the experimental results demonstrated that these methods could efficiently and accurately solve the inverse kinematic problem of multiple degrees-of-freedom (DOF) manipulators.

The geometry method, which uses geometric and trigonometric relationships, was also used to resolve the inverse kinematics. A new and improved differential evolution-constrained optimization algorithm was proposed to solve the redundancy of manipulators, which offers a significantly higher convergence accuracy and convergence rate than the conventional mutation differential evolution algorithm [36,37]. Song [38] used a new numerical sequence processing method with a closed-loop framework to solve the inverse kinematics of a seven-DOF redundant manipulator. Zhang [39] proposed a novel inverse kinematics solution framework combining unique domains and simplified the objective function using a kinematic decoupling method based on improved disconnection and reconnection. The simulation results confirmed the effectiveness of their proposed method. However, structural distortions caused by factors such as assembly errors and the processing environment may cause the criterion to fail.

Numerical methods typically require the inversion of a Jacobian matrix, which is complicated and time-consuming. Lechuga-Gutierrez [40] presented a set of generalized iterative algorithms for obtaining the inverse position kinematics of n-DOF kinematic chains with revolute joints. In addition, simulation experiments were performed using different DOF models, and real-time experiments were conducted to demonstrate the effectiveness of the proposed methods. The inverse solution of the desired state and joint optimum state of grippers can be regarded as minimum-value problems that can be solved using the Gaussian–Newton iterative algorithm [41].

Although researchers have progressed in solving the inverse kinematics of general manipulators, numerous problems remain unsolved. The PSO algorithm requires a significant amount of time to obtain the inverse, and the solution easily falls into a local optimum and does not provide sufficient accuracy. The neural network method is suitable for solving complex nonlinear problems; however, it requires a significant amount of training data to establish accurate mapping relationships and cannot easily obtain a significant amount of accurate inverse solution data. Thus, the generalizability of the network is not guaranteed. The numerical method is suitable for any manipulator; however, it has disadvantages such as a high calculation cost, long execution time, and large numerical errors. The geometric method is suitable for certain manipulators; however, its calculation process is complex and its universality is unsatisfactory. The Newton–Raphson method, although possessing high convergence speed and precision, depends significantly on the selection of the initial points. If the initial points are unreasonable, the algorithm may fall into a local optimal solution or fail to converge. In contrast, selecting a reasonable initial value allows the inverse solution of the manipulator to be solved rapidly and accurately.

To address these issues, this paper introduces a novel approach that combines workspace analysis and iterative step coefficient adjustment to solve the inverse kinematics problem of a general manipulator. In this study, the inverse kinematics problem of a general manipulator was transformed into an optimization problem subject to constraints, and iterative step coefficient were employed to obtain a numerical solution. Our simulation and experimental results confirmed the efficacy and precision of the proposed methodology. The primary contributions of this study are summarized as follows.

(1) A novel and versatile approach for solving the inverse kinematics problem of general manipulators that combines workspace analysis and step coefficient adjustment is presented. The simulation results demonstrate that this method yields high-precision results within a significantly reduced computational time.

(2) An error function is constructed to describe the position and posture error of the end-effector, and the joint variable value corresponding to its minimum value is used as the optimal initial value of the iterative method, thus avoiding the problem that the random initial value is difficult to converge.

(3) A mechanism for automatic adjustment of the iterative step coefficient is introduced, contingent upon the two adjacent iteration values. This method effectively circumvents oscillatory behavior induced by excessively large step sizes and ameliorates convergence challenges attributed to overly diminutive step increments.

The remainder of this paper is organized as follows. Section II presents a kinematic analysis of the manipulator of an anchor drilling robot. Section III introduces the inverse kinematics solution method applicable to general manipulators, where both the Monte Carlo and Newton–Raphson iteration techniques are leveraged. Section IV presents the experimental procedures and subsequent analysis of the obtained results. Finally, Section V concludes the paper.

II. Kinematic Analysis Of Anchor Drilling Robot

A. Structure of anchor drilling robot

As shown in Fig. 1, an anchor drilling robot is an automatic supporting equipment for the roof of a coal mine roadway, which comprises a cantilever roadheader and two multi-degree-of-freedom manipulators, along with a variety of high-precision sensors, such as visual sensors and displacement sensors. These components enable the robot to perform rapid tunnel excavation, automatic recognition of drilling holes, and autonomous control, thereby improving tunnel support efficiency.

The manipulators on both sides of the anchor drilling robot are identical in structure, each consisting of a six-degree-of-freedom manipulator with four revolute joints and two prismatic joints. These include the boom swing joint, boom pitch joint, boom telescopic joint, arm pitch joint, arm swing joint, and arm telescopic joint. To satisfy the requirements in certain scenarios, the manipulator was designed as a general manipulator, which does not satisfy the Pieper criterion.

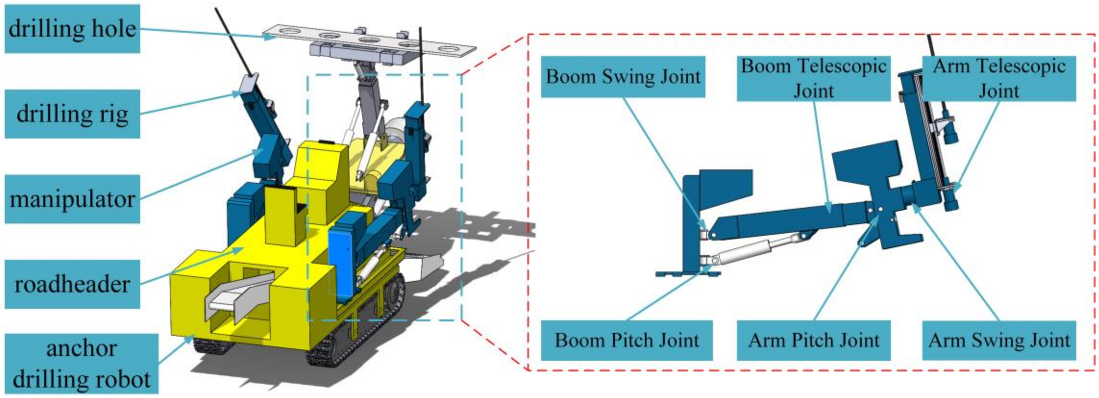


Fig. 1. Application of anchor drilling robot in coal mine roadway support.

The procedure for supporting coal mine roadways is outlined as follows. The positioning of the roof drilling hole was determined using a camera system integrated into the drilling rig, and this location was designated as the target point. Subsequently, an inverse kinematics computation was employed to derive the motion parameters for each joint, enabling precise control over the movement of the manipulator as it progressed towards the target position. This process culminates in the successful completion of the supporting task. The accuracy of the inverse kinematics solution directly affects the efficiency of roadway support operations. Hence, the development of a rapid and precise method for resolving the inverse kinematics of such general manipulators is of substantial significance. To obtain the inverse solution for the general manipulator, kinematic analysis was first performed.

B. Forward kinematics

The manipulator can be described using the Modified Denavit-Hartenberg (MDH) model, and its kinematic chain is shown in Fig. 2, where $x_i y_i z_i\left(i=0,1,\ldots 6\right)$ represents the coordinate system established on each joint, whereas $x_0 y_0 z_0$ is the base coordinate, and the $x_1 y_1 z_1$ coordinate is coincident with the base coordinate. The six-DOF manipulator including four revolute joints and two prismatic joints. Further, θ_1, θ_2, θ_4, and θ_5, denote the rotation angle of the corresponding joints, whereas d_3 and d_6 denote the translation distances of the corresponding joints. Based on the established joint coordinate system, the DH parameter table of the manipulator can be derived by measuring its relevant parameters. In the DH parameter table of the manipulator, as shown in Table 1, θ_i is the angle that rotates from the x_{i-1} to x_i axes measured along the z_{i-1} axis; d_i is the distance that translates from the x_{i-1} to x_i axes measured along the z_{i-1} axis; a_{i-1} is the distance that translates from the z_{i-1} to z_i axes measured along the x_i axis; and α_{i-1} is the angle that rotates from z_{i-1} to z_i about the x_i axis.

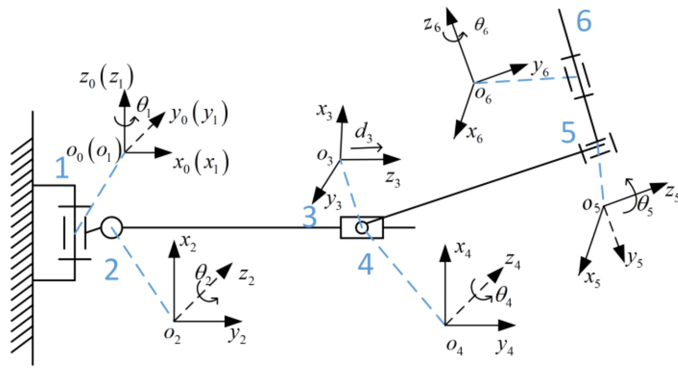


Fig. 2. The model of the manipulator based on MDH method.

