applied sciences

MDPI

Article

Existence Conditions and General Solutions of Closed-form Inverse Kinematics for Revolute Serial Robots

Wang Shanda [1], Luo Xiao [2,*], Luo Qingsheng [1] and Han Baoling [3]

[1] School of Mechatronical Engineering, Beijing Institute of Technology, Beijing 100081, China; 3120160137@bit.edu.cn (S.W.); luoqsh@bit.edu.cn (Q.L.)
[2] School of Computer Science & Technology, Beijing Institute of Technology, Beijing 100081, China
[3] School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China; hanbl@bit.edu.cn
* Correspondence: luox@bit.edu.cn; Tel.: +86-010-6891-8856

Received: 14 August 2019; Accepted: 10 October 2019; Published: 16 October 2019

Featured Application: This paper primarily studies the existence conditions for the closed-form inverse kinematic solution for revolute serial robots. Based on the results, a general closed-form inverse solution algorithm is designed.

Abstract: This study proposes a method for judging the existence of closed-form inverse kinematics solutions based on the Denavit–Hartenberg (DH) model. In this method, serial robots with closed-form solutions are described using three types of sub-problems from the viewpoint of solving algebraic equations. If a serial robot can be described using these three types of sub-problems, i.e., if the inverse kinematics problems can be solved by several basic problems, then there is a closed-form solution. Based on the above method, we design a set of universal closed-form inverse kinematics solving algorithms. Since there is a definite formula solution for the three types of sub-problems, the joint angles can be rapidly determined. In addition, because the DH parameters can directly reflect the linkage of the robot, the judgment of the sub-problems is also quick and accurate. More importantly, the algorithm can be applied to serial robots with low degrees of freedom. This enables the algorithm to not only quickly and accurately solve inverse kinematics problems but also to exhibit high universality. This proposed theory improves the existence conditions for closed-form reverse solutions and further promotes the development of motion control techniques for serial robots.

Keywords: inverse kinematics; Denavit–Hartenberg method; closed-form solution; existence condition; universal inverse kinematics algorithm

1. Introduction

Recently, the use of serial robots has increased owing to their extensive application in the field of bionics and in various industries [1–5]. Overcoming the usual problems in inverse kinematics is a key to controlling the motion of serial robots. However, inverse kinematics is a non-linear problem with multiple solutions. Among these solutions, the general numerical solution method is both time-consuming and unstable. Alternatively, the closed-form solution for inverse kinematics is commonly sought for practical applications but has two major limitations that remain unsolved. First, there is no general method for finding the closed-form solution. Second, the Pieper criterion is not complete.

To resolve these problems, a model for the robotic kinematics needs to be established. There are two major methods for robotic modeling using inverse kinematics, namely the Denavit–Hartenberg (DH) parametric method and the product of exponentials (POE) formula method.

The DH parametric modeling method is based on a transformation of coordinates [6]. Due to its easily visualized parameters, this convenient method has attracted the attention of many researchers and engineers. Under this model, the inverse kinematics problem is solved via two methods, i.e., numerical and closed-form solutions. Moreover, because the forward kinematics formula of the DH model provides a maximum of six independent equations, simplifying the algebraic equation using the numerical method can solve the inverse kinematics problem for most non-redundant robots [7]. In addition, the Jacobian matrix derived in the DH model can be used to solve the inverse kinematics problem for redundant robots [6,8–11]. Based on the numerical method, a series of objective functions can be added [12] that not only enable the robot to reach a designated position but also allow the completion of some extra tasks, such as keeping away from a singular point or maintaining its maximum motion ability. Studies on numerical solutions have primarily focused on efficiently and stably solving the equations or obtaining an approximate solution using the functional approximation method in combination with neural networks and intelligent algorithms [11,13,14]. However, owing to the existence of the robotic singular posture, the stability of the numerical solution and its convergence rate cannot be guaranteed. Therefore, this method is not suitable for cases where instantaneity is required.

Another method based on the DH model is the analytical method, which uses a closed formula to determine the relations between the terminal posture and various joint angles. Nonetheless, not all robots have a closed-form solution. In 1978, a general method for solving for the closed-form solution was proposed by Paul [15]. In 1968, Pieper studied six-degree-of-freedom (DOF) robots with three axes intersecting at one point [16] and concluded that a serial robot with the three terminal axes intersecting at one point has a closed-form solution. This criterion, i.e., that a six-DOF serial robot with three terminal axes intersecting at one point (defined as Pieper Criterion 1, let us say PC-1) or three adjacent parallel axes (defined as Pieper Criterion 2, let us say PC-2) has a closed-form solution, has been obtained in subsequent studies [17,18]. However, the method proposed by Paul needs to be manually derived and the formula needs to be rederived every time the robot structure changes, which limits the universality of the closed-form solution. Accordingly, studies on the closed-form solution based on the DH model have primarily focused on situations with already known structures, such as keeping the robot away from a singular value [19]. More importantly, the necessary conditions for the existence of the closed-form inverse solution given by Pieper are inadequate. For example, a robot with the DH parameters listed in Table 1 has three terminal axes intersecting at one point, as well as three parallel joints; however, its inverse kinematics problem cannot be solved.

Table 1. Counter-example of Pieper Criterion 1 (PC-1).

n	θ/°	d/mm	a/mm	α/°
1	θ_1	d_1	a_1	0
2	θ_2	d_2	a_2	0
3	θ_3	d_3	a_3	$\pi/2$
4	θ_4	d_4	0	$\pi/2$
5	θ_5	0	0	$-\pi/2$
6	θ_6	0	0	0

The DH method is a parameter selection norm based on a transformation of coordinates [6]. This method regulates the criterion for selecting three parameters between two adjacent axes, where a represents the linkage offset, i.e., the normal offset between the two adjacent axes, d is the linkage length, representing the axial offset between the two adjacent axes, and α is the linkage torsion, representing the included angle between the two adjacent axes.

On a similar note, the robot with the DH parameters listed in Table 2 has three adjacent and parallel joints; however, we cannot solve the inverse kinematics problem.

Table 2. Counter-example of Pieper Criterion 2 (PC-2).

n	θ/°	d/mm	a/mm	α/°
1	θ_1	d_1	0	0
2	θ_2	0	a_2	0
3	θ_3	0	a_3	$\pi/2$
4	θ_4	d_4	0	0
5	θ_5	d_5	a_5	0
6	θ_6	d_6	0	0

The inadequacy of the Pieper principle leads to defects in the inverse kinematics algorithm designed on the basis of this principle. However, no further studies on this problem have been conducted to supplement and improve this method.

Conversely, the modeling method for the POE formula has been presented in some detail [20]. The logic for the inverse kinematics solution problem based on POE primarily consists of a reduction of the original problem into several sub-problems followed by solving all the joint angles step by step. This method uses an abstract geometric model to solve the inverse kinematics problem, therefore enhancing the generalization of the algorithm. In POE-based studies, several universal inverse kinematics algorithms have been designed, of which the most typical universal sub-problem was proposed in 1986 [21,22]. There is subproblem 1. Rotation about a single axis, subproblem 2. Rotation about two subsequent axes, and subproblem 3. Rotation to a given distance. A robotic universal solving method that meets the Pieper principle has been proposed [23], and a new sub-problem was proposed to solve the universal inverse kinematics algorithm [24]. The universal inverse kinematics algorithm based on the POE model has definite geometrical significance and stable numerical values. However, in practice, it is impossible for some robots to utilize the known sub-problems to describe and solve the problem. In addition, the algorithm consumes enormous computational resources when selecting sub-problems. As a result, it is difficult for the universal inverse solution algorithm based on the POE model to be applied in cases with real-time requirements.

It can be seen that both the DH model and POE model have various drawbacks in solving inverse kinematics problems. This study proposes a method for judging the existence of closed-form solutions based on the DH model to deal with the first problem of the closed-form solution. In this method, revolute serial robots with closed-form solutions are described using three types of sub-problems from the viewpoint of solving the algebraic equations. They are sub-problem of translation components, sub-problem of rotational components, and sub-problem of sub-chains. In more detail, based on sub-problems, when the configuration of the robot satisfies some characteristics, some more concise and fixed formulas can be used to solve the joint angle. This article refers to this more specific situation as the basic problem of sub-problems, a total of ten. If a serial robot can be described using the three types of sub-problems, i.e., if the inverse motion problems can be solved by several basic problems, then there is a closed-form solution. Based on the above method, a set of universal closed-form inverse kinematics solving algorithms is designed to address the second problem mentioned above. Since there is a definite formula solution in the three types of sub-problems, the joint angles can be rapidly determined. In addition, because the DH parameters can directly reflect the linkage of the robot, the judgment of the sub-problems is also quick and accurate.

In general, the main contributions of this article are:

1. Proposes a more complete judgment condition for the existence of the closed-form solution of series robot to solve the defect of Pieper principle. In addition, this judgment method is a method that is suitable for all robots whose degree of freedom is less than six.
2. A general inverse kinematics algorithm is designed for this judgment method. We can use this algorithm to find the angles of all joints of the series robots meet the conditions quickly and accurately.

3. This universal inverse solution algorithm has a simple judgment method, the calculation speed of it is fast, and it is an algorithm that can be applied in occasions with real-time requirements. Usually, a special method is used for the motion control of series robot, only the common configuration is supported, and some parameters cannot be adjusted. Therefore, the proposed method can greatly improve the application range of the motion controller of series robots.
4. Proposed method and theory has also enriched the foundation of robotics.

The manuscript is organized as follows. Section 2 derives a kinematics equation for a robot based on the standard DH model. Relevant properties are obtained according to the forward kinematics equation to be used in the inverse kinematics problems. Section 3 analyzes the robotic inverse kinematics problems. According to the existence conditions, the inverse kinematics problems are divided into three sub-problems and 10 types of basic problems. Section 4 designs a universal inverse kinematics algorithm. Section 5 designed two experiments. The first experiment uses MATLAB and Robotics Toolbox [25] to test the completeness, versatility, and continuity of the algorithm. Subsequently, the algorithm is implemented and verified in a real-time system to demonstrate its correctness and real-time stability. Section 6 presents a summary of the findings of this paper.

2. Kinetic Models

2.1. Standard DH Parameters

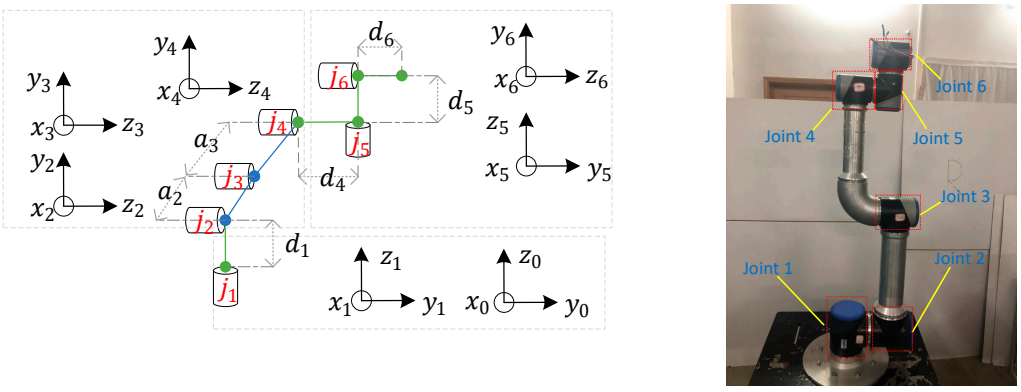
The DH method is a parameter selection norm based on a transformation of coordinates [6].

Let axis k denote the axis of the axis connecting link $k-1$ to link k; DH method adopted to define link Frame k:

Choose axis z_k along the axis of joint $k+1$. Locate the origin O_k at the intersection of axis z_k with the common normal to axes z_{k-1} and z_k. Locate also O_k, at the intersection of the common normal with axis z_{k-1}. Choose axis x_k along the common normal to axes z_{k-1} and z_k with direction from joint k to joint $k+1$. Choose axis y_k so as to complete a right-handed frame.

Once the link frames have been established, the position and orientation of Frame k with respect to Frame $k-1$ are completely specified by the following parameters: a_k distance between O_k and O_k, d_k coordinate of O_k, along z_{k-1}, α_i angle between axes z_{k-1} and z_k about axis x_k to be taken positive when rotation is made counter-clockwise, and θ_k angle between axes x_{k-1} and x_k about axis z_{k-1} to be taken positive when rotation is made counter-clockwise.

Using the DH method, the relations for a robot can be directly established in a visualized manner and the existence conditions for closed-form solutions can be derived in a more direct and visualized manner. The standard DH parameters in the case of the KingKong collaborative robot are shown in Table 3. A schematic diagram of the robot is shown in Figure 1.



(**a**) (**b**)

Figure 1. (**a**) Schematic for the KingKong parameters: d_k is the linkage length of the kth link and a_k is the linkage torsion of the kth link. j_k is the kth joint. (**b**) Visualization of the KingKong model.

Table 3. Denavit–Hartenberg (DH) parameters of the KingKong collaborative robot.

n	*θ*/°	*d*/mm	*a*/mm	*α*/°
1	θ_1	89.459	0	$\pi/2$
2	θ_2	0	−42.5	0
3	θ_3	0	−39.225	0
4	θ_4	109.15	0	$\pi/2$
5	θ_5	94.65	0	$-\pi/2$
6	θ_6	82.3	0	0

2.2. Forward Kinematics Formula

Using the following formula, the DH parameters [6] can be transformed into elements of a transformation matrix from [6]:

$$ {}^{k-1}_{k}T(\theta_i) = \begin{bmatrix} R_k & P_k \\ 0^{1\times3} & 1 \end{bmatrix} = \begin{bmatrix} cos\theta_k & -sin\theta_k cos\alpha_k & sin\theta_k sin\alpha_k & a_k cos\theta_k \\ sin\theta_k & cos\theta_k cos\alpha_k & -cos\theta_k sin\alpha_k & a_k sin\theta_k \\ 0 & sin\alpha_k & cos\alpha_k & d_k \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{1} $$

where k indicates the number of the current joint. ${}^{k-1}_{k}R$ is the rotational component of ${}^{k-1}_{k}T(\theta_k)$, a 3×3 orthogonal matrix, and ${}^{k-1}_{k}P$ is the translational component of ${}^{k-1}_{k}T(\theta_k)$. The end transformation matrix of the end executor of the robot relative to the coordinate system can be obtained using the following formula:

$$ {}^{0}_{i}T = {}^{0}_{1}T^{1}_{2}T^{2}_{3}T \cdots {}^{i-1}_{i}T. \tag{2} $$

i indicates the number of degrees of freedoms of the robot. According to Equation (2) from [18], we could obtain ${}^{0}_{i}T$ as follows:

$$ {}^{0}_{i}T = \begin{bmatrix} {}^{0}_{i}R & {}^{0}_{i}P \\ 0^{1\times3} & 1 \end{bmatrix} = \begin{bmatrix} \prod_{n=1}^{n=i} {}^{n-1}_{n}R & \sum_{n=1}^{n=i} {}^{0}_{1}R \ldots {}^{i-1}_{i}P \\ 0^{1\times3} & 1 \end{bmatrix}, \tag{3} $$

where ${}^{0}_{i}R$ is the rotational component of ${}^{0}_{i}T$. Since the rotational component ${}^{i-1}_{i}R$ is closed and orthogonal, ${}^{0}_{i}R$ is an orthogonal matrix. Here, ${}^{0}_{i}P$ is the translational component of ${}^{0}_{i}T$, which can be represented as follows:

$$ {}^{0}_{i}P = \sum_{n=1}^{n=i} {}^{0}_{1}R \ldots {}^{i-1}_{i}P = \sum_{n=1}^{n=i} P_n. \tag{4} $$

According to Equation (4), the positional component of the back linkage is influenced by the front joint angle.