Table 1
DH parameters of the six-DOF manipulator.

i	$\alpha_{i-1}/°$	a_{i-1}/mm	d_i/mm	$\theta_i/°$
1	0	0	0	θ_1
2	-90°	0	0	θ_2-90°
3	-90°	0	$1000+d_3$	0
4	90°	0	0	θ_4
5	-90°	0	0	θ_5+90°
6	90°	0	0	θ_6

The homogeneous transformation matrix can be used to describe the rotation and translation relationships between two coordinate systems and can be expressed in terms of the coordinate frameworks $\{o_i x_i y_i z_i\}$ and $\{o_{i-1} x_{i-1} y_{i-1} z_{i-1}\}$. By substituting these parameters into equation (1), the homogeneous transformation matrix for the adjacent joint coordinate systems can be obtained.

$$ {}^{i-1}_{\ \ i}T=\begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

The forward kinematics equation can be expressed as the product of the homogeneous matrices that represent each link from beginning to end. The homogeneous transformation matrices of the end-effector relative to the base are obtained by multiplying all the homogeneous transformation matrices, which are expressed as follows:

$$ {}^{0}_{6}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}, \qquad (2) $$

where R is a rotation matrix that represents the orientation of the end-effector coordinate framework with respect to the base coordinate framework, and vector p represents its position. The transformation matrix of the manipulator end-effector in the base coordinate system can be obtained by substituting the values of a set of joints into the above equation.

III. Inverse Kinematics Method Based On Monte Carlo Method

A. Objective function

Using the six-DOF general manipulator of the anchor drilling robot described herein as an example, the position and attitude of the end-effector relative to the base coordinate system can be expressed as shown by the following homogeneous transformation matrix.

$$ {}^{0}_{6}T_{\mathrm{d}} = \begin{bmatrix} n_{xd} & o_{xd} & a_{xd} & p_{xd} \\ n_{yd} & o_{yd} & a_{yd} & p_{yd} \\ n_{zd} & o_{zd} & a_{zd} & p_{zd} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_d & p_d \\ 0 & 1 \end{bmatrix} \qquad (3) $$

where ${}^{0}_{6}T_{\mathrm{d}}$ is the transformation matrix of the desired position and attitude of the end-effector in the base coordinate system, R_d is the rotation matrix, and p_d is the translation matrix. Therefore, ${}^{0}_{6}T_{\mathrm{d}}$ indicates that the coordinate system obtained by the rotational and translational changes in the base coordinate system is consistent with the desired position and attitude of the end-effector.

Euler angles are a commonly used method to describe rotational transformations, however, they have certain limitations, particularly the potential issue of gimbal lock. Specifically, when certain specific angles occur, such as when the second rotation angle reaches ±90°, the system loses a degree of freedom. In addition, the Euler angles calculated from the same rotation matrix may not be unique. To avoid these problems, this article chooses to use quaternions to describe rotational transformations.

A quaternion q is used to represent the rotational change of a coordinate system, and its general form can be expressed as $q = \omega + xi + yj + zk$, where ω, x, y, and z are real numbers, and they satisfy the condition $\omega^2 + x^2 + y^2 + z^2 = 1$. Additionally, a quaternion q can also be represented as $q = [\omega, x, y, z]$. For a given rotation matrix $R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$, we can convert it into a corresponding quaternion $q = [\omega, x, y, z]$ for representation.

Firstly, we can calculate the scalar part ω of the quaternion using the following equation:

$$ \omega = \frac{\sqrt{1 + r_{11} + r_{22} + r_{33}}}{2} \qquad (4) $$

This computation is based on the trace of the matrix (the sum of diagonal elements), ensuring a positive outcome.

Due to various possibilities, we must choose an appropriate calculation path based on the maximum element in the matrix to avoid division by zero and ensure accuracy. Here are the case discussions based on the dominant diagonal element:

If r_{11} is the largest:

$$ \begin{cases} x = \dfrac{r_{32} - r_{23}}{4\omega} \\ y = \dfrac{r_{13} - r_{31}}{4\omega} \\ z = \dfrac{r_{21} - r_{12}}{4\omega} \end{cases} \qquad (5) $$

If r_{22} is the largest:

$$ \begin{cases} x = \dfrac{r_{13} - r_{31}}{4\omega} \\ y = \dfrac{r_{21} + r_{12}}{4\omega} \\ z = \dfrac{r_{32} + r_{23}}{4\omega} \end{cases} \qquad (6) $$

If r_{33} is the largest:

$$ \begin{cases} x = \dfrac{r_{21} - r_{12}}{4\omega} \\ y = \dfrac{r_{31} + r_{13}}{4\omega} \\ z = \dfrac{r_{32} + r_{23}}{4\omega} \end{cases} \qquad (7) $$

Thus, for a given rotation matrix R, it can be represented in quaternion form utilizing equations (4) through (7).

Therefore, for any desired spatial position and attitude of the end-effector of the general manipulator, the homogeneous transformation matrix ${}^{0}_{6}T_{\mathrm{d}}$ of the end-effector relative to the base coordinate system can be determined. For the rotation matrix R_d, the quaternion $q_d = [\omega_d, x_d, y_d, z_d]$ can be obtained using the aforementioned equation. p_d represents the translation of the base coordinate system along the coordinate axis, that is, p_{xd}, p_{yd}, and p_{zd} along the X, Y, and Z axes, respectively. Therefore, we construct a transformation vector $M_d = [\omega_d, x_d, y_d, z_d, p_{xd}, p_{yd}, p_{zd}]$, which represents the transformation vector of a desired spatial position and attitude from the base coordinate system to the end-effector, where the first four elements of the vector represent a quaternion for rotation and the last three elements of the vector indicating distances of translation along the XYZ coordinate axis.

For the six-DOF general manipulator in this study, for a set of joint variables $\theta_c = [\theta_{1c}, \theta_{2c}, d_{3c}, \theta_{4c}, \theta_{5c}, d_{6c}]$, the transformation matrix corresponding to the current set of joint variable values can be obtained using (8).

$$
\begin{aligned}
{}^{0}_{6}T_{c} &= {}^{0}_{1}T\left(\theta_{1c}\right) {}^{1}_{2}T\left(\theta_{2c}\right) {}^{2}_{3}T\left(d_{3c}\right) {}^{3}_{4}T\left(\theta_{4c}\right) {}^{4}_{5}T\left(\theta_{5c}\right) {}^{5}_{6}T\left(d_{6c}\right) \\
&= \begin{bmatrix} n_{xc} & o_{xc} & a_{xc} & p_{xc} \\ n_{yc} & o_{yc} & a_{yc} & p_{yc} \\ n_{zc} & o_{zc} & a_{zc} & p_{zc} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_c & p_c \\ 0 & 1 \end{bmatrix}
\end{aligned} \tag{8}
$$

Similarly, based on the rotation matrix R_c corresponding to the current joint variable value, the corresponding quaternion $q_c = [\omega_c, x_c, y_c, z_c]$ can be obtained using the aforementioned equation. $M_c = [\omega_c, x_c, y_c, z_c, p_{xc}, p_{yc}, p_{zc}]$ represents the transformation vector of the current spatial position and attitude from the base coordinate system to the end-effector.

The difference between the current and desired positions/attitudes of the end-effector can be described by an error function. Therefore, the error function can be described as the difference between the transformation vectors M_c and M_d. The error function is primarily composed of two components, namely R_{error} and P_{error}, where R_{error} signifies the rotational transformation from the current coordinate system to the desired coordinate system, and P_{error} represents the translational transformation from the current coordinate system to the desired one.

In accordance with the preceding content, q_d represents the quaternion expression of the attitude transformation from the base coordinate system to the desired coordinate system, while q_c signifies the quaternion representation of the attitude transformation from the base coordinate system to the current coordinate system. Consequently, the quaternion representation of the attitude transformation between the current coordinate system and the desired coordinate system can be denoted by Δq, which can be expressed as follows:

$$
\Delta q = q_d \bullet q_c^{-1} \tag{9}
$$

where $q_c^{-1} = [\omega_c, -x_c, -y_c, -z_c]$.

The rotation angle corresponding to the quaternion Δq can be calculated using the following formula:

$$
\varphi = 2 \cdot \arccos\left(\left|\omega_{\Delta q}\right|\right) \tag{10}
$$

The obtained rotation angle φ ranges from [0°, 360°], and normalization is performed on it. The normalized angle is then used to describes the difference between the current and desired positions of the end-effector:

$$
R_{error} = \frac{\varphi}{360} \tag{11}
$$

Thus, it can be calculated using the first four elements of the transformation vectors, M_c and M_d. The variable R_{error} was normalized using (11), rendering it a dimensionless parameter. R_{error} is intricately linked to the disparity between the present and desired attitudes of the end-effector. When the attitude corresponding to the current end-effector was consistent with the desired attitude, the value of R_{error} was zero.

Analogously, P_{error} is expressed as

$$
P_{error} = \frac{\left|p_{xc} - p_{xd}\right| + \left|p_{yc} - p_{yd}\right| + \left|p_{zc} - p_{zd}\right|}{\left|p_{xd}\right| + \left|p_{yd}\right| + \left|p_{zd}\right|} \tag{12}
$$

This expression is similar to that of R_{error} The value of P_{error} was proportional to the positional difference. The greater the value of P_{error}, the greater the position error, and vice versa. A P_{error} value of zero implies that the current position is consistent with the desired position.

The error function can be expressed as the sum of R_{error} and P_{error} (as shown in the equation below) and can be used to indicate the current and desired position/attitude of the end-effector for general manipulators.

$$
f\left(M_c, M_d\right) = \rho R_{error} + \left(1 - \rho\right) P_{error}, \tag{13}
$$

where ρ denotes the weighting factor of R_{error} to the error function, whereas $1-\rho$ denotes the weighting factor of P_{error} to the error function. ρ can be adjusted based on importance. In this study, R_{error} and P_{error} are equally important to the error function; therefore, the value of ρ was set to 0.5.