In the definition of the DH parameter, the Z-direction of the joint coordinate indicates the orientation of the joint and can be used to characterize the relationship between the joints. Here, the joints of the serial robots are represented as z_k. To concisely indicate the linkage relations, the following definitions are made: $\alpha_k = 0$, then $z_k \parallel z_{k+1}$, whereas $\alpha_k = \pm\pi/2$, then $z_k \perp z_{k+1}$, where $z_k^{a,d}$ represents the linkage parameters $a_k = d_k = 0$.

2.3. Decoupling of the End Linkage

The coordinate transformation matrix is written as ${}^{0}_{i-1}T = \begin{bmatrix} {}^{0}_{i-1}R & {}^{0}_{i-1}P \\ 0^{1\times3} & 1 \end{bmatrix}$. Moreover, the transformation matrix of the last axis is ${}^{i-1}_{i}T = \begin{bmatrix} {}^{i-1}_{i}R & {}^{i-1}_{i}P \\ 0^{1\times3} & 1 \end{bmatrix}$. The rotational component R is written in the form of a row vector as $R = \begin{bmatrix} \vec{x} & \vec{y} & \vec{z} \end{bmatrix}$.

Then, ${}^{0}_{i}P$ can be written as:

$$ {}^{0}_{i}P = {}^{0}_{i-1}R \begin{bmatrix} cos\theta_i \\ sin\theta_i \\ 0 \end{bmatrix} a_i + {}^{0}_{i-1}\vec{z}\, d_i + {}^{0}_{i-1}P, \tag{5} $$

further:

$$ {}^{0}_{i-1}R \begin{bmatrix} cos\theta_i \\ sin\theta_i \\ 0 \end{bmatrix} = {}^{0}_{i}\vec{x}. \tag{6} $$

The following conclusions can be drawn:

when $\alpha_i = \pi/2$ then ${}^{0}_{i}\vec{y} = {}^{0}_{i-1}\vec{z}$,

$$ {}^{0}_{i-1}P = {}^{0}_{i}P - {}^{0}_{i}\vec{x}\, a_i - {}^{0}_{i}\vec{y}\, d_i; \tag{7} $$

when $\alpha_i = -\pi/2$ then ${}^{0}_{i}\vec{y} = -{}^{0}_{i-1}\vec{z}$,

$$ {}^{0}_{i-1}P = {}^{0}_{i}P - {}^{0}_{i}\vec{x}\, a_i + {}^{0}_{i}\vec{y}\, d_i; \tag{8} $$

and when $\alpha_i = 0$ then, ${}^{0}_{i}\vec{z} = {}^{0}_{i-1}\vec{z}$,

$$ {}^{0}_{i-1}P = {}^{0}_{i}P - {}^{0}_{i}\vec{x}\, a_i - {}^{0}_{i}\vec{z}\, d_i. \tag{9} $$

The above derivation demonstrates that when the current end transformation matrix ${}^{0}_{i}T$ is known, the translational component ${}^{0}_{i-1}P$ in the previous transformational matrix ${}^{0}_{i-1}T$ can be obtained based on the rotational component ${}^{0}_{i}R$ of ${}^{0}_{i}T$ and the translational component ${}^{0}_{i}P$. Given such a property, the last axis may not be considered when the translational component of the robot is being analyzed.

2.4. Translational Relation of the Rotational Component

According to Equation (3) form [6], ${}^{k-1}_{k}R = \mathrm{Rotz}(\theta_k)\mathrm{Rotx}(\alpha_k)$. The rotational component of the transformational matrix can be expressed as:

$$ {}^{0}_{k}R = \prod_{n=1}^{n=k} {}^{n-1}_{n}R = \prod_{n=1}^{n=k} \mathrm{Rotz}(\theta_n)\mathrm{Rotx}(\alpha_n). \tag{10} $$

If there exists several continuous parallel joints in the robot, then $z_l \parallel z_{l+1} \ldots \parallel z_m$ and $\alpha_l = \alpha_{l+1} \ldots = \alpha_{m-1} = 0$. In addition, the rotational transformation in the same direction has additivity:

$$ {}^{l}_{m}R = \prod_{n=l}^{n=m} \mathrm{Rotz}(\theta_n)\mathrm{Rotx}(\alpha_n) = \mathrm{Rotz}(\Theta)\mathrm{Rotx}(\alpha_m) = R_{\theta_s}, \tag{11} $$

where $\Theta = \theta_l + \theta_{l+1} \ldots + \theta_m$. Here, we refer to $z_l \parallel z_{l+1} \ldots \parallel z_m$ as a sub-chain s. This concept will be used in the following paragraphs. According to Equation (11), the translational components of the sub-chain $z_1 \parallel z_2 \ldots \parallel z_m$ can be defined and simplified as follows:

$$ {}^{m-1}_{m}P = R_{1.m-1}P_m = Rotz(\Theta) \begin{bmatrix} a_m \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ d_m \end{bmatrix} = \begin{bmatrix} a_m \cos\ \Theta \\ a_m \sin\ \Theta \\ d_m \end{bmatrix} = P_{\theta_s}, \tag{12} $$

where $\Theta = \theta_1 + \theta_2 \ldots + \theta_m$.

R_{θ_s} defined by the formula (11) and P_{θ_s} defined by the formula (12), will be used in the following paragraphs.

2.5. Solving the Trigonometric Equation

This section summarizes two common formula solutions for trigonometric functions. The first can be derived using Equation (4):

$$ {}^0_iP = {}^0_1P + {}^0_1R\sum_{n=2}^{n=i} {}^1_2R\cdots{}^{n-1}_nP = \begin{bmatrix} \left({}^1_iP_y sin\alpha_1 - n_1{}^1_iP_x\right)sin\theta_1 + \left({}^1_iP_z a_1 + a_1\right)cos\theta_1 \\ \left({}^1_iP_z a_1 + a_1\right)sin\theta_1 - \left(n_1{}^1_iP_y - n_1{}^1_iP_x\right)cos\theta_1 \\ sin\alpha_1{}^1_iP_x + n_1{}^1_iP_y + \left(1 + {}^1_iP_z\right)d_1 \end{bmatrix}, \tag{13} $$

where ${}^1_iP_x$, ${}^1_iP_y$, and ${}^1_iP_z$ represent the three components in ${}^0_1R^{-1}\left({}^0_iP - {}^0_1P\right)$ and $n_i = sign(\alpha_i)$. According to Equation (18), ${}^0_iP_x$, ${}^0_iP_y$, and θ_1 satisfy a special trigonometric function relation, as shown in Equation (14):

$$ \begin{bmatrix} {}^0_iP_x \\ {}^0_iP_y \end{bmatrix} = \begin{bmatrix} D & L \\ L & -D \end{bmatrix}\begin{bmatrix} sin\theta_1 \\ cos\theta_1 \end{bmatrix}, (D \neq 0 \; or \; L \neq 0). \tag{14} $$

Therefore,

$$ \theta_1 = atan2\left(D^0_iP_x + L^0_iP_y, L^0_iP_x - D^0_iP_y\right). \tag{15} $$

Another common trigonometric equation is shown in Equation (16) from [17]:

$$ Acos\theta + Bsin\theta = C, (A \neq 0 \; or \; B \neq 0), \tag{16} $$

whose solution is:

$$ \theta = 2atan\left(\frac{B \pm \sqrt{B^2 + A^2 - C^2}}{A + C}\right). \tag{17} $$

Equations (15) and (17) can map the joint angles within the interval of $[-\pi, \pi]$. Moreover, in the following paragraphs, the robotic inverse solution equation is transformed into Equations (14) and (16), resulting in a fixed formula, which can compute the equation parameters and further derive the common closed-form formula solution for serial robots. When the parameters for Equations (14) and (16) are zero, the equation has no solution. This provides a suitable answer to the question of whether a robot has a solution when the linkage parameters are sparse. For convenience, Equation (14) is represented as $\theta_1 = F_1(D, L, x, y)$ and Equation (15) is represented as $\theta_{k1}, \theta_{k2} = F_2(A, B, C)$.

3. Analysis of The Inverse Kinematics Problem

Given a transformational matrix with a known end ${}^0_iT = \begin{bmatrix} {}^0_iR & {}^0_iP \\ 0^{1\times3} & 1 \end{bmatrix}$, the joint angle is obtained inversely from the equation provided by ${}^0_iR$ and ${}^0_iP$; this is known as inverse kinematics.

3.1. Three Types of Sub-Problems Related to Closed-Form Inverse Kinematics

The French mathematician Évariste Galois proved that the quartic polynomial equation has a closed-form solution, whereas the sextic polynomial equation generally has none. The rotational and translational components of the robotic end transformational matrix provide three independent equations separately. The two equation sets are combined to solve the inverse kinematics problem, and then high-order polynomial equations may be required. Therefore, a robot has a closed-form inverse solution, partial joint angles will be preferably solved from a certain equation set followed by the other joint angles.

To guarantee the existence of a closed-form solution, a feasible solution guarantees mutual decoupling between the translational and rotational components [17,26]. Such traditional decoupling

enables the translational component to be associated with the front three joints, which in turn, are solved via ${}^{0}_{3}P$. The latter three joints are solved via the rotational component ${}^{3}_{6}R$.

$$ {}^{0}_{6}T = \begin{bmatrix} {}^{0}_{3}R & {}^{0}_{3}P \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{3}_{6}R & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} {}^{0}_{3}R\,{}^{3}_{6}R & {}^{0}_{3}P \\ 0 & 1 \end{bmatrix}. \tag{18} $$

Based on the derivation in Section 2.4, a robot has several parallel joints, the number of unknown numbers in the rotational component will decrease. Therefore, another feasible decoupling idea is to preferably find a solution using the rotational component ${}^{0}_{6}R(\varphi, \vartheta, \psi)$ and solve for the remaining joint angles using the translational component ${}^{0}_{6}P$. This strategy has not been mentioned in any previous studies.

$$ {}^{0}_{6}T = \begin{bmatrix} {}^{0}_{6}R & {}^{0}_{6}P \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} {}^{0}_{6}R(\varphi, \vartheta, \psi) & {}^{0}_{6}P \\ 0 & 1 \end{bmatrix}. \tag{19} $$

Based on the above two decoupling methods, we could obtain three types of sub-problems.

In the first type, when the number of unknowns in the rotational component is larger than three and the robot meets some structural characteristics, some joint angles can be preferably and directly solved using the translational component. This article refers to the first type sub-problem as the sub-problem of translation components.

In the second type, the robot has several parallel joints and the number of unknowns in the rotational component is no greater than 3, these unknowns can be solved using the rotational component. This article refers to the second type sub-problem as sub-problem of rotational components.

In the third and final type, when the second type is used to solve the unknowns for the robot and the joint angles are still not completely solved, the remaining joint angles are obtained using the translational component. This article refers to the third type sub-problem as a sub-problem of sub-chains.

Via an analysis of these sub-problems, it is possible to determine the solving conditions and methods in detail. In Sections 3.2–3.4, a more detailed analysis and derivation will be made for each sub-problem, and the formulas and conditions for solving the basic problems will be obtained.

3.2. The First Type of Sub-problem

When there exist more than three unknowns in the rotational component, the partial joint angles should preferably be solved using the translational component. With reference to the DH parameter table, the DOF of the robot is i and the number of the DOF with $\alpha = 0$ in the first $i-1$ DOFs is p. In the case of $i - p > 3$, there are more than three unknowns in the rotational component.

According to Equation (4), the back joint angles are influenced by the front joint angles in the translational component. Therefore, the association with a maximum of the front three joint angles in the translational equation must be first guaranteed to solve the first type of sub-problem.

When the first joint angle of the robot model $Robot_i$ with i DOFs is solved, ${}^{0}_{1}T$ can be calculated using the forward kinematics formula. Meanwhile, the first row of the DH parameters is deleted to form a new robot model $Robot_{i-1}$ and a new transformational matrix ${}^{0}_{i-1}T$:

$$ {}^{1}_{i}T = {}^{0}_{1}T^{-1}\,{}^{0}_{i}T = \begin{bmatrix} {}^{0}_{1}R^{-1} & -{}^{0}_{1}R^{-1}\,{}^{0}_{1}P \\ 0^{1\times 3} & 1 \end{bmatrix} {}^{0}_{i}T. \tag{20} $$

There may only be a parallel or vertical relationship between two adjacent joints. Since only the linkage relation of the front three axes is considered, the possible configuration between the first three axes is only the case where $C^1_2 C^1_2 = 4$. Moreover, $z_1 \perp z_2 \perp z_3$, $z_1 \parallel z_2 \perp z_3$, $z_1 \perp z_2 \parallel z_3$ and $z_1 \parallel z_2 \parallel z_3$. However, during the process of derivation, it was found that $z_1 \parallel z_2 \parallel z_3$ only provides two valid equations in X-direction and Y-direction, making it impossible to solve for three unknowns:

$$ {}^{0}_{3}P = \begin{bmatrix} a_3cos(\theta_1+\theta_2+\theta_3)+a_2cos(\theta_1+\theta_2)+a_1cos\theta_1 \\ a_3sin(\theta_1+\theta_2+\theta_3)+a_2sin(\theta_1+\theta_2)+a_1sin\theta_1 \\ d_1+d_2+d_3 \end{bmatrix}. \tag{21} $$

Such a case is the counter example mentioned in the preceding paragraphs. The Piper principle does not specifically describe such a case. Therefore, the linkage parametric limitations and the formula solution for the three basic problems are derived in detail in the following paragraphs.

3.2.1. First Three Joint Configured as $z_1 \parallel z_2 \perp z_3^a$

Here, the expression for the translational component is ${}^{0}_{3}P = P_1 + R_1P_2 + R_1R_2P_3$. However, $R_1R_2P_3$ contains the cubic term of the sine and cosine functions, which would directly increase the order of the equation to sextic. Only when $a_3 = 0$ will the requirements for a closed solution be met. According to Equation (18), The formula solution of θ_1 can be obtained as follows:

$$ \begin{matrix} \theta_{11} = F_1\left(D, L, {}^{0}_{3}P_x, {}^{0}_{3}P_y\right) \\ \theta_{12} = F_1\left(D, -L, {}^{0}_{3}P_x, {}^{0}_{3}P_y\right) \end{matrix}, \text{ where} \begin{cases} D = n_1d_2 \\ L = \sqrt{{}^{0}_{3}P_x{}^2 + {}^{0}_{3}P_y{}^2 - D^2} \end{cases}. \tag{22} $$

Once θ_1 is obtained, the forward kinematics formula can be used to forward simplify the robot to obtain the expression of ${}^{1}_{i}T$ and the DOFs of $i-1$. The basic problem can still be employed to seek a solution to θ_2.