The value of the error function is 0 implies that the current position and attitude of the end-effector are consistent with the desired position and attitude, respectively, and that the joint variable value $\theta_c = \left[\theta_{1c}, \theta_{2c}, d_{3c}, \theta_{4c}, \theta_{5c}, d_{6c}\right]$, corresponding to the current position and attitude, is a set of inverse solutions of the general manipulator. Therefore, the essence of the inverse solution is to determine the minimum error function within the range of each joint value. The corresponding joint variable value is the inverse solution of the manipulator.

B. Select optimal initial value (OIV) based on workspace analysis

The workspace of a general manipulator pertains to a set of spatial coordinates that can be accessed by the end-effector of the manipulator while each joint move within its specified range of motion. The extent of a manipulator workspace is intrinsically related to the structure of the manipulator and the motion range of its constituent joints. The workspace of the manipulator remained invariant for a certain range of motion. Various methodologies exist for determining a manipulator workspace, among which the Monte Carlo method is one of the most widely adopted. The Monte Carlo method, which is based on random sampling, offers a robust solution for addressing complex mathematical challenges, and can be used to characterize the workspace of general manipulators.

To illustrate this, a set of values denoted as θ_i was randomly selected within the range of motion for each joint, such that θ_i must adhere to the following condition:

$$
\theta_{i\min} \le \theta_i \le \theta_{i\max}, \tag{14}
$$

where i denotes the number of joints in the manipulator, θ_i

denotes the value of the i-th joint, and $\theta_{i\min}$ and $\theta_{i\max}$ represent the minimum and maximum ranges of the i-th joint, respectively.

The homogeneous transformation matrix T, which represents the relationship between the end-effector and base coordinate system, can be derived by applying the forward kinematics formula of the manipulator. This matrix describes the position and attitude associated with joint variable values in the spatial coordinate system. By performing numerous iterations, the obtained spatial points can effectively depict the workspace of the manipulator. Increasing the number of iterations resulted in a workspace that closely resembled the actual operating conditions. Every spatial point reached by the end-effector contains important information. First, each spatial point corresponds to a distinct position and attitude of the manipulator's end-effector within the spatial coordinate system and is associated with a homogeneous transformation matrix denoted as T. Second, by performing forward kinematics analysis, the joint variable values Q corresponding to the homogeneous transformation matrix T can be discerned for each of these points. Finally, by applying the aforementioned theory, the transformation vector M can be derived, which enables a transition from the base coordinate system of the manipulator to the coordinate system in which the current end-effector is located. In summary, each point within the spatial domain contains pertinent information, such as the spatial coordinates and attitude of the end-effector, homogeneous transformation matrix T, transformation vector M, and joint variable value Q.

The workspace, which was initially determined using the Monte Carlo method, was partitioned to determine the minimum value of the error function efficiently and precisely. The aim of this partitioning is to identify the minimum error function value along with its associated spatial coordinates within the workspace. Subsequent analysis of these spatial points facilitates the derivation of the corresponding transformation vector M and the relevant joint variable values Q.

The accessible position of the end-effector of the manipulator based on the Monte Carlo method can be approximated as workspace C1 of the manipulator, and its general contour is consistent with the actual workspace of the manipulator. However, they cannot include all contour edge points within a small range. To enhance the system robustness and computational convenience, the boundaries of the circumscribed cube encompassing the workspace should be expanded, and the maximum and minimum values along each coordinate axis must be rounded off to distance them from the origin. This process yielded a larger circumscribed cube (denoted as C2) that served as the redundant workspace of the manipulator. Subsequently, the maximum workspace C2 is segmented into N smaller cubes C3 by applying a fixed step-size e in the X, Y, and Z directions. N can be calculated as follows:

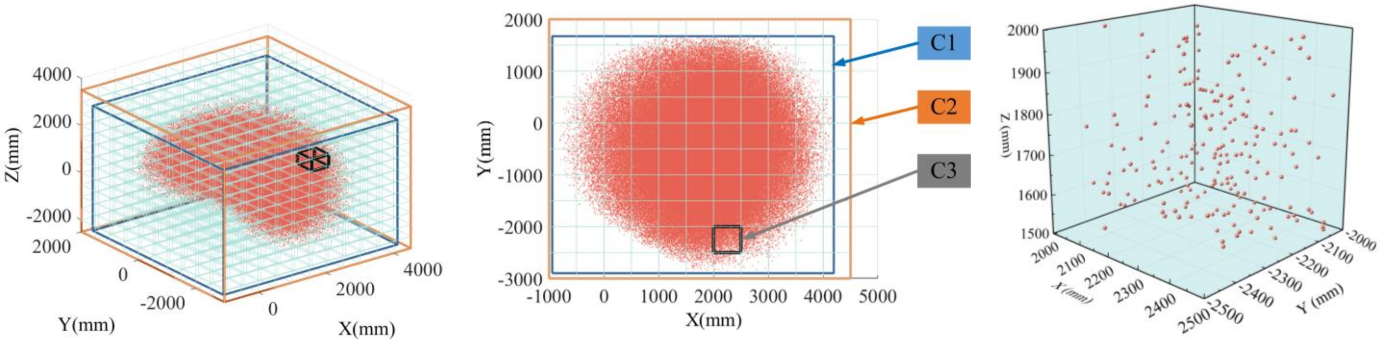
$$N = \frac{X_{c2\max} - X_{c2\min}}{e} * \frac{Y_{c2\max} - Y_{c2\min}}{e} * \frac{Z_{c2\max} - Z_{c2\min}}{e}, \quad (15)$$

where $X_{c2\max}$, $Y_{c2\max}$, and $Z_{c2\max}$ denote the maximum values along the X, Y, and Z axes, respectively, within the C2 workspace, and $X_{c2\min}$, $Y_{c2\min}$, and $Z_{c2\min}$ denote the maximum values along the X, Y, and Z axes, respectively, within the C2 workspace.

The workspace of the manipulator shown was operated based on the aforementioned method, and the results are shown in Fig. 3. In Fig. 3a, the outline W formed by the red space points represents the workspace of the manipulator obtained using the Monte Carlo method, the blue cube C1 represents the circumscribed cube of W, and the orange cube C2 represents the circumscribed cube with a certain redundancy.

The six-DOF manipulator of the anchor drilling robot analyzed in this study features joint motion ranges of (-π/6, 0), (-π/6, π/6), (0, 1200), (-π/2, π/2), (-π/2, π/2), and (0, 1500) for its respective joints. The simulation results show that the external bounding rectangle of the workspace, represented by the blue cube C1, spans the coordinate ranges [−896.1, 4198.5], [−2895.1, 1667.8], and [−2238.4, 3011.2] along the respective axes. As previously mentioned，C2 is derived from C1 by introducing a certain redundancy, following the principle that the boundaries of C2 should be adjusted to the nearest values, farther from the coordinate origin, that are divisible by the segmentation step size e. Consequently, the minimum and maximum values of the circumscribed cube C2 within the workspace of the manipulator along the X, Y, and Z axes were [–1000, 4500], [–3000, 2000], and [–2500, 3500], respectively. In addition, the cyan lines delineate cube C2 with a fixed step size e = 100 mm. The black cube is the smallest cube obtained after the segmentation of C2. Evidently, when the step size was small, the image could not be clearly observed because of its high density. Hence, we set e = 500 mm (as shown in Fig. 3), which allowed N = 1320 cubes with a side length of 500 mm to be obtained via the calculation. Fig. 3(b) shows the segmentation of Fig. 3(a) in the XOY plane, and Fig. 3(c) shows an enlarged view of the black cube in Fig. 3(a).

To solve the inverse kinematics corresponding to the specified spatial position and attitude of the manipulator end-effector, the relevant small cube was first identified based on the established mapping relationship between the spatial coordinates and small cubes. Subsequently, using the error function derived in this study, the spatial points within the identified small cubes were systematically analyzed. The value of the error function was calculated based on the transformation vectors corresponding to each spatial point. A zero-error function indicates that the corresponding joint variable values of the spatial point represent a valid set of inverse solutions aligned with the desired position and orientation of the manipulator. Theoretically, as the number of workspace points approaches infinity, certain spatial points exist that satisfy a zero-error function, indicating the existence of a set of inverse solutions for the manipulator. However, in practical applications, as the number of points within the workspace increases, the task of solving the workspace and traversing spatial points within a small cube becomes computationally intensive and time consuming. Consequently, locating a spatial point within a small cube using a zero-error function is extremely challenging. Hence, in this study, a method is employed in which a spatial point is searched within a small cube that minimizes the error function value. Subsequently, the improved Newton–Raphson method is employed to further refine the solution.



(a) (b) (c)

Fig. 3. Workspace segmentation. (a) Workspace of manipulator and circumscribed cubes. (b) Projection of manipulator workspace in XY direction. (c) Divided cubes and their internal space points.

C. Dynamic adjustment of step coefficients

The Newton–Raphson method is an iterative numerical technique employed for solving equations. It leverages the Taylor expansion of functions to approximate the roots of functions. In this method, the function $f(x)$ is expressed as a Taylor series of order n in the vicinity of a reference point x_0, as follows.

$$f(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\ldots +\frac{f^n(x_0)}{n!}(x-x_0)^n+R_n(x) \quad ,(16)$$

where $R_n(x)$ represents the Taylor remainder of order n.