To guarantee the existence of only the front two joint angles in the translational equation, the linkage of $z_{k>3}$ needs to satisfy $d = a = 0$. However, according to the end decoupling formula proposed in Section 2.3, the linkage parameters of the last joint are not restricted.

3.2.2. First Three Joint Configured as $z_1 \parallel z_2 \perp z_3$

The formula solution for $z_1 \parallel z_2 \perp z_3$ is derived in the following paragraphs. Its translational component can be written as ${}^{0}_{3}P = P_1 + P_{12} + R_{12}P_3$, and its translational equation can be arranged as follows:

$$ \begin{bmatrix} a_1cos\theta_1 + D\sin(\theta_1+\theta_2) + L\cos(\theta_1+\theta_2) \\ a_1sin\theta_1 + L\sin(\theta_1+\theta_2) - D\cos(\theta_1+\theta_2) \\ n_2a_3sin\theta_3 \end{bmatrix} = \begin{bmatrix} {}^{0}_{3}P_x \\ {}^{0}_{3}P_y \\ {}^{0}_{3}P_z - d_1 - d_2 \end{bmatrix}. \tag{23} $$

Arranging Equation (23) gives the formula solution Equation (24):

$$ \begin{aligned} &\theta_{31}, \theta_{32} = F_2(A_3, B_3, C_3), \text{ where} \begin{cases} A_3 = 0 \\ B_3 = n_2a_3 \\ C_3 = {}^{0}_{3}P_z - d_1 - d_2 \end{cases}; \\ &\theta_{11}, \theta_{12} = F_2(A_1, B_1, C_1), \text{ where} \begin{cases} A_1 = {}^{0}_{3}P_x \\ B_1 = {}^{0}_{3}P_y \\ D = n_2d_3 \\ L = a_2 + a_3cos\theta_3 \\ C_1 = \frac{{}^{0}_{3}P_x{}^2 + {}^{0}_{3}P_y{}^2 + a_1{}^2 - L^2 - D^2}{2a_1} \end{cases} \\ &\theta_1 + \theta_2 = F_1\left(\mathrm{D}, \mathrm{L}, \widetilde{{}^{0}_{3}P_x}, \widetilde{{}^{0}_{3}P_y}\right), \text{ where} \begin{cases} \widetilde{{}^{0}_{3}P_x} = {}^{0}_{3}P_x - a_1cos\theta_1 \\ \widetilde{{}^{0}_{3}P_y} = {}^{0}_{3}P_y - a_1sin\theta_1 \end{cases}; \\ &\theta_2 = f[(\theta_1+\theta_2) - \theta_1]_{\pm\pi}; \end{aligned} \tag{24} $$

where $f[\theta]_{\pm\pi}$ in the last equation indicates that θ is mapped onto the interval of $[-\pi, \pi]$.

Under such a structure, θ_3 is preferably solved using the equation in the Z-direction. Therefore, the existence of z_4^a after z_3 does not influence the solution of θ_3, resulting in the derivation of two different structures, $z_1 \parallel z_2 \perp z_3 \perp z_4^a$ and $z_1 \parallel z_2 \perp z_3 \parallel z_4^a$.

According to the translational equation of $z_1 \parallel z_2 \perp z_3 \perp z_4^a$ and Equation (23), the critical coefficient is:

$$\begin{cases} D = n_2 d_3 + n_2 d_4 \\ L = a_2 + a_3 \cos\theta_3 \end{cases}. \tag{25}$$

Moreover, according to Equation (23), the critical coefficient under $z_1 \parallel z_2 \perp z_3 \parallel z_4^a$ is:

$$A_3 = -n_2 n_3 d_4, \text{ where } \begin{cases} D = n_2 d_3 \\ L = a_2 + a_3 \cos\theta_3 + n_3 d_4 \sin\theta_3. \end{cases} \tag{26}$$

To guarantee that only the front three joint angles exist in the translational equation, the linkage of $z_{k>4}$ needs to satisfy $d = a = 0$. There is not restricted to the linkage parameters of the last joint.

In addition, in the basic problems mentioned above, when $d_4 = a_4 = 0$ is true in z_4, it can also be used to solve $z_1 \parallel z_2 \perp z_3$. Similarly, it can be used to solve $z_1 \parallel z_2$. Therefore, such a type of basic problem contains two sets of formula solutions that can be used to solve four cases.

3.2.3. First Three Joint Configured as $z_1 \perp z_2 \parallel z_3$

The solution method for $z_1 \perp z_2 \parallel z_3^a$ in Section 3.4.1 also applies here. In addition, if $z_1 \perp z_2 \parallel z_3 \perp z_4^a$ and $D = n_1(d_2 + d_3)$, θ_1 can still be directly solved. This results in a special situation. Further, suppose there is a robot with i degrees of freedom, in which the configuration of the first g joints is $z_1 \perp z_2 \parallel \ldots \parallel z_{g-1} \perp z_g^a$, and the linkage of $z_{k>g}$ needs to satisfy $d = a = 0$. These are not restricted to the linkage parameters of the last joint. The formula solution of θ_1 can be obtained as follows:

$$\begin{matrix} \theta_{11} = F_1\left(D, L, {}^0_gP_x, {}^0_gP_y\right) \\ \theta_{12} = F_1\left(D, -L, {}^0_gP_x, {}^0_gP_y\right) \end{matrix}, \text{ where } \begin{cases} D = n_1 \sum_{m=2}^{g-1} d_m \\ L = \pm\sqrt{{}^0_gP_x{}^2 + {}^0_gP_y{}^2 - D^2} \end{cases}. \tag{27}$$

According to the forward kinematics formula, after θ_1 is solved, ${}^0_1T$ can be cancelled to allow the robot model to reduce a joint angle and become a new model. At that time, selective solving can be performed in all of the sub-problems.

Basic problems 3.2.1 and 3.2.3 can be solved using the same formula solution. In the following paragraphs, basic problem 3.4.1 will be used for the description.

Therefore, there exist two types of basic problems in the first type of sub-problem. This section gives the corresponding relevant existence conditions and formula solutions.

3.3. Solving the Second Type of Sub-Problem

Since the rotational component only provides three independent equations, a maximum of three unknowns can be solved. First, the rotational matrix Rotx(θ) is introduced. Rotx(θ) represents a θ radian rotation around the X-coordinate axis. Similarly, Rotz(θ) and Roty(θ) represent rotations around the Y- and Z-axes. The Euler angle theorem indicates that a generic rotation matrix can be obtained by composing a suitable sequence of three rotations while guaranteeing that two successive rotations are not made around parallel axes [18]. Therefore, any rotational matrix R can be expressed as Rotz(φ), Roty(ϑ), and Rotz(ψ).

The DH parameters of a serial robot with a certain DOF are shown in Table 4.

Table 4. The DH parameters of a serial robot with three degrees-of-freedoms (DOFs).

n	θ/°	d/mm	a/mm	α/°
1	θ_1	d_1	a_1	$-\pi/2$
2	θ_2	d_2	a_2	$\pi/2$
3	θ_3	d_3	a_3	0

According to the forward kinematics formula,

$$\begin{gathered} {}^{0}_{3}R = \mathrm{Rotz}(\theta_1)\mathrm{Rotx}(\alpha_1)\mathrm{Rotz}(\theta_2)\mathrm{Rotx}(\alpha_2)\mathrm{Rotz}(\theta_3)\mathrm{Rotx}(\alpha_3), \\ {}^{0}_{3}R = \mathrm{Rotz}(\theta_1)\mathrm{Roty}(\theta_2)\mathrm{Rotz}(\theta_3). \end{gathered} \tag{28}$$

A serial robot has three continuous joints meeting $\alpha_1 = -\frac{\pi}{2}, \alpha_2 = \frac{\pi}{2}$ and $\alpha_3 = 0$, whose rotational components ${}^{0}_{3}R$ are known, two sets of solutions can be found using Euler's formula.

3.3.1. Three Mutually Perpendicular Sub-Chains

Based on α in the DH parameters, the conclusions in Section 2.4 can be used to simplify the sub-chains and obtain the expression of the rotational matrix,

$${}^{0}_{3}R = \begin{bmatrix} cos\theta_{s_1}cos\theta_{s_2}cos\theta_{s_3} + n_{s_1e}n_{s_2e}sin\theta_{s_1}sin\theta_{s_3} & n_{s_1e}n_{s_2e}sin\theta_{s_1}cos\theta_{s_3} - cos\theta_{s_1}cos\theta_{s_2}sin\theta_{s_3} & n_{s_2e}cos\theta_{s_1}sin\theta_{s_2} \\ sin\theta_{s_1}cos\theta_{s_2}cos\theta_{s_3} - n_{s_1e}n_{s_2e}cos\theta_{s_1}sin\theta_{s_3} & -n_{s_1e}n_{s_2e}cos\theta_{s_1}cos\theta_{s_3} - sin\theta_{s_1}cos\theta_{s_2}sin\theta_{s_3} & n_{s_2e}sin\theta_{s_1}sin\theta_{s_2} \\ n_{s_1e}sin\theta_{s_2}cos\theta_{s_3} & -n_{s_1e}sin\theta_{s_2}sin\theta_{s_3} & -n_{s_1e}n_{s_2e}cos\theta_{s_2} \end{bmatrix}, \tag{29}$$

for which the following expressions can be obtained:

$$\begin{cases} \theta_{s_1} = \mathrm{atan2}(n_{s_2e}r_{23}, n_{s_2e}r_{13}) \\ \theta_{s_2} = \mathrm{atan2}\left(\sqrt{{r_{13}}^2 + {r_{23}}^2}, -n_{s_1e}n_{s_2e}r_{33}\right) \\ \theta_{s_3} = \mathrm{atan2}(-n_{s_1e}r_{32}, n_{s_1e}r_{31}) \end{cases}, \tag{30a}$$

or

$$\begin{cases} \theta_{s_1} = \mathrm{atan2}(-n_{s_2e}r_{23}, -n_{s_2e}r_{13}) \\ \theta_{s_2} = \mathrm{atan2}\left(-\sqrt{{r_{13}}^2 + {r_{23}}^2}, -n_{s_1e}n_{s_2e}r_{33}\right) \\ \theta_{s_3} = \mathrm{atan2}(n_{s_1e}r_{32}, -n_{s_1e}r_{31}) \end{cases}, \tag{30b}$$

where θ_{s_k} represents the sum of the rotation angles of the kth sub-chain of the robot and n_{s_ke} represents the symbol of the last linkage α in the kth sub-chain. r_{xy} represents the element at the position of the xth row and the yth column of the rotation matrix ${}^{0}_{3}R$. The problem of solving the sum of the rotation angles of three mutually perpendicular sub-chains is referred to as basic problem 3.3.1.

3.3.2. Two Mutually Perpendicular Sub-Chains

Similarly, when there are only two mutually perpendicular sub-chains, the expression of the rotational component at that time is as follows:

$${}^{0}_{2}R = \begin{bmatrix} cos\theta_{s_1}cos\theta_{s_2} & -cos\theta_{s_1}sin\theta_{s_2} & n_{s_1e}sin\theta_{s_1} \\ sin\theta_{s_1}cos\theta_{s_2} & -sin\theta_{s_1}sin\theta_{s_2} & -n_{s_1e}cos\theta_{s_1} \\ n_{s_1e}sin\theta_{s_2} & n_{s_1e}cos\theta_{s_2} & 0 \end{bmatrix}, \tag{31}$$

with the corresponding solution:

$$\begin{cases} \theta_{s_2} = \mathrm{atan2}(n_{s_1e}r_{31}, n_{s_1e}r_{33}) \\ \theta_{s_1} = \mathrm{atan2}\left(\frac{r_{21}}{cos\theta_{s_2}}, \frac{r_{11}}{cos\theta_{s_2}}\right) \end{cases}. \tag{32}$$

r_{xy} represents the element at the position of the xth row and the yth column of the rotation matrix ${}^{0}_{2}R$. The problem of solving the sum of the rotational degrees of two mutually perpendicular sub-chains is referred to as basic problem 3.3.2.

3.4. The Third Type of Sub-Problem

Before the second type of sub-problem is employed by the governing algorithm, the DOF of the robot at that time is i, then the number of DOFs with $\alpha = 0$ in the first $i-1$ DOFs is p. If $p \neq 0$, not all

joint angles are obtained by the second type of sub-problem. In this case, the translational component needs to be employed to solve for the remaining joint angles. Such problems are called the third type of sub-problem.

For any of this type of sub-problem, when the first chain only has one joint, the forward kinematics formula can be directly used to simplify the robotic model. Based on that and according to Equation (4), the translational components of a robot with a DOF of i can be divided into i components. Since the second type of sub-problem can solve the sum of the rotational angles of the sub-chains of the robot, based on Equation (17), the translational equation can be arranged as

$$ {}^{0}_{i}P - P_{\theta_{s_1}} - R_{\theta_{s_1}} P_{\theta_{s_2}} - R_{\theta_{s_1}} R_{\theta_{s_2}} P_{\theta_{s_3}} = \sum_{n=1}^{n=i} {}^{n-1}_{n}P - P_{\theta_{s_1}} - R_{\theta_{s_1}} P_{\theta_{s_2}} - R_{\theta_{s_1}} R_{\theta_{s_2}} P_{\theta_{s_3}}. \tag{33} $$

$P_{\theta_{s_k}}$ and $P_{\theta_{s_k}}$ represent the rotational and translational components of the kth sub-chain, which is defined by the formula (11) and formula (12). All terms on the left side of Equation (33) are known and are written as $\hat{P}$ in the following paragraphs. Conversely, there are unknown quantities on the right side of the equation. Since the third type of problem has a maximum of three sections of sub-chains, $[s_1, s_2, s_3]$ is used to represent the length of each sub-chain on the right side of the equation. Depending on the length of each sub-chain, the appropriate formula solutions are presented with $s_i t$ representing the tth joint in the ith section of the sub-chain.

As an example, for $z_1 \parallel z_2 \perp z_3 \parallel z_4$, there exist two sub-chains, s_1 and s_2, either of which has a sub-chain length of two. After the second type of sub-problem is solved, the length of each sub-chain decreases by one. For example, the length of each sub-chain on the right side of Equation (33) is $[1, 1, 0]$.

All possible cases will be analyzed and solved for in the following cases.

3.4.1. Sub-Chain Length as [1,0,0]

$$ \begin{aligned} \theta_{s_1 1} &= atan2\left(a_{s_1 1}\hat{P}_y, a_{s_1 1}\hat{P}_x\right). \\ \theta_{s_1 2} &= \left[\theta_{s_1} - \theta_{s_1 1}\right]_{\pm\pi}. \end{aligned} \tag{34} $$

3.4.2. Sub-Chain Length as [2,0,0]

$$ \begin{aligned} \theta_{s_1 11}, \theta_{s_1 12} &= F_2(\mathrm{A}, \mathrm{B}, \mathrm{C}), \text{ where } \begin{cases} A = \hat{P}_x \\ B = \hat{P}_y \\ C = \frac{\hat{P}_x{}^2 + \hat{P}_y{}^2 + a_{s_1 1}{}^2 - a_{s_1 2}{}^2}{2a_{s_1 1}} \end{cases}. \\ \theta_{s_1 1} + \theta_{s_1 2} &= F_1\left(\mathrm{D}, \mathrm{L}, \widetilde{P_x}, \widetilde{P_y}\right), \text{ where } \begin{cases} \widetilde{P_x} = \hat{P}_x - a_{s_1 1}\cos\theta_{s_1 1} \\ \widetilde{P_y} = \hat{P}_y - a_{s_1 1}\sin\theta_{s_1 1} \\ L = a_{s_1 2} \\ D = 0 \end{cases} \\ \theta_{s_1 2} &= f\left[\left(\theta_{s_1 1} + \theta_{s_1 2}\right) - \theta_{s_1 1}\right]_{\pm\pi} \\ \theta_{s_1 3} &= f\left[\theta_{s_1} - \left(\theta_{s_1 1} + \theta_{s_1 2}\right)\right]_{\pm\pi} \end{aligned} \tag{35} $$

3.4.3. Sub-Chain Length as [2,1,0] and [1,1,0]

$$ \theta_{s_2 11}, \theta_{s_2 12} = F_2(A, \mathrm{B}, \mathrm{C}), \text{ where } \begin{cases} A = 0 \\ B = n_{s_1 e} a_{s_2 1} \\ C = \hat{P}_z - d_{s_1 1} - d_{s_1 2} \end{cases}. \tag{36a} $$

$$ \theta_{s_2 2} = f\left[\theta_{s_1} - \theta_{s_1 1}\right]_{\pm\pi}. \tag{36b} $$

$$\hat{P} - R_{s_1} P_{s_2 1} = P_{s_1 1} + R_{s_k 1} P_{s_k 2}. \tag{36c}$$

$R_{s_k t}$ and $P_{s_k t}$ represent the rotational and translational components of the tth joint in the kth section of the sub-chain. According to Equations (36a) and (36b), all joint angles in the sub-chain s_2 can be solved. Moreover, according to Equation (36c), the first sub-chain solving problem is transformed into either basic problem 3.6.1 or basic problem 3.6.2.

3.4.4. Sub-Chain Length as [1,2,0]

$$\theta_{s_1 11}, \theta_{s_1 12} = F_2(A, \mathrm{B}, \mathrm{C}), \text{ where } \begin{cases} A = n_{s_1 2} a_{s_1 2} \sin\theta_{s_1} \\ B = -n_{s_1 2} a_{s_2 1} \sin\theta_{s_1} \\ C = \left(R_{s_1}{}^T \hat{P}\right)_z - d_{s_2 1} - d_{s_2 2} \end{cases}. \tag{37a}$$

$$\theta_{s_1 2} = f\left[\theta_{s_1} - \theta_{s_1 1}\right]_{\pm\pi}. \tag{37b}$$

$$R_{s_1}{}^T\left(\hat{P} - P_{s_1 1}\right) = P_{s_2 1} + R_{s_2 1} P_{s_2 2}. \tag{37c}$$

According to Equations (37a) and (37b), we could solve all joint angles in the sub-chain s_1. According to Equation (37c), the second sub-chain solving problem can be transformed into basic problem 3.4.2.

3.4.5. Sub-Chain Length as [2,0,1] or [1,0,1]

$$\theta_{s_3 11}, \theta_{s_3 12} = F_2(A, B, C), \text{ where } \begin{cases} A = 0 \\ B = n_{s_1 e} a_{s_3 1} sin\theta_{s_3} \\ C = \hat{P}_z - d_{s_1 1} - d_{s_1 2} + n_{s_1 3} n_{s_2 1} d_{s_3 1} cos\theta_{s_3} \end{cases}. \tag{38a}$$

$$\theta_{s_3 2} = f\left[\theta_{s_3} - \theta_{s_3 1}\right]_{\pm\pi}. \tag{38b}$$

$$\hat{P} - R_{s_1} R_{s_2} P_{s_3 2} = P_{s_2 1} + R_{s_2 1} P_{s_2 2}. \tag{38c}$$

According to Equations (38a) and (38b), all joint angles in the sub-chain s_3 can be solved. According to Equation (38c), the first sub-chain solving problem can be transformed into either basic problem 3.4.2 or basic problem 3.4.1.

3.4.6. Sub-Chain Length as [1,0,2]

$$\theta_{s_1 21}, \theta_{s_1 22} = F_2(A, \mathrm{B}, \mathrm{C}), \text{ where } \begin{cases} A = 0 \\ B = n_{s_2 1} a_{s_1 1} sin\theta_{s_2} \\ C = (R_{s_2}{}^T R_{s_1}{}^T \hat{P})_z - d_{s_3 1} - d_{s_3 2} + n_{s_1 2} n_{s_2 1} d_{s_1 1} cos\theta_{s_2} \end{cases}. \tag{39a}$$

$$\theta_{s_1 1} = f\left[\theta_{s_1} - \theta_{s_1 2}\right]_{\pm\pi}. \tag{39b}$$

$$R_{s_2}{}^T R_{s_1}{}^T \hat{P} - R_{s_2}{}^T R_{s_1}{}^T P_{s_1 1} = P_{s_3 1} + R_{s_3 1} P_{s_3 2}. \tag{39c}$$

According to Equations (39a) and (39b), we could solve all joint angles in the sub-chain s_1. According to Equation (39c), the third sub-chain solving problem can be transformed into basic problem 3.4.2.

In the case of [1,1,1], it may be impossible to find a closed-form solution due to the high complexity of the equation. Therefore, the third type of sub-problem has six basic problems.

3.5. Summary

The above-mentioned content proposed the two decoupling methods of series robots firstly. Three types of sub-problems are proposed through these two decoupling methods. Ten basic problems were derived through analyzing the solvable condition and solution method of each type of sub-problems specifically. If a series robot can be described through the three types of sub-problem based on these ten basic problems, the robot must be a closed inverse solution.

P represents the number of the DOF with $\alpha = 0$ in the first $i-1$ DOFs, i.e., the number of $\alpha = 0$ in the first $i-1$ rows of the DH parameter table.

In the case of $i - P > 3$, it means that there are more vertical joints in the robot. There will be more than three unknown numbers that cannot be solved in the rotational component. If there is a closed solution, a low-order trigonometric function equation can only be solved through translation components. There are two basic problems that are feasible. The restrictions and solution formula of the two basic problems is shown in Table 5.

Table 5. The restrictions and solution formula of the two basic problems.

Base Problem	Configuration Restrict	Solution Formula
3.2.1	1. The configuration of the first g joints is $z_1 \perp z_2 \parallel \ldots \parallel z_{g-1} \perp z_g^a$, 2. The linkage of $z_{k>g}$ needs to satisfy $d = a = 0$.. 3. The linkage parameters of the last joint are not restricted.	Equation (30) can be used to solve θ_1
3.2.2	1. The configuration of the first three joints is $z_1 \parallel z_2 \perp z_3$, 2. The linkage of $z_{k>4}$ needs to satisfy $d = a = 0$. 3. The linkage parameters of the last joint are not restricted.	Equations (25) and (26) can be used to solve critical coefficient, formula (24) can be used to solve the angle of first three joints.

In the case of $i - P < 3$, it means that there are more parallel joints in the robot. There will be less than three unknown numbers in the rotational component, then, the unknown number can be solved through in the rotational component directly. The parallel joints in the rotation component may be an independent joint angle or the sum of the rotation angles of the sub-chains; therefore, they can be represented as unknown number here. In the case of $i - P = 3$, the formula (30) can be used to solve the three-unknown number. In the case of $i - P = 2$, the formula (32) can be used to solve the two-unknown number. In the case of $i - P = 1$, the first formula in the formula (32) can be used to solve the only unknown number.

If all joint angles are not solved after using the second sub-problems, it indicates that there is long sub-chain in the translational component. At this time, the remaining joint angles need to be solved by combing the third sub-problems with the result of the second sub-problems. The six basic problems in the third sub-problems are shown in Table 6.

Table 6. The six basic problems in the third sub-problems.

Base Problem	Remaining Length of Each Sub-Chain	Solution Formula
3.4.1	[1,0,0]	Equation (34)
3.4.2	[2,0,0]	Equation (35)
3.4.3	[2,1,0], [1,1,0]	Equation (36) and Base problem 3.6.2.
3.4.4	[1,2,0]	Equation (37) and Base problem 3.6.2.
3.4.5	[2,0,1], [1,0,1]	Equation (38) and Base problem 3.6.2 or 3.6.1.
3.4.6	[1,0,2]	Equation (39) and Base problem 3.6.2.

Since the third type of problem has a maximum of three sections of sub-chains, $[s_1, s_2, s_3]$ is used to represent the length of each sub-chain on the right side of the equation. Depending on the length of

each sub-chain, the appropriate formula solutions are presented with $s_k t$ representing the tth joint in the kth section of the sub-chain.

4. Algorithm Design for Universal Inverse Kinematics

The translational and rotational components of a serial robot with a closed-form solution are mutually decoupled. Therefore, the partial joint angles of the robot can be solved based on the first or second type of sub-problem. This simplifies the original robot model into a new robot model. If the new robot model has a closed-form solution, the solution will definitely continue along one of the three types of sub-problems until all the problems are solved. Therefore, the process of solving the closed-form inverse solution of a serial robot can be completed via constant model simplification. The logical flow chart of the entire algorithm is shown in Figure 2.

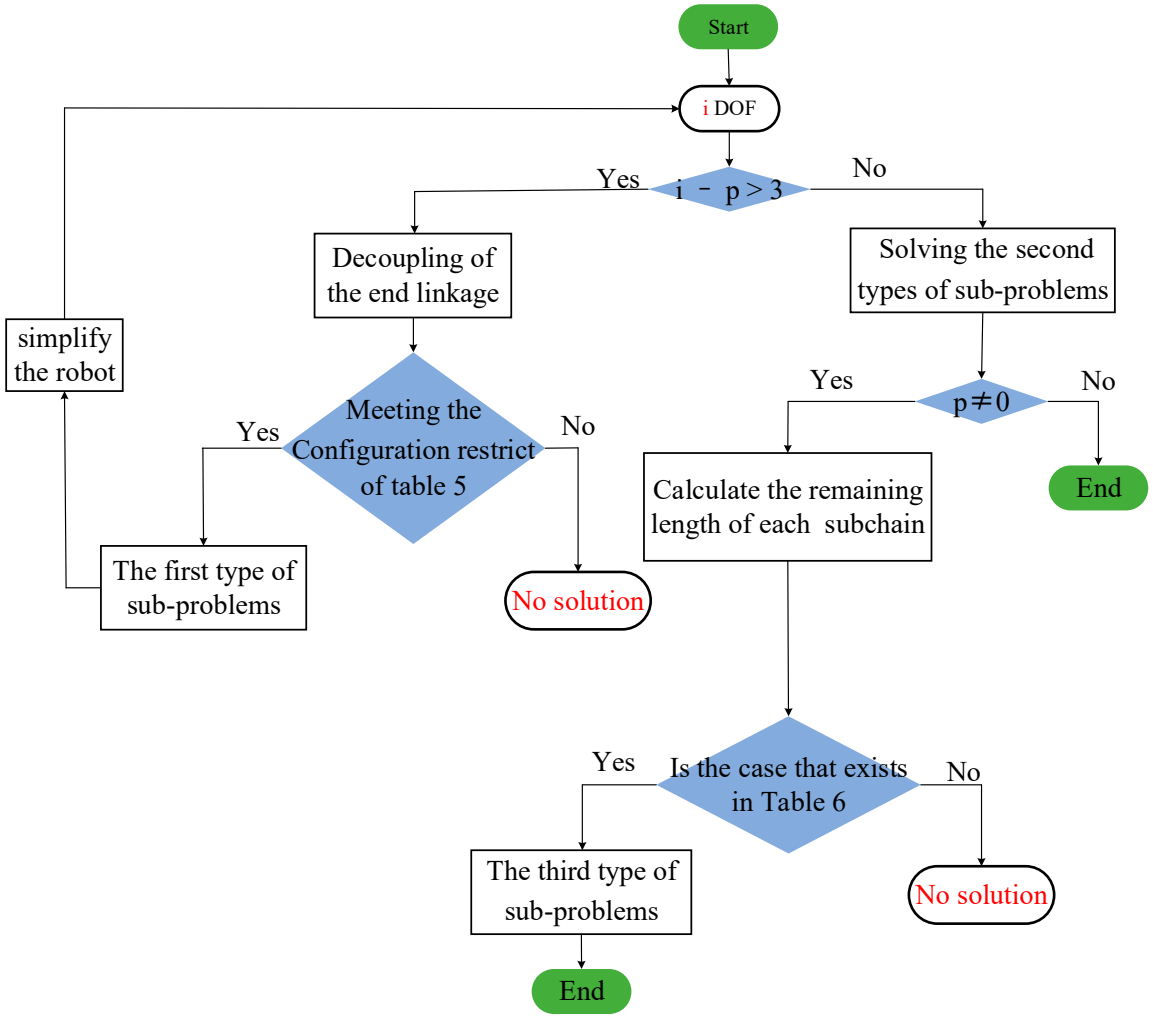


Figure 2. Logic for the universal inverse kinematics algorithm.

In Figure 2, P represents the number of the DOF with $\alpha = 0$ in the first $i-1$ DOFs, i.e., the number of $\alpha = 0$ in the first $i-1$ rows of the DH parameter table.

The inverse kinematics problem has multiple solutions. Nonetheless, the selection of these solutions necessitates a relevant upper logic design. In addition, changes in the robotic joints must be continuous:

$$arg\ \min_{k} \sum_{i=1}^{6} \left| \theta_{i,cur} - \theta_{i,k} \right|. \tag{40}$$

In cases where the robot is in a stationary state, the joint angle of the k set can be first solved and then Equation (40) can be used to determine the inverse motion index value corresponding to the current position.

The main purpose of this paper was to find a more complete inverse solution judgment condition, which is more accurate and specific than the Pieper principle. The general inverse solution algorithm designed based on this judgment condition is also directed to a series robot with a closed inverse solution. For this paper purposed ten basic problems, which means a series of robots with closed inverse solutions can be obtained through combining the basic problems. For example, although $z_1 \parallel z_2 \perp z_3 \parallel z_4^a \perp z_5^{a,d} \perp z_6$, or $z_1 \parallel z_2 \perp z_3 \parallel z_4 \parallel z_5 \parallel z_6$ are not common, but it is indeed that there are robots with a closed inverse solution. In addition, maintaining the robot configuration does not change, only the direction of rotation and the positive or negative of the offset are changed, the algorithm can still calculate correctly. This can provide great convenience for the application of an engineer. Finally, the logical design of the algorithm is complete. This means that once you use a situation other than ten basic questions to design the robot, the algorithm will jump out and terminate. Therefore, the algorithm is not to solve the inverse solution problem of any robot, but to judge whether the closed inverse solution exists and to solve it.

5. Experiments

The previous section discussed the design of the universal inverse kinematics solution algorithm. In this section, four of the experiments will be carried out to verify the completeness, versatility, stability, and real-time performance of the algorithm.

Frist of all, the algorithm was implemented in MATLAB 2016b on a 64-bit Windows 10 operating system. The completeness, versatility and stability of the algorithm will be verified in MATLAB. Subsequently, the algorithm was performed using the Beckhoff-C6920 controller. We built a six-DOF KingKong robot using the Kollmorgen RGM motor for the real-time testing.