Consider the equation $f(x)=0$, the initial two terms of the Taylor polynomial can be used as an approximation. Thus, the first-order Taylor polynomial can be expressed as

$$f(x)=f(x_0)+f'(x_0)(x-x_0)=0, \tag{17}$$

Subsequently, the iterative function was derived as follows:

$$\begin{aligned} X_n &= X_{n-1}-f(X_{n-1})/f'(X_{n-1}) \\ &= X_{n-1}-f(X_{n-1})*f'(X_{n-1})^{-1}. \end{aligned} \tag{18}$$

where X denotes an m-dimensional column vector and $f'(X)$ signifies the set of partial derivatives of the multivariate function $f(X_{n-1})$ with respect to all its variables.

Beginning with the initial value, the iteration process is executed based on (18) as described earlier. The iteration continues until the change in the function value is below a predefined threshold, at which point the cycle terminates, yielding X as a solution to $f(X)=0$ .Therefore, this methodology is applicable to resolving the inverse kinematics of multi-DOF manipulators.

In the context of the six-DOF manipulator, the variable vector is denoted as $\theta=[\theta_1,\theta_2,d_3,\theta_4,\theta_5,d_6]$. Subsequently, the function is formulated as follows:

$$F_\theta=T_\theta-T_{desire}=\begin{bmatrix} f_1 & f_2 & f_3 & f_4 \\ f_5 & f_6 & f_7 & f_8 \\ f_9 & f_{10} & f_{11} & f_{12} \\ 0 & 0 & 0 & 0 \end{bmatrix}=0 \tag{19}$$

where F_θ represents the constructed functional equation that appears as a 4×4 matrix. The final row of this matrix comprises zeros only. The individual elements, denoted as f_i, are functions that depend on the vector $\theta=[\theta_1,\theta_2,d_3,\theta_4,\theta_5,d_6]$. Furthermore, T_θ and T_{desire} represent the homogeneous transformation matrices corresponding to the current and desired spatial positions and attitudes, respectively. These matrices can be expressed as a set of equations, as follows:

$$\begin{cases} F_\theta(1,1)=f_1(\theta)=T_\theta(1,1)-T_{desire}(1,1)=0 \\ F_\theta(1,2)=f_2(\theta)=T_\theta(1,2)-T_{desire}(1,2)=0 \\ F_\theta(1,3)=f_3(\theta)=T_\theta(1,3)-T_{desire}(1,3)=0 \\ F_\theta(1,4)=f_4(\theta)=T_\theta(1,4)-T_{desire}(1,4)=0 \\ F_\theta(2,1)=f_5(\theta)=T_\theta(2,1)-T_{desire}(2,1)=0 \\ F_\theta(2,2)=f_6(\theta)=T_\theta(2,2)-T_{desire}(2,2)=0 \\ F_\theta(2,3)=f_7(\theta)=T_\theta(2,3)-T_{desire}(2,3)=0 \\ F_\theta(2,4)=f_8(\theta)=T_\theta(2,4)-T_{desire}(2,4)=0 \\ F_\theta(3,1)=f_9(\theta)=T_\theta(3,1)-T_{desire}(3,1)=0 \\ F_\theta(3,2)=f_{10}(\theta)=T_\theta(3,2)-T_{desire}(3,2)=0 \\ F_\theta(3,3)=f_{11}(\theta)=T_\theta(3,3)-T_{desire}(3,3)=0 \\ F_\theta(3,4)=f_{12}(\theta)=T_\theta(3,4)-T_{desire}(3,4)=0 \end{cases}. \tag{20}$$

The partial derivative of F_θ, also referred to as the Jacobian matrix $J(\theta)$, is represented as follows:

$$J(\theta)=\frac{\partial f_i(\theta)}{\partial \theta_i} \quad .(21)$$

Hence, the iterative formula for the joint variable θ is expressed as follows:

$$\theta^{i+1}=\theta^i-J^{-1}f(\theta^i). \tag{22}$$

Because the Jacobian matrix is not always square and cannot be directly inverted, it is typically substituted with a pseudo-inverse matrix, which is denoted as J^+. Consequently, the equation above can be expressed as follows:

$$\begin{aligned}\theta^{i+1} &= \theta^i - \Delta\theta^i = \theta^i - J^+ f\left(\theta^i\right)\\ &= \theta^i - \left(J^T J\right)^{-1} J^T f\left(\theta^i\right)\end{aligned}, \tag{23}$$

where $\Delta\theta^i$ represents the iteration step that significantly affects the solution. When the iteration step size was extremely small, each iterative step contributed only slightly to the solution, resulting in a protracted convergence process. Conversely, if the iteration step size is excessively large, an overshoot that potentially surpasses the solution point or optimal solution can occur, resulting in algorithmic instability. In such cases, iterations may diverge instead of converging, thereby introducing oscillations or numerical instabilities.

To enhance the convergence rate of this method, an iterative step coefficient, denoted σ, is introduced, which is dynamically adjusted through a comparative analysis of the step sizes between consecutive iterations. In $\Delta\theta^{i+1} \le \Delta\theta^i$, configuring the step coefficient σ with a value exceeding 1 augments the iteration step, thus reducing the overall iteration time. Conversely, when $\Delta\theta^{i+1} \ge \Delta\theta^i$ is used, setting σ to a value less than one effectively diminishes the iteration step, thus mitigating the potential for abrupt oscillations or numerical instability that can occur when iterations exceed the optimal solution boundary. Empirical evidence derived from a series of experiments shows that in certain scenarios, setting σ to 1.6 improves precision and reduces the computational time. Conversely, in instances where $\Delta\theta^{i+1} \ge \Delta\theta^i$, a value of 0.8 for σ yields superior accuracy and minimizes time consumption.

In this study, a set of joint variables corresponding to the aforementioned minimum error function is selected as the initial value and incorporated into an improved iterative solution process. The iteration terminates when $\Delta\theta^i \le \varepsilon_{threshold}$ is applied, which is a predefined threshold. At this point, a set of joint variables satisfying (19) can be obtained. These values correspond to the spatial position and attitude of the given end-effector, thus completing the inverse kinematics solution for the unconventional multi-DOF manipulator that does not adhere to the Pieper principle.

IV. Simulation Results

Three sets of simulations are conducted. The computer CPU used is Intel (R) Core (TM) i9-13900H, the memory is 16.0 GB, and the simulation program is written in Matlab2020a.

In the first experiment, the results obtained by the proposed method were compared with those obtained by the conventional Newton–Raphson method and the QPSO method to verify the effectiveness and accuracy of the proposed method. In the second and third experiments, the effects of the iterative OIV and improved method involving adjustments to the parameters on the calculation results were investigated.

A. Error of joint variable value

The homogeneous transformation matrix corresponding to the desired target point D in space is represented as

$$T_d = \begin{bmatrix} 0.2803 & 0.6124 & -0.7392 & 992.8519 \\ -0.7392 & -0.3536 & -0.5732 & -1973.2233 \\ -0.6124 & 0.7071 & 0.3536 & 919.2388 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

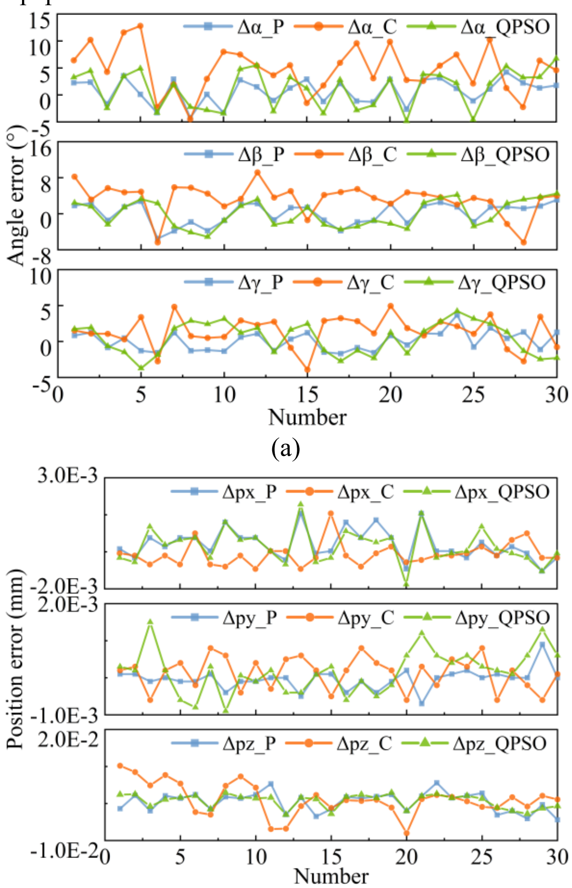
The corresponding values of the joint variables are the $\theta_d = \left[-\pi/6, 0, 1000, \pi/4, \pi/3, 1200\right]$, which is a given set of joint variables. The homogeneous transformation matrix is processed using aforementioned method, and its corresponding transformation vector is obtained as: $M_d = \left[0.8, -0.0519, 0.0396, -0.3019, 992.8519, -1973.2233, 919.2388\right]$.

Employing the proposed methodology, space point C within the vicinity of the target point was identified using (14) to minimize the error function, where the number of cycles Q is 300,000. Subsequently, the joint variable value $\theta_C = \left[\theta_{C1}, \theta_{C2}, d_{C3}, \theta_{C4}, \theta_{C5}, d_{C6}\right]$ corresponding to this space point is adopted as the initial value for the improved Newton–Raphson method. A set of inverse solutions, denoted as $\theta_S = \left[\theta_{S1}, \theta_{S2}, d_{S3}, \theta_{S4}, \theta_{S5}, d_{S6}\right]$, was obtained for the six-DOF manipulator, and the associated transformation vector $M_s = [\omega_s, x_s, y_s, z_s, p_{xs}, p_{ys}, p_{zs}]$ was computed.