5.1. Verification Experiment for Algorithm Completeness

Firstly, the algorithm designed in this paper was logically complete. The first simulation experiment would test the completeness of the algorithm. Here, a robot with three parallel joints was selected as the experimental object. The DH parameter of it is shown in the Table 7. This robot is a common series robot, but it cannot be described by a single sub-problem.

Table 7. The DH parameters of the robot with three parallel joints.

n	θ/°	d/mm	a/mm	α/°
1	θ_1	0.1	0.1	0
2	θ_2	0.1	−0.2	0
3	θ_3	0.1	0.3	0

First, the algorithm receives the DH parameters of the robot. Algorithm will enter in the second type of sub-problem to solve the rotation angle and θ_{s_1} of the sub-chain s_1. The length of the sub-chain was $[2, 0, 0]$ after been simplified and calculated, therefore, the remaining joint angles was solved by the basic problem 3.6.2 in the third type sub-problems.

Consequently, the initial position was $\Theta = \begin{bmatrix} 90^\circ \\ 90^\circ \\ 45^\circ \end{bmatrix}$, and the end posture was:

$$\begin{bmatrix} -0.7071 & 0.7071 & 0 & -0.0121 \\ -0.7071 & -0.7071 & 0 & -0.1121 \\ 0 & 0 & 1 & 0.3 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (41)$$

All joint angles were solved with the algorithm, as shown in Table 8.

Table 8. Two inversely solved joint angle sets.

No.	θ_1/°	θ_2/°	θ_3/°
1	90	90	45
2	−36.8699	−90	−8.1301

It is easy to verify that the two sets of joint angles are correct.

A robot with four parallel joints was now used as the experimental object. Its DH parameters are shown in Table 9.

Table 9. The DH parameters of the robot with three parallel joints.

n	θ/°	*d*/m	*a*/m	α/°
1	θ_1	0.1	0.1	0
2	θ_2	0.1	−0.2	0
3	θ_3	0.1	0.3	0
4	θ_4	0.1	0.1	0

The algorithm will enter in the second type of sub-problem the solve the rotation angle and θ_{s_1} of the sub-chain s_1. The length of the sub-chain was [3,0,0] after being simplified and calculated, the algorithm was ended for it was not in the basic problem of the third type sub-problems. In fact, there are only three valid equations for series robots with four-degree-of-freedom, which is impossible to solve its inverse motion problem.

Experiment 5.1 verified the completeness of the algorithm by two examples. The first example was a three-parallel joint robot, the correct joint angles were obtained through an algorithm. Then, we used a four-parallel joint robot to test, and the algorithm was terminated for it was not in the basic problems. This shows that the algorithm would exit correctly when it encounters an unsolvable problem, and it will be able to calculate the result when it encounters a solvable problem.

This experiment also proved that the proposed judgment method could be used to low-degree-of-freedom robots. In addition, when there were four adjacent parallel joints in a six-degree-of-freedom robot, the Pieper criterion was satisfied, but the proposed algorithm could judge that the robot did not have a closed inverse solution. This complements the shortcomings of the Pieper criterion.

5.2. Verification Experiment of Algorithm Versatility

The proposed algorithm could solve those serial robots that could be described by three types of sub-problems. Since the description of the ten basic problems was determined, a series robot with a closed-form inverse solution could be constructed based on ten basic problems. Two uncommon robots were used as examples to verify the versatility of the algorithm in this experiment.

We constructed a robot Bot_1, whose structure was $z_1 \parallel z_2 \perp z_3 \parallel z_4^a \perp z_5^{a,d} \perp z_6$ based on basic problem 3.4.2 and 3.5.1. Its DH parameters are shown in Table 10.

Table 10. The DH parameters of Bot_1.

n	θ/°	*d*/m	*a*/m	α/°
1	θ_1	0.1	0.35	0
2	θ_2	0.1	0.3	$\pi/2$
3	θ_3	0.1	0.5	0
4	θ_4	0.1	0	$-\pi/2$
5	θ_5	0	0	$\pi/2$
6	θ_6	0.1	0.1	0

The robot had three vertical joints, so the decoupling of the end was carried out firstly. Then, we found that $z_1 \parallel z_2 \perp z_3 \parallel z_4^a \perp z_5^{a,d}$ could be solved by the basic problem 3.2.2. When the first three joint angles were solved, the simplified robot $z_4^a \perp z_5^{a,d} \perp z_6$ could be solved by the second type of sub-problems.

Consequently, the initial position was $\Theta = \begin{bmatrix} 90° \\ 60° \\ 60° \\ 60° \\ 30° \\ 30° \end{bmatrix}$, and the end posture was:

$$\begin{bmatrix} 0.5625 & 0.1752 & 0.8080 & -0.0228 \\ -0.8248 & 0.1875 & 0.5335 & 0.6441 \\ -0.0580 & -0.9665 & 0.2500 & 0.7192 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (42)$$

All joint angles were solved with an algorithm, as shown in Table 11.

Table 11. Four inversely solved joint angle sets.

No.	θ_1/°	θ_2/°	θ_3/°	θ_4/°	θ_5/°	θ_6/°
1	90.0000	60.0000	90.0000	−120.0000	−30.0000	−150.0000
2	90.0000	60.0000	90.0000	60.0000	30.0000	30.0000
3	116.7078	7.3801	90.0000	−177.4728	−14.4919	−89.1752
4	116.7078	7.3801	90.0000	2.5272	14.4919	90.8248

It is easy to verify that the four sets of joint angles are correct. Then, we kept the configuration of the robot unchanged and modified the positive direction of the rotation of the robot joint, the positive and negative of a and the value of d. This was used to verify whether the algorithm could cope with parameter changes. The changed DH parameters are shown in Table 12.

Table 12. The DH parameters of Bot_1 after changing the parameters.

n	θ/°	d/m	a/m	α/°
1	θ_1	0	−0.35	0
2	θ_2	0	0.3	$\pi/2$
3	θ_3	0	−0.5	0
4	θ_4	0	0	$\pi/2$
5	θ_5	0	0	$\pi/2$
6	θ_6	0	0.1	0

Consequently, the initial position was $\Theta = \begin{bmatrix} 90° \\ 60° \\ 60° \\ 60° \\ 30° \\ 30° \end{bmatrix}$, and the end posture was:

$$\begin{bmatrix} 0.5625 & -0.8248 & 0.0580 & -0.2036 \\ 0.1752 & 0.1875 & 0.9665 & -0.1825 \\ -0.8080 & -0.5335 & 0.2500 & 0.4192 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (43)$$

All joint angles were computed with the algorithm, as shown in Table 13

Table 13. Four inversely solved joint angle sets.

No.	θ_1/°	θ_2/°	θ_3/°	θ_4/°	θ_5/°	θ_6/°
1	90.0000	60.0000	90.0000	−120.0000	−30.0000	−150.0000
2	90.0000	60.0000	90.0000	60.0000	30.0000	30.0000
3	−14.8218	−60.0000	90.0000	105.2403	−108.0015	51.7522
4	−14.8218	−60.0000	90.0000	−74.7597	108.0015	−128.2478

It is easy to verify that the four sets of joint angles are correct.

We constructed the robot Bot_2, whose structure was $z_1 \parallel z_2 \perp z_3 \parallel z_4 \parallel z_5 \parallel z_6$ based on basic problem 3.5.1 and 3.6.6. Its DH parameters are shown in Table 14.

Table 14. The DH parameters of Bot_2.

n	θ/°	d/m	a/m	α/°
1	θ_1	0.1	0.35	0
2	θ_2	0.1	0.3	$-\pi/2$
3	θ_3	0.1	0.3	$\pi/2$
4	θ_4	0.1	0.25	0
5	θ_5	0.1	0.2	0
6	θ_6	0.1	0.1	0

The robot had three sub-chains, and the second type of sub-problem was firstly used to obtain the sum of the rotation angle of the three segments. Then, algorithm turned into the third type sub-problems for there were joint angles that had not been solved. In the third sub-problem, we could use the basic problem 3.6.6 to solve, and then solve all joint angles.

Consequently, the initial position was $\Theta = \begin{bmatrix} 90^\circ \\ 60^\circ \\ 60^\circ \\ 45^\circ \\ 45^\circ \\ 45^\circ \end{bmatrix}$, and the end posture was:

$$\begin{bmatrix} -0.0474 & 0.6597 & -0.7500 & -0.9344 \\ -0.7891 & 0.4356 & 0.4330 & 0.2573 \\ 0.6124 & 0.6124 & 0.5000 & -0.0017 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (44)$$

All joint angles were solved with the algorithm, as shown in Table 15.

Table 15. Four inversely solved joint angle sets.

No.	θ_1/°	θ_2/°	θ_3/°	θ_4/°	θ_5/°	θ_6/°
1	−150.0000	−60.0000	60.0000	−95.1093	100.7234	129.3859
2	−150.0000	−60.0000	60.0000	−9.6646	−100.7234	−114.6120
3	90.0000	60.0000	60.0000	45.0000	45.0000	45.0000
4	90.0000	60.0000	60.0000	84.7298	−45.0000	95.2702

It is easy to verify that the four sets of joint angles are correct.

In this group of experiments, the test subjects were two uncommon robots constructed by basic questions. However, according to the proposed theory, these two robots had a closed inverse solution. In the experiment, the inverse solution of both robots was solved correctly. In addition, for the first robot, its configuration remains unchanged, but the link parameters were greatly modified, the correct

result was obtained in the end. It could be seen that the algorithm had good versatility. As long as a given robot can be described by three sub-problems, its closed-from solution can be obtained whether it is in low degree of freedom or six degrees of freedom. At the same time, for the robot with the same configuration, the change of the link parameters does not affect the inverse kinematics solution.

5.3. Verification Experiment for the Algorithm Continuity

This experiment would use the common Puma560 as the subject. In the experiment, the robot should move according to the specified trajectory. We needed to observe whether the joint angle after the inverse solution was continuous when solving the continuous space trajectory. This is important in the actual movement of robot. If there is a jump or discontinuity in the joint space curve, it will cause great impact to the motor, and the curve of the end also does not move according to the specified trajectory. The DH parameters of the Puma560 robot are shown in Table 16.

Table 16. The DH parameters of the Puma560 robot.

n	θ/°	d/mm	a/mm	α/°
1	θ_1	0	0	$\pi/2$
2	θ_2	0	43.18	0
3	θ_3	150.03	20.3	$-\pi/2$
4	θ_4	43.18	0	$\pi/2$
5	θ_5	0	0	$-\pi/2$
6	θ_6	0	0	0

The quantity of the vertical joints was judged following the input of the algorithm into the DH model of the Puma560 robot. Subsequently, the algorithm entered the first type of sub-problem. Then, joint decoupling was performed for J_6 and $J_1 \perp J_2 \parallel J_3 \perp J_{4,a} \perp J_{5,a,d}$. Based on basic problem 3.4.1, the angle value of θ_1 was obtained. In addition, after θ_1 was obtained, ${}^1_6T$ was obtained using to the forward kinematics formula. After it was simplified, $J_2 \parallel J_3 \perp J_{4,a} \perp J_{5,a,d} \perp J_6$ formed a new robot model and was substituted into the algorithm. Similarly, $J_2 \parallel J_3 \perp J_{4,a} \perp J_{5,a,d}$ was solved using basic problem 3.2.2 and θ_2 and θ_3 were subsequently obtained. After the reduction in the DH parameters and the transformational matrices, ${}^4_6T$ and the robot model $J_{4,a} \perp J_{5,a,d} \perp J_6$ were substituted into the algorithm to solve the rational component in the second type of sub-problem. All joint angles were solved at that time.

Consequently, the initial position was $\Theta = \begin{bmatrix} 25.5667^\circ \\ -0.0624^\circ \\ 3.0736^\circ \\ -25.5975^\circ \\ 87.2840^\circ \\ 1.3005^\circ \end{bmatrix}$, and the end execution posture was:

$$\begin{bmatrix} 0 & 0 & -1 & 0.4521 \\ 0 & 1 & 0 & 0.0499 \\ 1 & 0 & 0 & 0.4318 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{45}$$

All joint angles were computed with the algorithm, as shown in Table 17.

Here we also used the kinematic inverse function in the Robotics Toolbox, and could also find eight groups of solutions, as shown in Table 18. It could be seen that part of the joint angles have exceeded the scope of $\begin{bmatrix} -\pi & \pi \end{bmatrix}$. The reason is that there was no uniform inverse trigonometric function selected in the Robotics Toolbox. In actual use, for example, the Beckhoff-C6920 controller and the Kollmorgen RGM motor, both use real numbers to describe the motion of the motor rather than a positive real number. Therefore, the method proposed in this paper had certain advantages from this point of view.

Table 17. Eight inversely solved joint angle sets.

No.	θ_1/°	θ_2/°	θ_3/°	θ_4/°	θ_5/°	θ_6/°
1	167.0434	89.6199	3.0711	−78.4733	166.7756	101.8339
2	167.0434	89.6199	3.0711	101.5267	−166.7756	−78.1661
3	167.0434	−179.9370	−177.6878	167.0362	92.3124	0.5341
4	167.0434	−179.9370	−177.6878	12.9638	−92.3124	−179.4659
5	**25.5654**	**−0.0630**	**3.0711**	**−25.5998**	**87.2844**	**1.3006**
6	25.5654	−0.0630	3.0711	154.4002	−87.2844	178.6994
7	25.5654	90.3801	−177.6878	−84.3875	154.2989	−83.7756
8	25.5654	90.3801	−177.6878	95.6125	−154.2989	96.2244

Table 18. Eight inversely solved joint angle sets by the Robotics Toolbox.

No.	θ_1/°	θ_2/°	θ_3/°	θ_4/°	θ_5/°	θ_6/°
1	167.0434	89.6199	3.0711	−78.4733	166.7756	101.8339
2	167.0434	89.6199	3.0711	101.5267	−166.7756	−78.1661
3	167.0434	180.0624	182.3097	−167.0320	92.3124	0.5322
4	167.0434	180.0624	182.3097	12.9638	−92.3124	−179.4659
5	25.5654	90.3801	182.3097	−84.3875	154.2989	−83.7756
6	25.5654	90.3801	182.3097	95.6086	−154.2989	96.2197
7	25.5654	−0.0624	3.0736	−25.5975	87.2840	1.3005
8	25.5654	0.0624	3.0736	154.4025	−87.2840	−178.6995

The robot was instructed to move as per the Equation (51) spiral line with a step size of 0.01:

$$\begin{cases} x = 0.3t \\ y = 0.2cos(2\pi t) - 0.2 \\ z = 0.2sin(2\pi t) \end{cases}, t \in [0, 1]. \quad (46)$$

According to formula (40), with the fifth set of solutions selected. According to the joint angle, the end position was recomputed with the motion trajectory shown in Figure 3. The joint angle was inversely solved as per the end trajectory to obtain the continuous joint position, as shown in Figure 4.

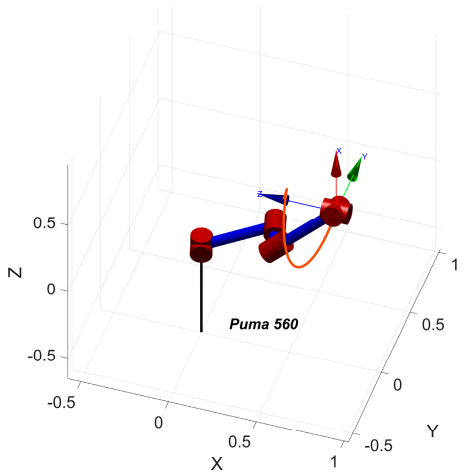


Figure 3. The motion trajectory of the Puma560 robot.

As shown in Table 17, the algorithm could solve for multiple sets of joint angles, and it could be confirmed via positive kinematics that all the joint angles were correct. In addition, in the planning experiment of the Cartesian space trajectory, the joint angle inversely solved according to the target trajectory is shown in Figure 4 and its curve changed continuously without jumps. In Figure 3, the end

trajectory was recalculated from the obtained joint angle and was consistent with the planned trajectory. This proved that the algorithm could correctly solve the inverse kinematics of the spatial trajectory.

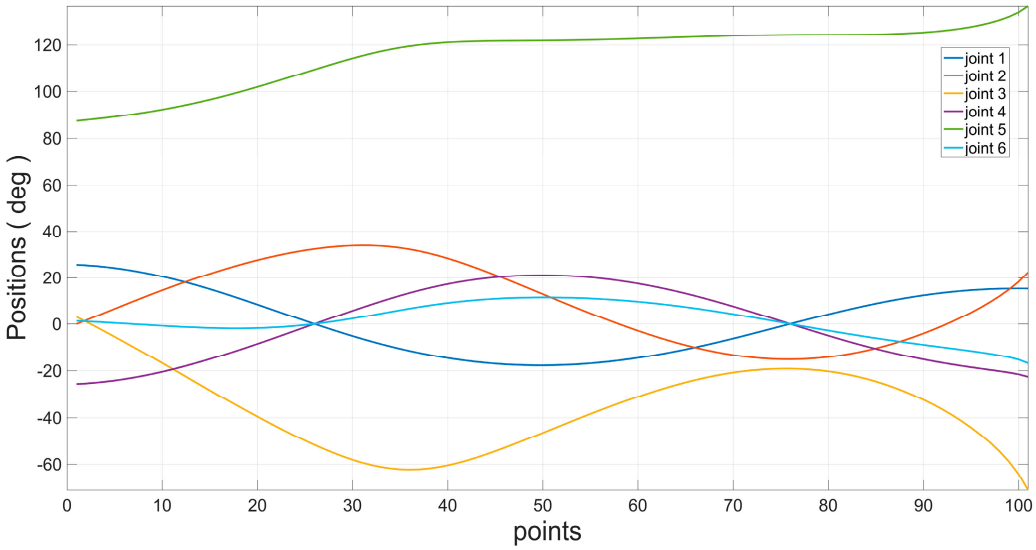


Figure 4. The joint trajectory after the inverse kinematics solution.

5.4. Real-Time Verification Experiment of the Algorithm

In the industry, the robot controller sends position information to the motor driver in a certain cycle, the shorter the cycle, the better the real-time performance. For some occasions with high precision, if the cycle is too long, a big error will occur in the outline of the end. In addition, if the desired end trajectory is constantly changing, the controller is also required to be able to react and plan quickly during the cycle. Therefore, real-time performance is an important performance indicator in the robot controlling field. In order to improve the closed-loop period of the system, in addition to the purchase of more powerful hardware devices, the more important factor is the operation speed of algorithm. Therefore, this paper designed a verification experiment whose real-time performance was closed to prove that the algorithm proposed in this paper was not only accurate but also efficient and fast.

The six-DOF KingKong robot built with a Beckhoff-C6920 controller and a Kollmorgen RGM motor was used in the experiment testing the correctness and real-time performance of the algorithm. The DH parameters of the robot are listed in Table 3. The C6920 controller in Figure 5. The C6920 controller used a 32-bit Windows 7 operating system and was equipped with an Intel Core i5 processor.

Figure 5. The C6920 controller.

Figure 6 displays the control framework for the entire system. When the controller received the order of motion, the critical parameters of the locus equation were computed according to the order.

Subsequently, the position point at the next time point was planned in the Cartesian spatial planner. Based on the position points in Cartesian space, the inverse kinematics solving method proposed here was employed to obtain the position to be reached by each motor at the next time point. The closed-loop circle of the motor controller was 2 ms. The controller sent the motion order to the RGM motor via CanOpen. CanOpen is a high-level communication protocol based on the Controller Area Network. It is often used in embedded systems and is a type of field bus commonly used in industrial control.

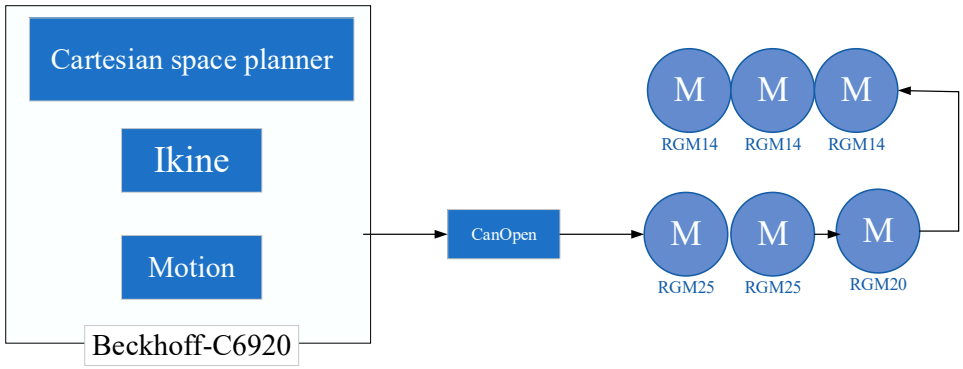


Figure 6. Control structure of the system.

The number of vertical joints was judged according to the KingKong DH model. Then, the algorithm assessed the first type of sub-problem. After joint decoupling was performed for J_6, $J_1 \perp J_2 \parallel J_3 \parallel J_4 \perp J_{5,a}$ was able to solve for θ_1 using basic problem 3.2.1. After θ_1 was obtained, ${}^1_6T$ and $J_2 \parallel J_3 \parallel J_4 \perp J_{5,a} \perp J_6$ were obtained using the forward kinematics formula. At the time, the robot belonged to the second type of sub-problem and we could directly solve for $\theta_2 + \theta_3 + \theta_4, \theta_5$ and θ_6. Subsequently, basic problem 3.4.2 was used to solve for θ_2 and $\theta_2 + \theta_3$ and to obtain θ_3 and θ_4.

In this experiment, the robot would be commanded to move according to the specified spatial trajectory in the real environment, and the acceleration process of the motor must be considered. A common acceleration planning method is a seven-segment S curve. This method uses the square wave Jerk curve as the input of the system, and the acceleration, velocity, and position quantities obtained through the integrating of time, as is shown Figure 7.

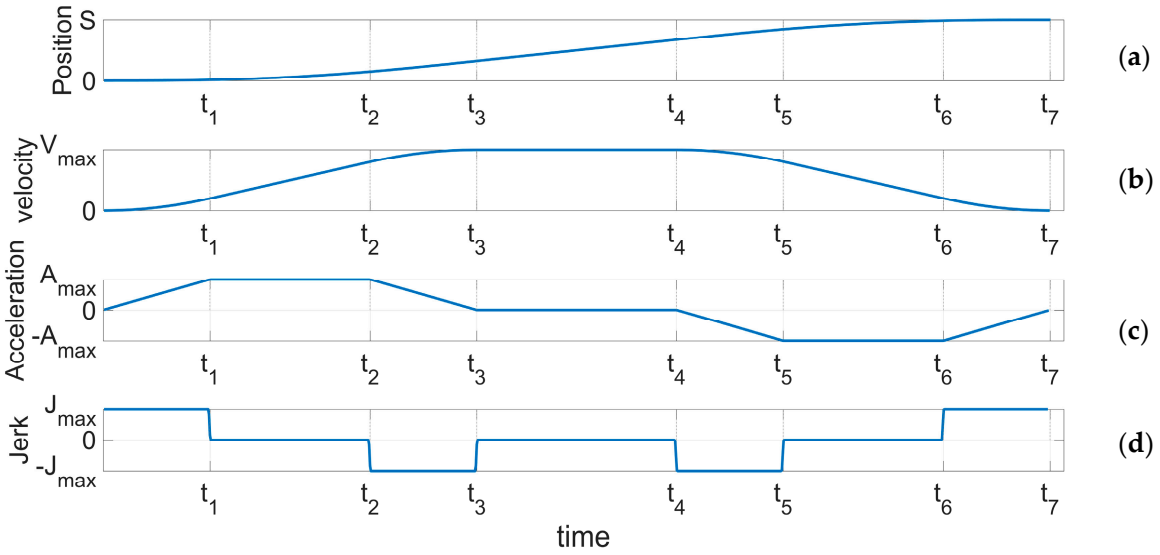


Figure 7. (**a**) Displacement of the seven-segment S curve over time. (**b**) Velocity of the seven-segment S curve over time. (**c**) Acceleration of the seven-segment S curve over time. (**d**) Jerk of seven-segment S curve over time.

It can be seen from the figure that the acceleration curve of the seven-segment S-curve constructed by square wave Jerk input is a linear piecewise function. Although it is continuous, but there are a finite number of non-conductible breakpoints. If the acceleration can be made to have higher order differentiable properties, the robot movement can be smoother.

In the Descartes spatial planner of the controller, the 15-segment S-curve spatial trajectory planning method was used. When the objective position, S, and the maximum values of the curve, $\dot{S}$, $\ddot{S}$, and $\dddot{S}$, are given, this method can provide a smooth curve and the speed utilization rate can approximate the optimal solution. In 15-segment S planning, the jerk input uses the sine piecewise function Equation (52) to replace the jerk square wave input in the traditional seven-segment S curve. A schematic diagram for the 15-segment S curve is shown in Figure 8.

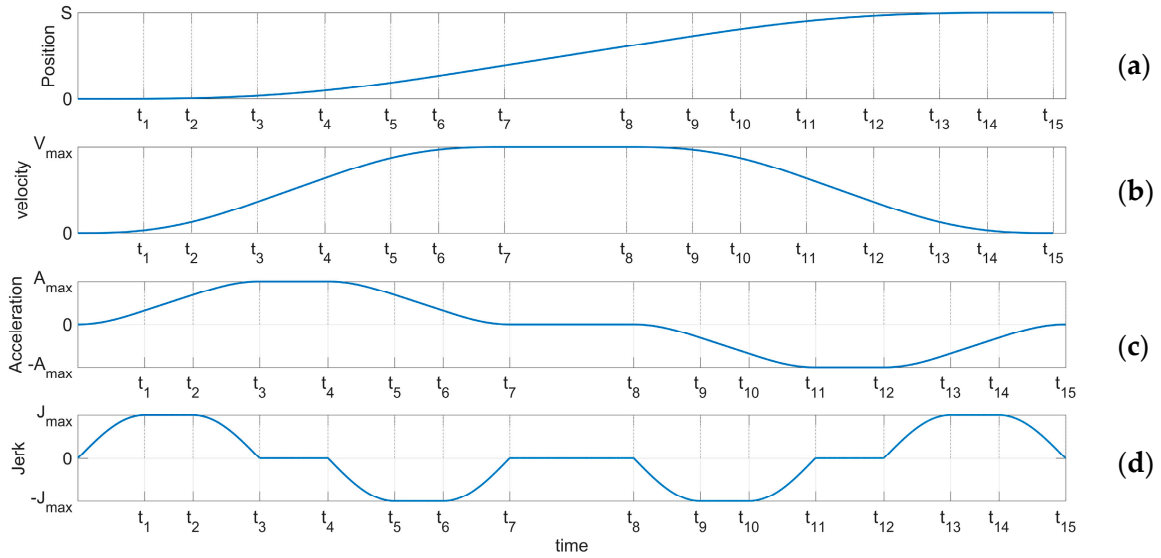


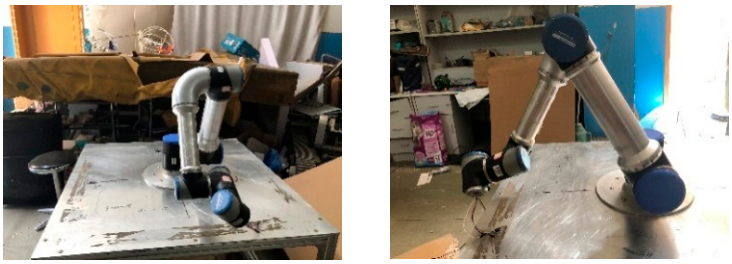
Figure 8. (**a**) Displacement of the 15-segment S curve over time. (**b**) Velocity of the 15-segment S curve over time. (**c**) Acceleration of the 15-segment S curve over time. (**d**) Jerk of 15-segment S curve over time.

Compared with the traditional seven-segment S curve, the 15-segment curve proposed in this paper was constructed by sinusoidal function. We know that sinusoidal functions have infinite differentiable properties. Therefore, the acceleration curve of the 15-segment S curve also has infinite differentiable properties. The curve obtained by this planning method is smoother.

$$h_n(t) = \begin{cases} sin\frac{\pi}{2}nt & x \in \left[0, \frac{1}{n}\right] \\ 1 & x \in \left[\frac{1}{n}, \frac{1}{n} + m\right] \\ cos\frac{\pi}{2}n(t-m) & x \in \left[\frac{1}{n} + m, \frac{2}{n} + m\right] \end{cases} . \tag{47}$$

In Equation (47), n is an adjustable parameter to adjust the rising speed of the jerk curve and m depends on the maximum values of $\ddot{S}$ and $\dddot{S}$.

The initial position of the KingKong robot was $\Theta = \begin{bmatrix} 0° \\ -60° \\ 120° \\ -135° \\ -45° \\ -45° \end{bmatrix}$, as shown in Figure 9.



(**a**) (**b**)

Figure 9. (**a**) Robot in its initial position from the front. (**b**) Robot in its initial position from the side.

At that time, based on the forward kinematics computation, the end posture was:

$$\begin{bmatrix} -0.5536 & 0.8124 & 0.1830 & -0.4961 \\ 0.5000 & 0.5000 & -0.7071 & -0.1330 \\ -0.6660 & -0.3000 & -0.6830 & 0.0895 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{48}$$

The algorithm was used to obtain the inverse solution and eight sets of joint angles, as shown in Table 19. According to formula (40) the fourth set of solutions was selected and applied in the motion control logic.

Table 19. Eight sets of reversely solved joint angles.

No.	θ_1/°	θ_2/°	θ_3/°	θ_4/°	θ_5/°	θ_6/°
1	0	38.3143	−80.0936	146.7793	45.0000	135.0000
2	0	−35.7525	80.0936	60.6588	45.0000	135.0000
3	0	47.6172	−120.0000	−2.6172	−45.0000	−45.0000
4	**0**	**−60.0000**	**120.0000**	**−135.0000**	**−45.0000**	**−45.0000**
5	−159.2347	−125.5572	−120.8932	−16.9043	136.5570	−15.1304
6	−159.2347	126.1566	120.8932	−150.4045	136.5570	−15.1304
7	−159.2347	−140.7508	−79.3110	136.7071	−136.5570	164.8696
8	−159.2347	145.8820	79.3110	51.4522	−136.5570	164.8696

The spatial planning was as follows. The experiment used a spiral line as the expected end trajectory. The parameter equation of the end trajectory spiral line is:

$$\begin{cases} x = 0.15\cos(2\pi t) - 0.15 \\ y = 0.15\sin(\pi t) \\ z = 0.05t \end{cases}, t \in [0, 5]. \tag{49}$$

First, according to Equation (50), the arc length of the spiral line is:

$$\updownarrow = \int_0^5 \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2}\, dt = 2.3694 \text{ m}. \tag{50}$$

Given $\dot{S}_{max} = 0.125$ m/s, $\ddot{S}_{max} = 0.025$ m/s, $\dddot{S}_{max} = 0.01$ m/s, and $n = 1$, the 15-segment S curve was used to plan the end arc length. The total time consumed was 28.1821 s. The variation curve at each stage of the trajectory is shown in Figure 10.