Therefore, Δp_x, Δp_y, and Δp_z represent the position errors. In order to visually and intuitively represent the error in rotation angles, the obtained quaternion is converted into Euler angles of rotation around the ZYX axes. Given that experimental data does not contain rotations about the Y-axis with angles of ±90 degrees, this approach circumvents the issue of gimbal lock. Namely, $\Delta\alpha$, $\Delta\beta$, and $\Delta\gamma$ represent the angle errors. The proposed method, the conventional Newton–Raphson method and QPSO method were used to solve the inverse kinematics of the manipulator, and the results of the position and angle errors are shown in Fig. 4. In addition, 300 particles are defined for the QPSO method. It should be clarified that, in Fig. 4, the notations '_P,' '_C,' and '_QPSO' represent the inverse kinematics errors of the general manipulator calculated using the proposed method, the conventional Newton-Raphson method, and the QPSO method, respectively.

As shown in Fig. 4a, 30 simulations were conducted to evaluate the performance of the proposed method. The rotation angle errors associated with the proposed method fluctuated within the ranges of (–4.2205°, 4.2357°), (–5.5114°, 3.0920°), and (–1.6886°, 3.6833°). The corresponding average values of these errors were approximately 2.1147°, 2.1426°, and 1.1800°, respectively. In contrast, the rotation angle errors resulting from the conventional Newton–Raphson method fluctuated within the ranges (–4.5004°, 12.8009°), (–6.3727°, 9.1724°), and (–3.9209°, 4.9458°), with average values of approximately 5.4610°, 4.3649°, and 2.1709°, respectively. Simultaneously, the rotation angle errors resulting from the QPSO method fluctuated within the ranges of (–4.8426°, 6.7590°), (–5.1209°, 4.4920°), and (–3.7286°, 4.2246°), with average values of approximately 3.4564°, 2.7195°, and 2.0757°, respectively.

Evidently, the proposed method consistently outperformed the conventional Newton–Raphson and QPSO methods, and yielded significantly smaller angle errors. In particular, the proposed method yielded lower rotation angles of 61.27%, 50.91%, and 45.65% for the X, Y, and Z axes, respectively, compared with the conventional method. Compared with the QPSO method, the angular errors along the X, Y, and Z axes were reduced by 38.82%, 21.21%, and 43.15%, respectively. Compared with the other two methods, the method proposed in this paper has the smallest error fluctuation.



(a)

(b)

Fig. 4. Angle and position errors of the proposed method, the conventional Newton–Raphson method, and the QPSO method for inverse kinematics. (a) Angle error of the three methods for inverse kinematics. (b) Position error of the three methods for inverse kinematics.

As shown in Fig. 4b, the blue, orange, and green dotted lines represent the position errors corresponding to the proposed, conventional Newton–Raphson, and QPSO methods, respectively. The average positioning errors of the three methods were 0.0010, 0.0014, and 0.0010 mm. The average errors of the methods proposed in this study are comparable to those of the QPSO method and marginally smaller than those of the conventional Newton–Raphson method. Generally, the position errors for all the three methods are relatively small, approximately 0.001 mm, which meets the requirements of motion planning and control of the anchor drilling robot.

The computational times and iteration count for the three methods were analyzed meticulously, and the results are presented in Fig. 5. The blue, orange, and green bar graphs depict the time required by the proposed, conventional, and QPSO methods, respectively, to compute the inverse solution for the manipulator. A clear trend was observed, indicating that the computational process of the proposed method was generally more efficient than those of the conventional and QPSO approaches. The proposed method exhibited a wide range of computation times, with the shortest, longest, and average processing times of 291.71, 698.11, and 485.56 ms, respectively. Conversely, the computation time for the conventional method varied within the range of (223.94, 925.67), with an average processing time of 588.72 ms. The computation time for the QPSO method varied within the range of (180.95, 763.88), with an average processing time of 509.92 ms. The proposed method reduced the average computation time by 17.99% and 4.78% compared with the other two methods, respectively, which represents a significant enhancement in terms of efficiency. The accompanying blue, orange, and green lines illustrate the number of iterations required for the proposed, conventional, and QPSO methods, respectively. The proposed, conventional, and QPSO methods presented average iteration counts of 1798, 2248, and 1690, respectively. Essentially, the number of iterations using the proposed method was reduced by 20.01% compared to the conventional method, whereas the number of iterations using the QPSO method was reduced by 6.39% compared to the proposed method.

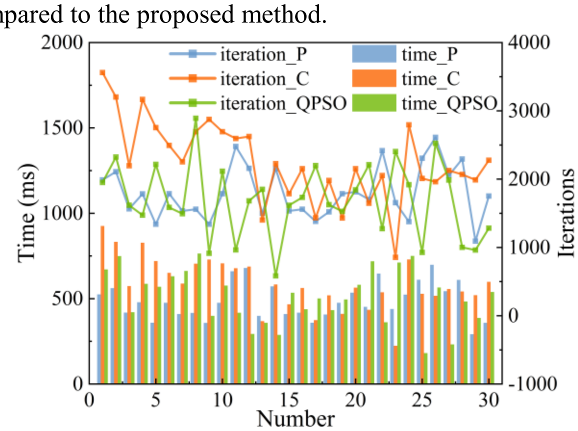


Fig. 5: Computational time and iteration counts of the proposed method, conventional Newton–Raphson method and QPSO method for inverse kinematics.

Based on a comprehensive analysis of the simulation results, the proposed method demonstrates a reduction in rotation angle error when compared to both the conventional and QPSO methods. However, the position error results are comparable across all methods. Notably, the proposed method exhibits the shortest computational time, indicating the highest computational efficiency. Overall, the method proposed in this study outperforms the other two methods in terms of both calculation accuracy and computational efficiency.

In order to verify the effectiveness and robustness of the inverse kinematics solution method of general manipulator described in this paper, NJ-220 robot is taken as the research object, and its inverse solution is obtained by using the method described in this paper. NJ-220 is a six-DOF articulated manipulator, and the axes of the three axes at the end do not intersect, namely it does not satisfy the Pieper criterion. Therefore, as a general manipulator, there is no closed solution.

Using the proposed method, the conventional Newton–Raphson method, and the QPSO method, the simulation results are presented in Fig. 6. The coordinate error shown in Fig. 6 reflects the joint angles computed by each algorithm. The corresponding coordinates (p_x, p_y, p_z) are compared to the target coordinates (p'_x, p'_y, p'_z). The distance error e_d is calculated as $e_d = \sqrt{(p_x - p'_x)^2 + (p_y - p'_y)^2 + (p_z - p'_z)^2}$. The angle error e_a represents the difference between the optimal angle (α, β, γ) obtained by the algorithms and the exact solution $(\alpha', \beta', \gamma')$. Specifically, $e_a = |\alpha - \alpha'| + |\beta - \beta'| + |\gamma - \gamma'|/3$ is defined as the mean of the total absolute differences between the corresponding components of these solutions, serving as a metric to evaluate the quality of the obtained solution.

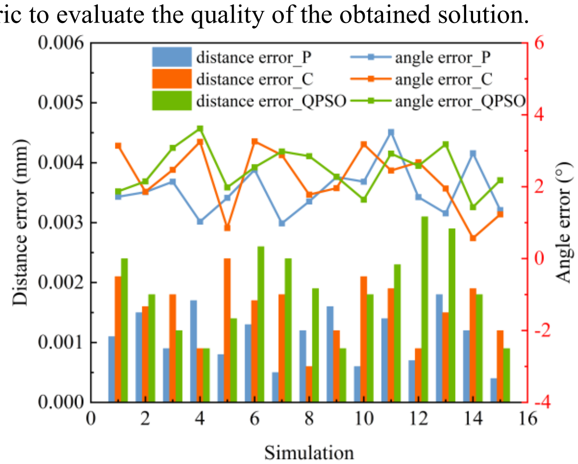


Fig. 6. Comparison of the proposed, conventional Newton–Raphson, and QPSO method simulation result with NJ-220 robot.

Fig. 6 shows that the distance error of the proposed method fluctuates within the range of 0.0004 mm to 0.0018 mm across multiple simulation experiments, with an average error of 0.0011 mm. By contrast, the distance errors of the conventional Newton–Raphson method and the QPSO method range from 0.0006 mm to 0.0024 mm and 0.0009 mm to 0.0031 mm, with corresponding average errors of 0.0016 mm and 0.0019 mm, respectively. All three methods exhibit notably small distance errors, with variations on the order of 0.001 mm, highlighting the high precision of the solutions obtained.

For angle errors, the proposed method fluctuates between 0.98° and 3.52°, with an average error of 1.91°. In comparison, the angle errors of the conventional Newton–Raphson method and the QPSO method range from 0.57° to 3.26° and 1.43° to 3.62°, with average errors of 2.32° and 2.48°, respectively. The proposed method achieves a reduction in angle error of 17.67% and 22.98% compared to the conventional Newton–Raphson and QPSO methods, respectively. These simulation results confirm that the proposed method offers a significant improvement in angle error reduction, while maintaining comparably small distance errors across all three methods.

Therefore, the aforementioned simulation results indicate that the inverse kinematics solution method for the anchor drilling robot described in this paper has significant advantages in terms of accuracy and real-time performance compared to the conventional Newton–Raphson method and the QPSO method. For general robot with different structures, the inverse kinematics solution method presented in this study can maintain a certain level of precision and speed. Additionally, the proposed method also exhibits a notable level of robustness.