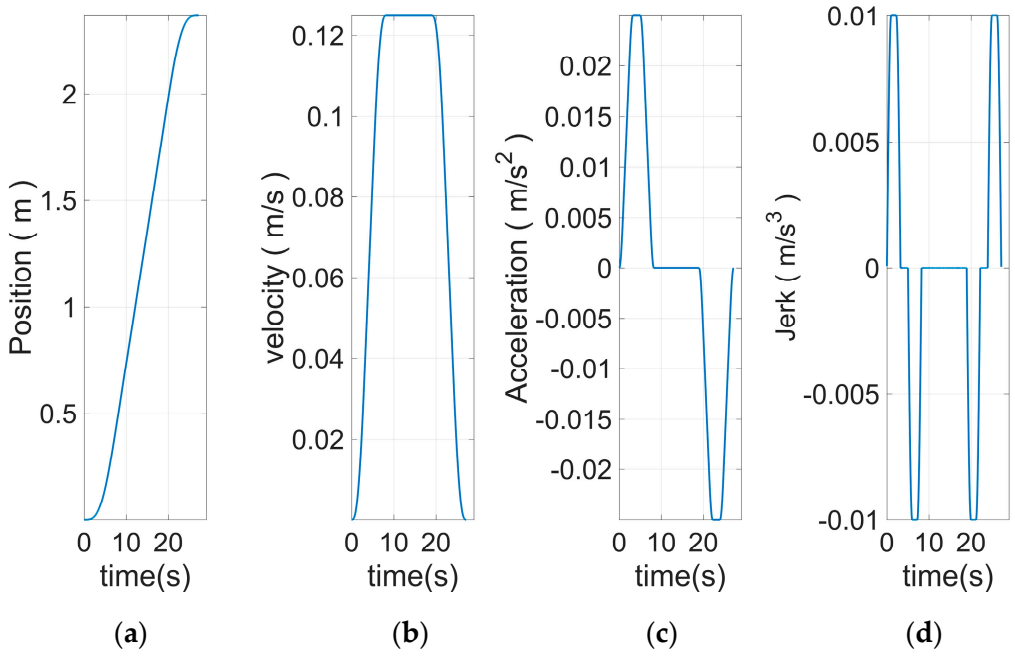


Figure 10. (**a**) Displacement of the spatial trajectory over time. (**b**) Velocity of the spatial trajectory over time. (**c**) Acceleration of the spatial trajectory over time. (**d**) Jerk of the spatial trajectory over time.

The arc length obtained according to Equation (50) was used to rewrite the parametric equation:

$$\begin{cases} x = 0.15\cos\left(\frac{l}{0.1581}\right) - 0.15 \\ y = 0.15\sin\left(\frac{l}{0.1581}\right) \\ z = 0.1581\, l \end{cases}, l \in [0, \updownarrow]. \tag{51}$$

In summary, the position curve in Cartesian space and its various derivatives could be obtained, together with the planned position curves of each joint and their various derivatives, as shown in Figures 11–16.

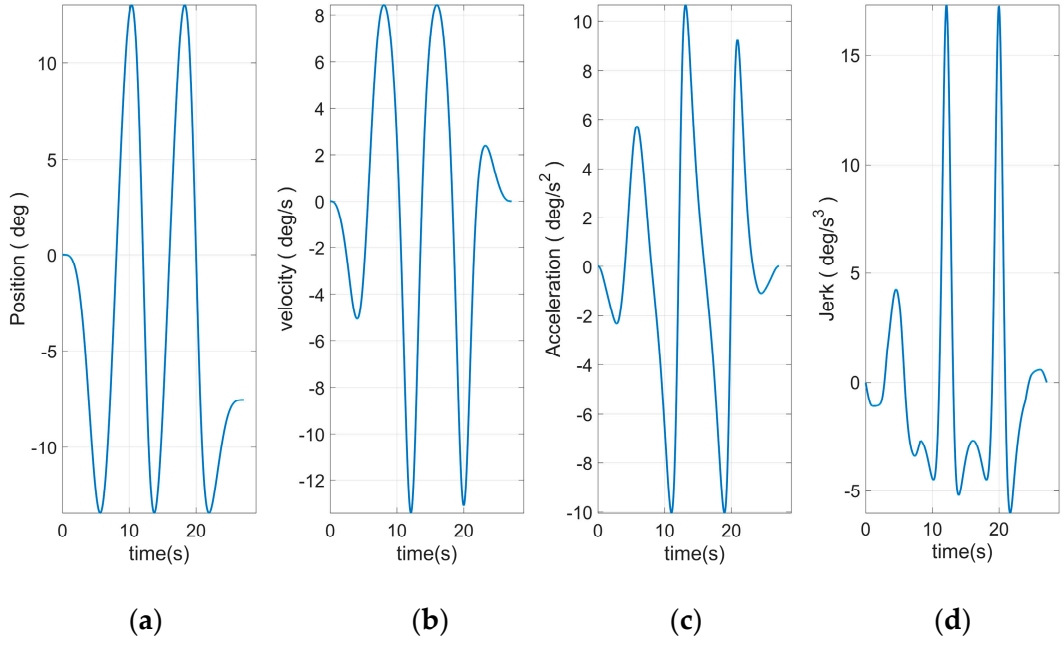


Figure 11. (**a**) Displacement of axis I over time. (**b**) Velocity of axis I over time. (**c**) Acceleration of axis I over time. (**d**) Jerk of axis I over time.

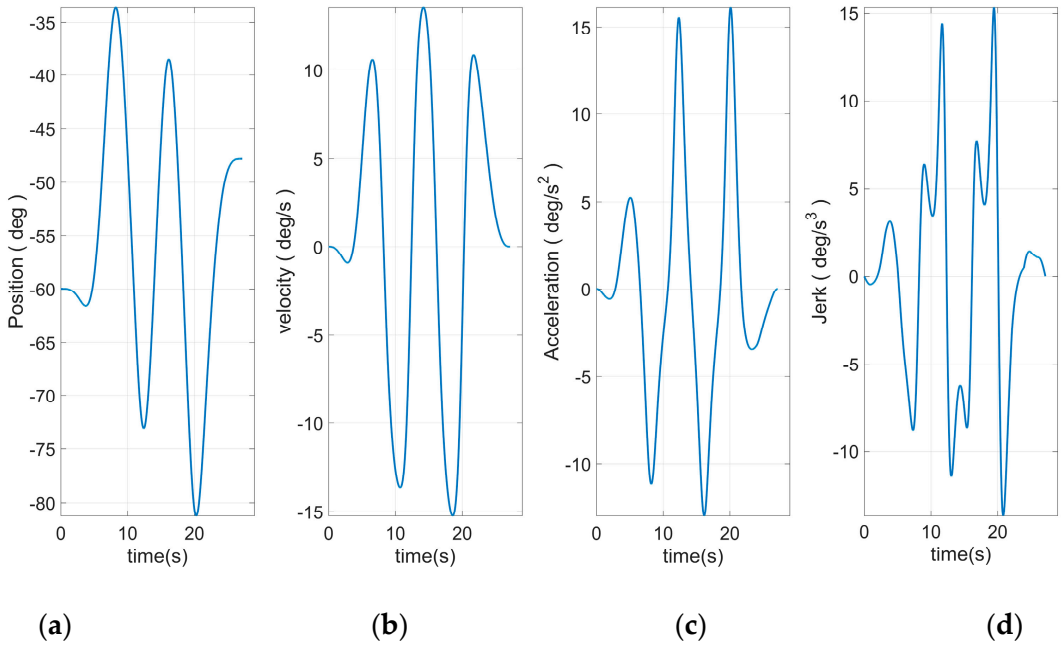


Figure 12. (**a**) Displacement of axis II over time. (**b**) Velocity of axis II over time. (**c**) Acceleration of axis II over time. (**d**) Jerk of axis II over time.

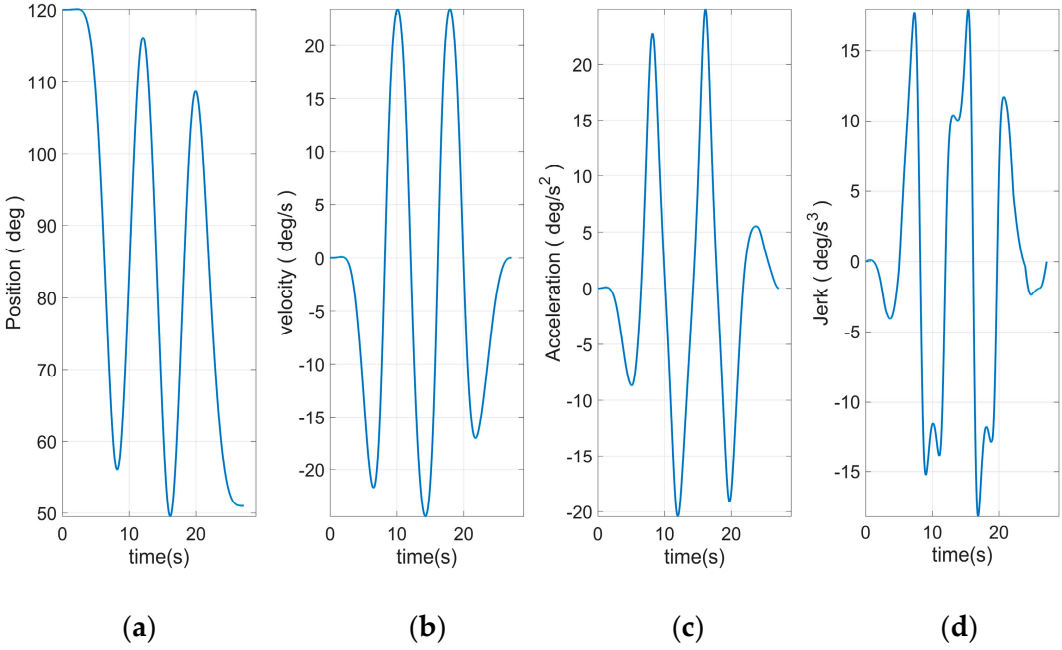


Figure 13. (**a**) Displacement of axis III over time. (**b**) Velocity of axis III over time. (**c**) Acceleration of axis III over time. (**d**) Jerk of axis III over time.

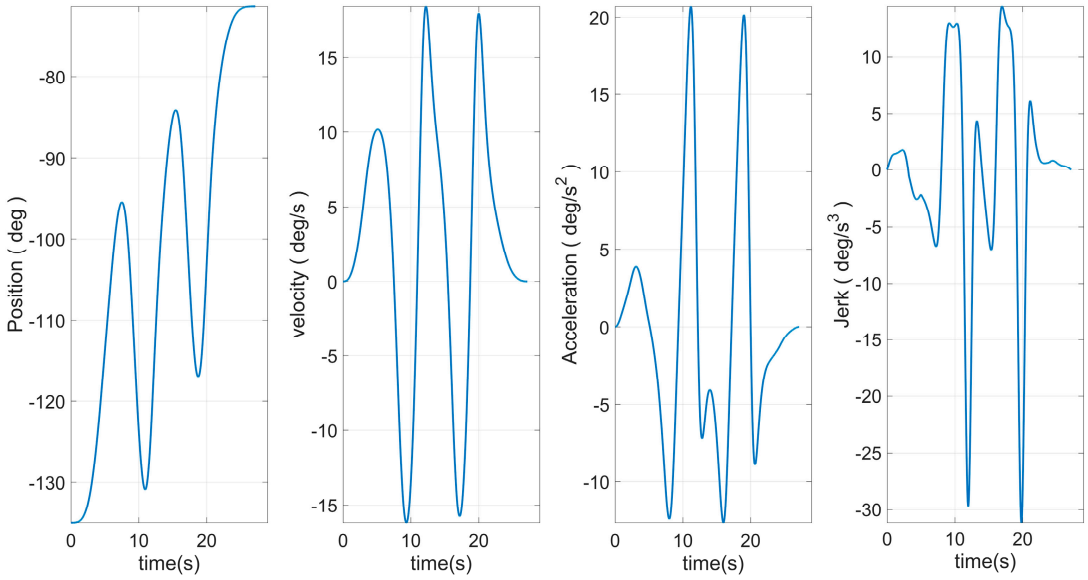


Figure 14. (**a**) Displacement of axis IV over time. (**b**) Velocity of axis IV over time. (**c**) Acceleration of axis IV over time. (**d**) Jerk of axis IV over time.

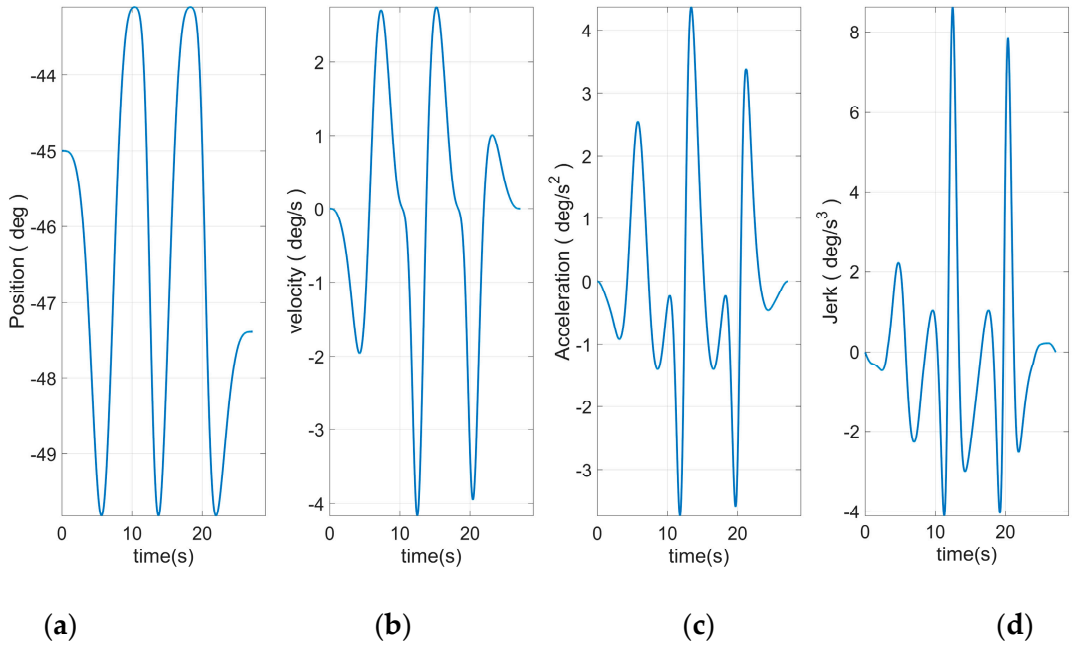
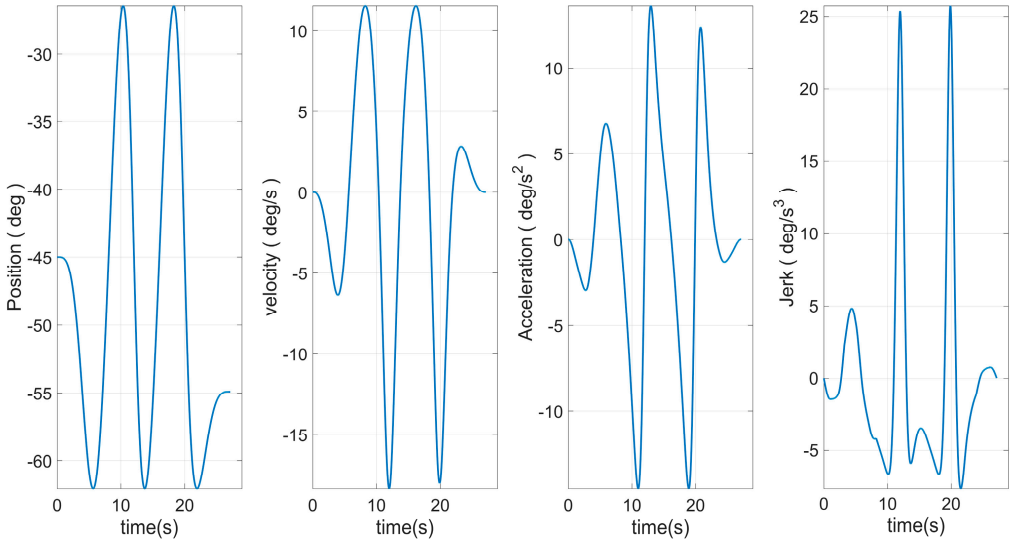


Figure 15. (**a**) Displacement of axis V over time. (**b**) Velocity of axis V over time. (**c**) Acceleration of axis V over time. (**d**) Jerk of axis V over time.