B. Effect of initial value

The conventional Newton–Raphson method is particularly sensitive to the initial value, which is randomly given. An appropriate initial value can significantly enhance the accuracy and computational efficiency of a solution. To assess the effect of the OIV derived via the error function minimization method presented herein on the computational outcomes, a total of 30 simulation experiments were conducted, and the value of the number of cycles was 300,000. The iterations were stopped when the change in the joint variables was less than a threshold of 0.0001. A comparative analysis of the inverse solutions for the manipulators was performed by comparing the computational results of OIV and random initial value (RIV). The experimental results are presented in Fig. 7.

In Fig. 7(a), the angle errors are graphically depicted, with the orange line representing those yielded by RIV (the conventional Newton–Raphson method) and the blue line representing those yielded by the optimal the iterative method, which utilizes the OIV and fixed step coefficient. The overall trend is that the blue line consistently exhibited smaller angle errors compared with the orange line. The angle errors denoted by $\Delta\alpha\_OIV$, $\Delta\beta\_OIV$ and $\Delta\gamma\_OIV$ pertain to the rotations about the X, Y, and Z axes, respectively. Their values fluctuated within the ranges of (–4.1902°, 6.9628°), (–5.4840°, 6.5233°), and (–2.8398°, 2.9821°), respectively, with corresponding average values of 2.6347°, 2.3684°, and 1.3197°, respectively. Meanwhile, $\Delta\alpha\_RIV$, $\Delta\beta\_RIV$ and $\Delta\gamma\_RIV$ represent angle errors for rotations about the X, Y, and Z axes, respectively, yielded by the conventional Newton–Raphson method, and the results were consistent with those presented in Section IV. A comparison of the results obtained using the two methods showed significant reduction (by 51.75%, 45.74%, and 39.21% on the X, Y, and Z axes, respectively).

Fig. 7(b) displays the position errors associated with both methods. The position errors for both methods were exceedingly small and negligible. Corresponding to the method with the OIV in this paper, the average position errors along X, Y, and Z axes are 0.006, 0.002 and 0.0049 mm respectively. The average position errors of the RIV method are 0.006, 0.003 and 0.0034 mm respectively. It can be observed that compared with the method based on the RIV, the method based on the OIV proposed in this study can effectively reduce the angular error of the solution.

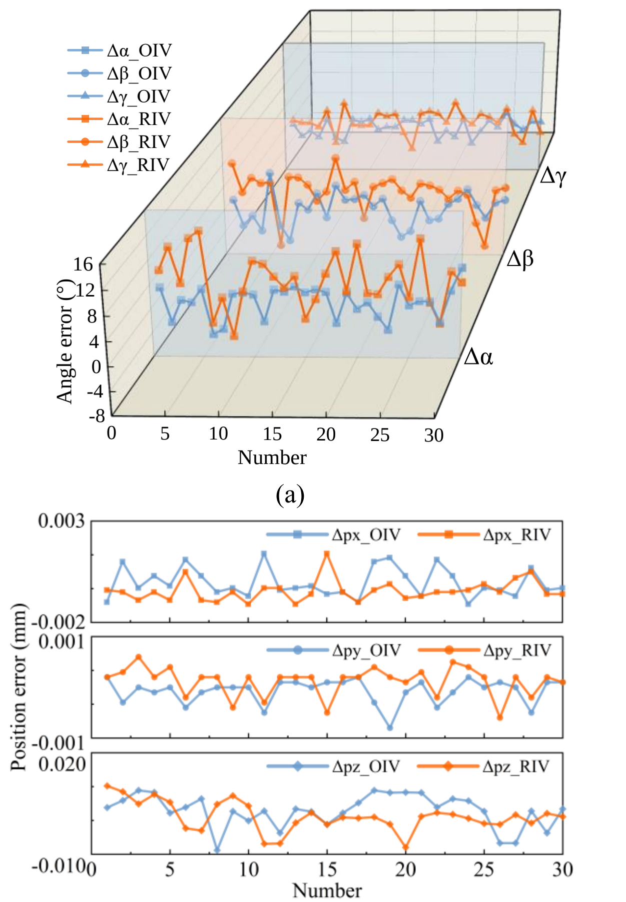


Fig. 7. Angle and position errors of the OIV and RIV for inverse kinematics. (a) Angle errors. (b) Position errors.

Similarly, the consumption time and iteration count for methods with the OIV and the RIV were computed, and the results are shown in Fig. 8. The blue and orange bar graphs represent the time consumed by the methods with the OIV and the RIV, respectively, which clearly show that the method with the OIV required a shorter processing time. Based on the results of 30 sets of simulation experiments, the shortest, longest, and average processing time were 240.48, 818.33, and 510.59 ms, respectively. This average value was significantly lower, corresponding to a reduction of 81.46 ms or 13.76%, compared with the corresponding value obtained by the method with the RIV. The line chart in Fig. 8 illustrates the iteration counts for both methods. The iteration count for the method with the OIV fluctuated within (925, 3157), and the average count was 1964. This represents a significant reduction of 294 iterations, or 13.02%, compared to the results obtained by the method with RIV.

The experimental findings presented in this section highlight the effectiveness of employing the joint variable derived from the minimum error function as the initial value for the iterative process. This approach significantly reduced the angle errors in the calculation results. Furthermore, it reduced the computation time by 81.46 ms or 13.76%, and reduced the number of iterations by 294 or 13.02%, thus improving the computational efficiency.

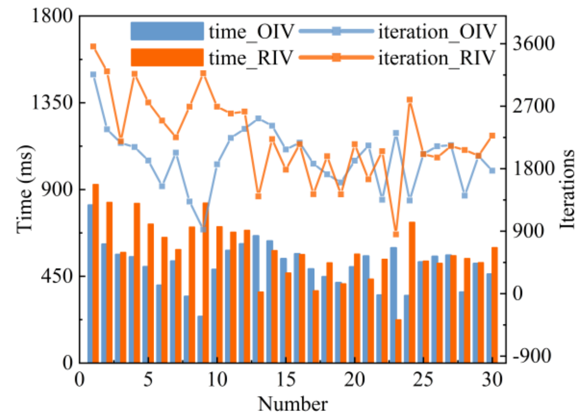


Fig. 8. Computational time and iteration count of methods with the OIV and RIV for inverse kinematics.

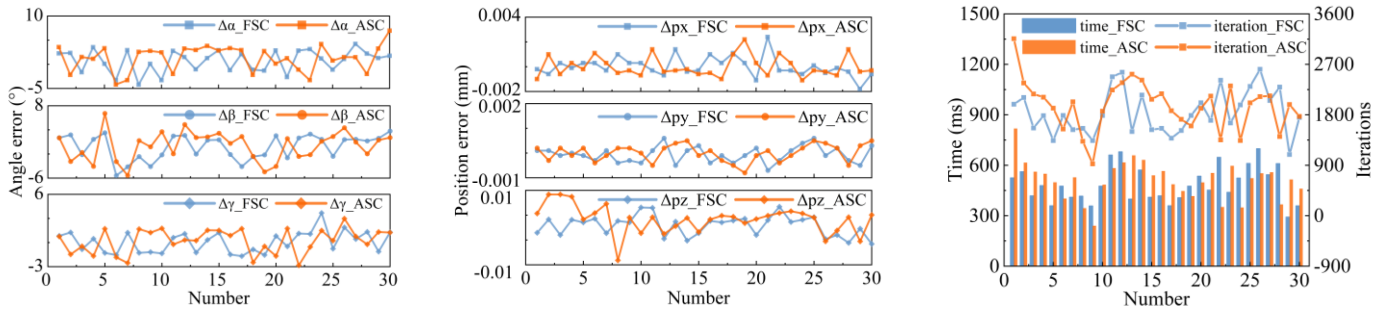
C. Effect of iterative step coefficient

Thirty sets of simulation experiments were conducted to assess the effects of the step coefficient adjustment method described in the solution. The number of cycles was 300,000. The iterations were stopped when the change in the joint variables was less than a threshold of 0.0001. Using the OIV as the foundation, a comparative analysis was conducted between the method employing the adjusted step coefficients (ASC) and that employing the fixed step coefficients (FSC). The results of this comparative study are shown in Fig. 9.

In Fig. 9(a), the green and blue lines represent the angle errors associated with the ASC and the FSC, respectively. The angle errors corresponding to the green lines were consistently smaller than those associated with the blue lines. The angle errors depicted by the green lines, fluctuated within the ranges of (–4.2205°, 4.2357°), (–5.5114°, 3.0920°), and (–1.6886°, 3.6833°), with average values of 2.1147°, 2.1426° and 1.1800°, respectively. Compared with the angle errors obtained using the FSC method, the average angle error was reduced by 0.5200°, 0.2258°, and 0.1397°, corresponding to reductions of 19.74%, 9.53%, and 10.59%, respectively. These findings underscore the tangible improvements in angle error achieved by employing the ASC method.

In Fig. 9(b), the position errors associated with both methods are presented, with the errors approaching negligible values, that is, approximately zero. More specifically, the average position errors of the method with FSC along X, Y, and Z axes are 0.0007, 0,0003 and 0.0036 mm, respectively, which are very close to the results of the method with the ASC. The values of the position errors are very small, almost negligible.

Fig. 9(c) provides insights into the time consumption and iteration counts for methods with FSC and ASC. The time consumed by the method with adjusted step coefficient fluctuated within the range of (291.71ms, 698.11ms), with an average time consumption of approximately 485.56ms. Conversely, the time consumed by the method with fixed step coefficient was higher, with an average value of approximately 510.59 ms. Time consumption is intricately related to the number of iterations. The coefficient adjustment method indicated a significant reduction in the average iteration count, that is, a decrease of 126 iterations (6.42%) compared with that yielded by the fixed coefficient method. These findings highlight the tangible efficiency gains achieved by adopting the coefficient adjustment method.



(a) (b) (c)

Fig. 9. Influence of iterative step coefficient. (a) Influence on angle error. (b) Influence on position error. (c) Influence on time consumption and iteration counts.

V. Experiment Results

The experiments were conducted in the anchor drilling robot established in the laboratory environment as shown in Fig. 10. The inclination sensors and displacement sensors were set to each joint respectively, which can accurately measure the values of the rotation angles and displacement distances of each joint. When the end-effector of the manipulator reached any position in space, the data from each sensor were collected and processed to obtain a set of joint variable values $\theta_s=[\theta_{s1},\theta_{s2},d_{s3},\theta_{s4},\theta_{s5},d_{s6}]$ corresponding to this state. The transformation matrix T_S and the transformation vector $M_s=[\omega_s,x_s,y_s,z_s,p_{xs},p_{ys},p_{zs}]$ were calculated. Subsequently, we employed the novel technique proposed in this study to compute the inverse kinematics solution for the transformation matrix T_S, thereby obtaining the joint variable values denoted as $\theta_c=[\theta_{c1},\theta_{c2},d_{c3},\theta_{c4},\theta_{c5},d_{c6}]$. Similarly, the corresponding transformation matrix T_c and transformation vector $M_c=[\omega_c,x_c,y_c,z_c,p_{xc},p_{yc},p_{zc}]$ can be obtained. Based on the transformation vectors M_s and M_c, the corresponding rotation angles α_s, β_s, γ_s, α_c, β_c, γ_c and displacement distances p_{xs}, p_{ys}, p_{zs}, p_{xc}, p_{yc}, p_{zc} can be calculated. As a result, the distance error e_d and the angle error e_a can be calculated, which can be used to represent the accuracy of the calculation results.

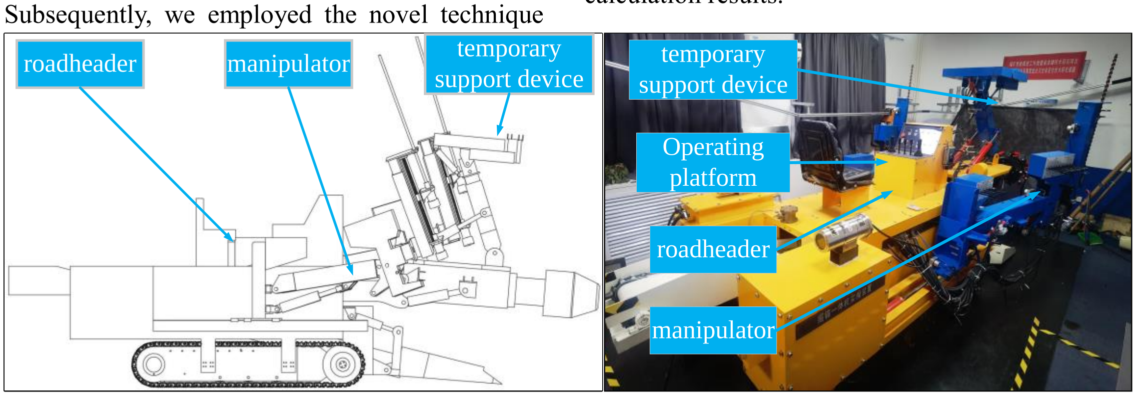


Fig. 10. Six-DOF manipulator structure of anchor drilling robot.

The joint states of the manipulator are adjusted to achieve different spatial postures of the end-effector, with the experiment repeated 20 times. Sensor measurements are used to accurately record the variable values of each joint. The data are then processed using the method proposed in this paper, as well as conventional Newton–Raphson and QPSO methods, to compute the corresponding angle error e_a and distance error e_d errors for each dataset. Additionally, the computational time for each process is recorded. The experimental results are shown in Fig. 11.

As shown in Fig. 11(a), the angle and distance errors associated with the proposed method are generally smaller than those of the other two methods. Additionally, the distance error distribution for the proposed method is more concentrated, indicating better stability of the results. It is important to note that during the operation of the manipulator of the anchor drilling robot, the threshold for angle error is set at 5°. The results obtained using the proposed method all meet this criterion, while the conventional method yielded results exceeding 5°, further demonstrating the effectiveness of the proposed approach.

Fig. 11(b) presents the time consumption for the three methods. In general, the time required for each method ranges from (271.35 ms, 678.85 ms), (205.59 ms, 863.59 ms), and (218.52 ms, 976.24 ms), with average values of 426.89 ms, 494.84 ms, and 557.94 ms, respectively. Compared to the other two methods, the proposed approach reduces the time consumption by 13.73% and 23.49%, respectively, indicating that the proposed method achieves faster convergence when solving the inverse kinematics of a general-purpose manipulator.

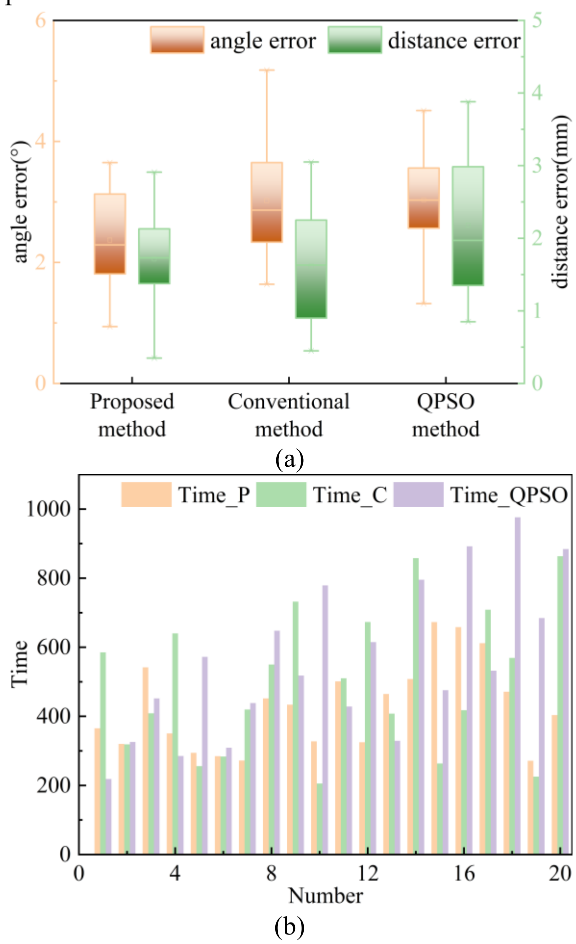


Fig. 11. Results of the experiments based on the proposed method, the conventional Newton–Raphson and QPSO methods. (a) the angle and distance error of the three methods. (b) the time consumption of the three methods.

VI. Conclusion

To solve the inverse kinematics of the general manipulator rapidly and with high precision, a novel approach based on workspace analysis and iterative step coefficient adjustment was proposed. The novelty of this method lies in its ability to identify the joint variable values corresponding to the minimum error function within the workspace of a general manipulator using the Monte Carlo method. Additionally, a iterative step coefficient adjustment was introduced to expedite and enhance the convergence of the iterative process toward the desired error threshold. The simulation results revealed substantial improvements over the conventional Newton–Raphson and QPSO methods. Specifically, the angle errors about the X, Y, and Z axes were significantly reduced by 61.27%, 50.91%, and 45.65%, respectively, compared to the conventional method. The angle errors were reduced by 38.82%, 21.21%, and 43.15%, respectively, compared to the QPOS method. The position errors essentially remained within a narrow range centered around zero. Moreover, the computation time decreased by 17.99% and 4.78% compared with the two methods, respectively, and the average number of iterations decreased by 20.01% compared with the conventional method. This approach offers several distinct advantages including swift computation, high stability, and efficient convergence. Moreover, the underlying principles of the approach are applicable to other robot configurations, and the method can be extended to different robotic systems.

In future research, the authors aim to extend the proposed methodology for inverse kinematics to general robotic manipulators across various domains. This broader application will allow for a more comprehensive evaluation of the method's versatility and effectiveness. Additionally, the authors plan to investigate how to improve the robustness of the proposed approach in the presence of environmental disturbances, such as sensor noise and dynamic changes in the operating environment. Enhancing the robustness of the method under these conditions represents a key direction for future research.

References

[1] Z. Zhakypov, J. Paik, Design methodology for constructing multimaterial origami robots and machines, IEEE Trans. Robot, 34 (2018) 151-165.

[2] C. Mao, S. Li, Z. Chen, X. Zhang, C. Li, Robust kinematic calibration for improving collaboration accuracy of dual-arm manipulators with experimental validation, Measurement, 155 (2020) 107524.

[3] Y. Yin, X. Wang, D. Xu, F. Liu, Y. Wang, W. Wu, Robust visual detection–learning–tracking framework for autonomous aerial refueling of UAVs, IEEE Trans. Instrum. Meas, 65 (2016) 510-521.

[4] M. Fanni, A. Khalifa, A new 6-DOF quadrotor manipulation system: design, kinematics, dynamics, and control, IEEE/ASME Trans. Mechatronics, 22 (2017) 1315-1326.

[5] R. Mur-Artal, J.D. Tardos, Orb-slam2: an open-source slam system for monocular, stereo, and RGB-D cameras, IEEE Trans. Robot, 33 (2017) 1255-1262.

[6] F. Xu, H. Wang, W. Chen, Y. Miao, Visual servoing of a cable-driven soft robot manipulator with shape feature, IEEE Rob. Autom. Lett, 6 (2021) 4281-4288.

[7] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, D. Quillen, Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, Int. J. Robot. Res, 37 (2018) 421-436.

[8] K. Guo, Y. Pan, H. Yu, Composite learning robot control with friction compensation: a neural network-based approach, IEEE Trans. Ind. Electron, 66 (2019) 7841-7851.

[9] Q. Liu, J. Zhao, H. Zhu, G. Wang, J.D. McLennan, Review, classification and structural analysis of downhole robots: core technology and prospects for application, Robot Auton Syst, 115 (2019) 104-120.

[10] S. Zhang, S. Cheng, S. Cheng, Z. Jin, Visual measurement method and application of mobile manipulator pose estimation based on PPMCC-IMM filtering, IEEE Trans. Instrum. Meas, 72 (2023) 1-12

[11] M. Amersdorfer, J. Kappey, T. Meurer, Real-time freeform surface and path tracking for force controlled robotic tooling applications, Robot Cim-Int Manuf, 65 (2020) 101955.

[12] P. Radoglou-Grammatikis, P. Sarigiannidis, T. Lagkas, I. Moscholios, A compilation of UAV applications for precision agriculture, Comput. Netw, 172 (2020) 107148.

[13] Q. Ma, E. Kobayashi, H. Suenaga, K. Hara, J. Wang, K. Nakagawa, I.

Sakuma, K. Masamune, Autonomous surgical robot with camera-based markerless navigation for oral and maxillofacial surgery, IEEE/ASME Trans. Mechatronics, 25 (2020) 1084-1094.
[14] K. Lin, Y. Li, J. Sun, D. Zhou, Q. Zhang, Multi-sensor fusion for body sensor network in medical human–robot interaction scenario, Inform Fusion, 57 (2020) 15-26.
[15] Y. Yu, K. Zhang, L. Yang, D. Zhang, Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN, Comput Electron Agr, 163 (2019) 104846.
[16] B. Arad, J. Balendonck, R. Barth, O. Ben Shahar, Y. Edan, T. Hellström, J. Hemming, P. Kurtser, O. Ringdahl, T. Tielen, B. Tuijl, Development of a sweet pepper harvesting robot, J Field Robot, 37 (2020) 1027-1039.
[17] T. Liu, R. Jackson, D. Franson, N.L. Poirot, R.K. Criss, N. Seiberlich, M.A. Griswold, M.C. Cavusoglu, Iterative jacobian-based inverse kinematics and open-loop control of an MRI-guided magnetically actuated steerable catheter system, IEEE/ASME Trans. Mechatronics, 22 (2017) 1765-1776.
[18] I. Zaplana, L. Basanez, A novel closed-form solution for the inverse kinematics of redundant manipulators through workspace analysis, Mech Mach. Theory, 121 (2018) 829-843.
[19] Z. Wang, G. Li, L. Yao, Y. Cai, T. Lin, J. Zhang, H. Dong, Intelligent fault detection scheme for constant-speed wind turbines based on improved multiscale fuzzy entropy and adaptive chaotic Aquila optimization-based support vector machine, ISA Trans, 138 (2023) 582-602.
[20] Z. Wang, Q. Luo, H. Chen, J. Zhao, L. Yao, J. Zhang, F. Chu, A high-accuracy intelligent fault diagnosis method for aero-engine bearings with limited samples, Comput Ind, 159-160 (2024) 104099.
[21] G. Kanagaraj, S. Sheik Masthan, V.F. Yu, Meta-Heuristics Based Inverse Kinematics of Robot Manipulator's Path Tracking Capability Under Joint Limits, Mendel, 28 (2022) 41-54.
[22] J. Lu, T. Zou, X. Jiang, A neural network based approach to inverse kinematics problem for general six-axis robots, Sensors-Basel, 22 (2022) 8909.
[23] O. Lakhal, A. Melingui, R. Merzouki, Hybrid approach for modeling and solving of kinematics of a compact bionic handling assistant manipulator, IEEE/ASME Trans. Mechatronics, 21 (2016) 1326-1335.
[24] H. Wu, X. Zhang, L. Song, C. Su, L. Gu, A hybrid improved BRO algorithm and its application in inverse kinematics of 7R 6DOF robot, Adv Mech Eng, 14 (2022) 168781322210851.
[25] X. Wang, X. Liu, L. Chen, H. Hu, Deep-learning damped least squares method for inverse kinematics of redundant robots, Measurement, 171 (2021) 108821.
[26] A.H.B. Boutchouang, A. Melingui, J.J.M. Ahanda, O. Lakhal, F.B. Motto, R. Merzouki, Learning-based approach to Inverse kinematics of wheeled mobile continuum manipulators, IEEE/ASME Trans. Mechatronics, 27 (2022) 3451-3462.
[27] J. Hernandez-Barragan, C. Lopez-Franco, N. Arana-Daniel, A.Y. Alanis, A. Lopez-Franco, A modified firefly algorithm for the inverse kinematics solutions of robotic manipulators, Integr Comput-Aid E, 28 (2021) 257-275.
[28] F. Zhao, Y. Cheng, H. Pan, Y. Cheng, X. Zhang, B. Wu, Y. Hu, Solving redundant inverse kinematics of CMOR based on chaos-driven particle swarm optimization algorithm, Fusion Eng Des, 192 (2023) 113712.
[29] O. Duymazlar, D. Engin, Boomerang algorithm based on swarm optimization for inverse kinematics of 6 DOF open chain manipulators, Turk J Electr Eng Co, 31 (2023) 342-359.
[30] S.M. S A R, G. Kanagaraj, V.F. Yu, Gravitation search-based hybrid algorithm for solving inverse kinematics of an n-link redundant manipulator, J Comput Des Eng, 10 (2023) 2019-2035.
[31] R.V. Ram, P. P, S.J. Junco, Inverse kinematics of mobile manipulator using bidirectional particle swarm optimization by manipulator decoupling, Mech Mach. Theory, 131 (2019) 385-405.
[32] H. Deng, C. Xie, An improved particle swarm optimization algorithm for inverse kinematics solution of multi-DOF serial robotic manipulators, Soft Comput, 25 (2021) 13695-13708.
[33] F. Liu, H. Huang, B. Li, F. Xi, A parallel learning particle swarm optimizer for inverse kinematics of robotic manipulator, Int J Intell Syst, 36 (2021) 6101-6132.
[34] G. Zhao, D. Jiang, X. Liu, X. Tong, Y. Sun, B. Tao, J. Kong, J. Yun, Y. Liu, Z. Fang, A tandem robotic arm inverse kinematic solution based on an improved particle swarm algorithm, Front Bioeng Biotech, 10 (2022).
[35] S. Dereli, R. Köker, A meta-heuristic proposal for inverse kinematics solution of 7-DOF serial robotic manipulator: quantum behaved particle swarm algorithm, Artif Intell Rev, 53 (2020) 949-964.
[36] S. Fan, X. Xie, X. Zhou, Optimum manipulator path generation based on improved differential evolution constrained optimization algorithm, Int J Adv Robot Syst, 16 (2019) 172988141987206.
[37] T. Nguyen, T. Bui, H. Pham, Using proposed optimization algorithm for solving inverse kinematics of human upper limb applying in rehabilitation robotic, Artif Intell Rev, 55 (2022) 679-705.
[38] G. Song, S. Su, Y. Li, X. Zhao, H. Du, J. Han, Y. Zhao, A closed-loop framework for the inverse kinematics of the 7 degrees of freedom manipulator, Robotica, 39 (2021) 572-581.
[39] X. Zhang, G. Li, F. Xiao, D. Jiang, B. Tao, J. Kong, G. Jiang, Y. Liu, An inverse kinematics framework of mobile manipulator based on unique domain constraint, Mech Mach. Theory, 183 (2023) 105273.
[40] L. Lechuga-Gutierrez, E. Macias-Garcia, G. Martínez-Terán, J. Zamora-Esquivel, E. Bayro-Corrochano, Iterative inverse kinematics for robot manipulators using quaternion algebra and conformal geometric algebra, Meccanica, 57 (2022) 1413-1428.
[41] Y. Xie, X. Wu, Z. Shi, Z. Wang, J. Sun, T. Hao, The path planning of space manipulator based on Gauss–Newton iteration method, Adv Mech Eng, 9 (2017) 168781401771847.

Declaration of interests

☒The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Title: Inverse Kinematics Solution Method Based on Workspace Analysis and Iterative Step Coefficient Adjustment of General Robot

Author information:

Mengyu Lei[a,b], Xuhui Zhang[a,b,*] , Wenjuan Yang[a,b], Guangming Zhang[b,c]

[a] *School of Mechanical Engineering, Xi'an University of Science and Technology, Xi'an 710054, PR China*

[b] *Shaanxi Key Laboratory of Intelligent Detection and Control for Mining Electromechanical Equipment, Xi'an 710054, PR China*

[c] *General Engineering Research Institute, Faculty of Engineering and Technology, Liverpool John Moores University, Liverpool L3 3AF, U.K.*

E-mail address of authors:

19105016001@stu.xust.edu.cn(M. Lei.); zhangxh@xust.edu.cn(X. Zhang.); yangwj@xust.edu.cn (W. Yang.); 20105016013@stu.xust.edu.cn(J. Wan.); g.zhang@ljmu.ac.uk(G. Zhang.)

* Corresponding author: School of Mechanical Engineering, Xi'an University of Science and Technology, Xi'an, Shaanxi, China.

E-mail address: zhangxh@xust.edu.cn(X. Zhang).