(**a**) (**b**) (**c**) (**d**)

Figure 16. (**a**) Displacement of axis VI over time. (**b**) Velocity of axis VI over time. (**c**) Acceleration of axis VI over time. (**d**) Jerk of axis VI over time.

Figure 17 shows the position after the robot has finished its run. The curve of the actual motion of the robot in Cartesian space was calculated from the forward kinematics and is shown together with the planned spatial curve in Figure 18. After the run was completed, the actual motion curve of each joint was acquired from the controller, as shown in Figure 19.

(**a**) (**b**)

Figure 17. (**a**) Posture of the robot after completion of its run from the front. (**b**) Posture of the robot after completion of its run from the side.

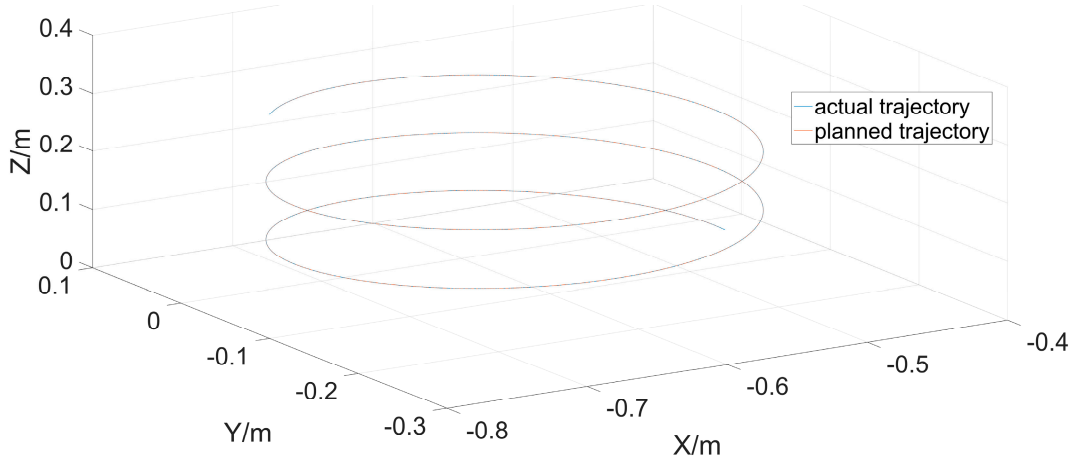


Figure 18. Robot end trajectory.

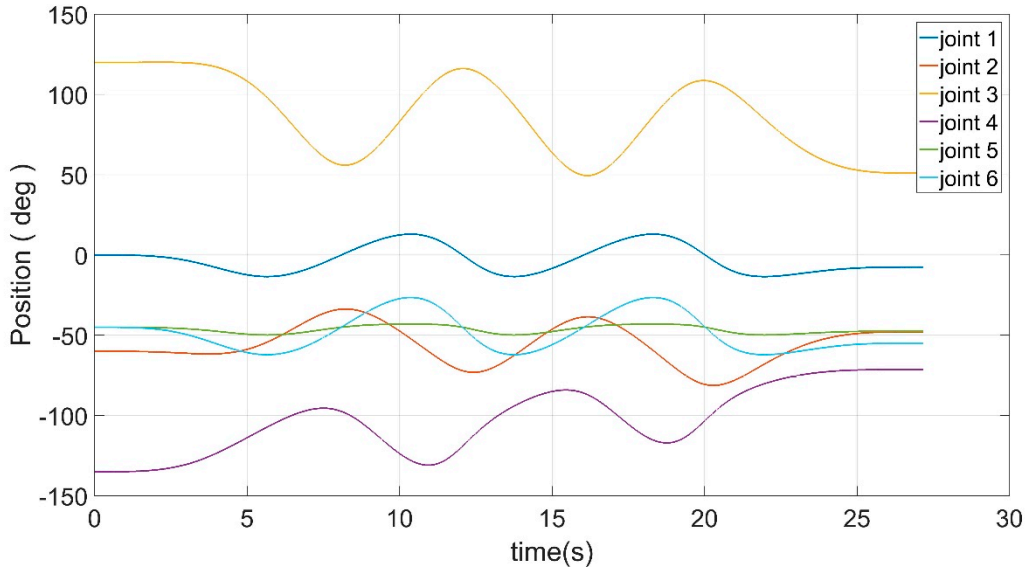


Figure 19. The joint trajectory after the inverse kinematics solution.

In the fourth experiment, the Beckhoff controller, which is commonly used in the industry, was used and the general algorithm was implemented in the controller. The robot controller sends motion commands to the drive at a certain period. This requires the controller to complete the kinematic inverse solution within a specified amount of time to ensure the continuity. A shorter period results in a higher real-time performance. The period used in this article was 2 ms. The 2 ms cycle is sufficient to meet the general accuracy requirements. For example, in Ref. [5] the medical robot can accept a closed loop period of 20 ms. In other words, the proposed algorithm in this paper could achieve a closed-loop period higher than industrial demand in equipment commonly used in the industry. The fast closed-loop period means that the algorithm has higher computational efficiency.

Since our algorithm could decompose complex inverse motion problems into several sub-problems and solve them one by one, all joint angles could be solved accurately and quickly. The final experimental results show that the system could complete the inverse solution of the kinematics within the specified period. This allows the algorithm to run stably in situations with real-time requirements.

Figure 8 shows the end curve planned to use 15 S-curve segments. The apparent rise and fall process in the jerk curve could be seen. This allows the entire curve to have a smoother transition at start–stop. Based on Equation (46) and the plan in Figure 10, the angular curve of each joint could be obtained, as shown in Figures 11–16.

It can be seen from Figures 11–16 that all joint motion curves, as well as the various derivative curves could be ensured to start from 0 during the start-up phase smoothly. The continuous and smooth was kept in the whole moving process. Moreover, the curve was smoothly reduced to zero during the stop phase. Especially the fourth derivative had kept the continuous and smooth fluctuations, which was the effect that could not be realized through traditional planning method. This paper introduced the joint curve obtained through using the traditional seven-segment planning there to facilitate the comparison, as shown in Figures 20–25. It can be clearly seen that the Jerk curve of each joint had an obvious jump phenomenon during the start–stop phase. Meanwhile, thumbing changes occurred in the operation process. These problems would cause the motor to operate unstably in the end.

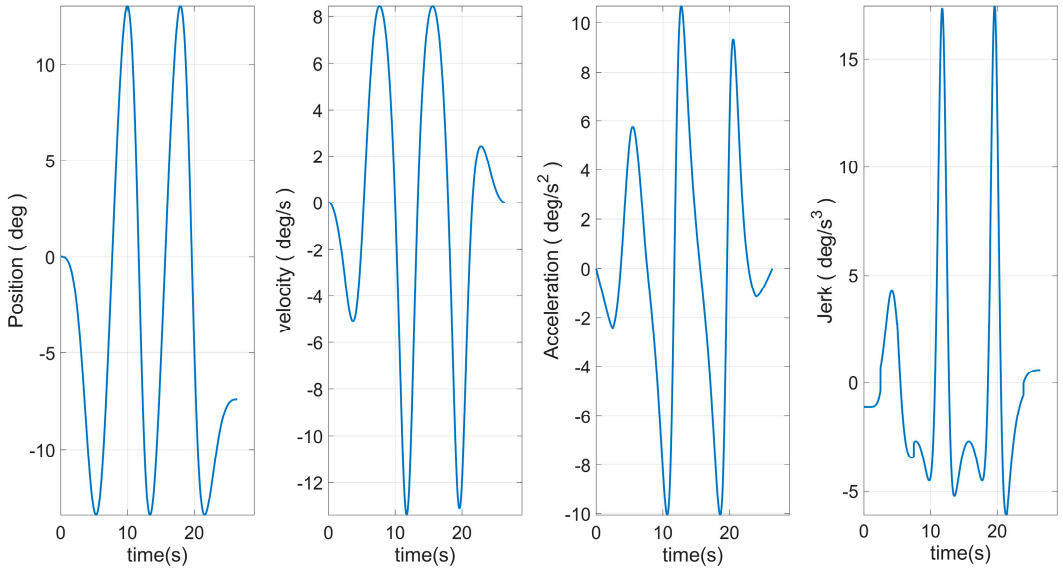


Figure 20. (**a**) Displacement of axis I over time by using seven segments S-curve. (**b**) Velocity of axis I over time by using seven segments S-curve. (**c**) Acceleration of axis I over time by using seven segments S-curve. (**d**) Jerk of axis I over time by using seven segments S-curve.

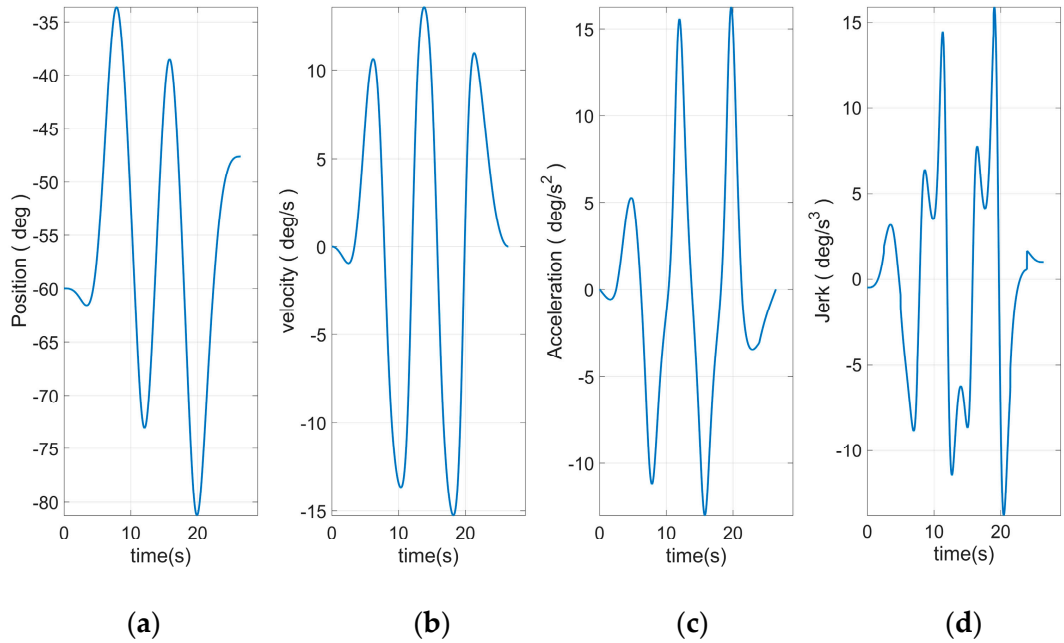


Figure 21. (**a**) Displacement of axis II over time by using seven segments S-curve. (**b**) Velocity of axis II over time by using seven segments S-curve. (**c**) Acceleration of axis II over time by using seven segments S-curve. (**d**) Jerk of axis II over time by using seven segments S-curve.

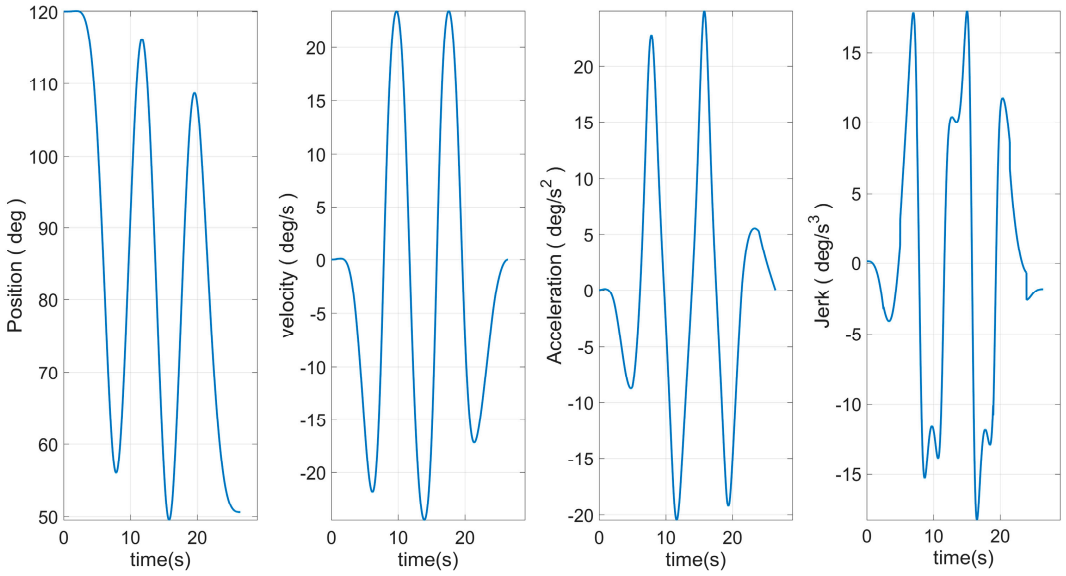


Figure 22. (**a**) Displacement of axis III over time by using seven segments S-curve. (**b**) Velocity of axis III over time by using seven segments S-curve. (**c**) Acceleration of axis III over time by using seven segments S-curve. (**d**) Jerk of axis III over time by using seven segments S-curve.

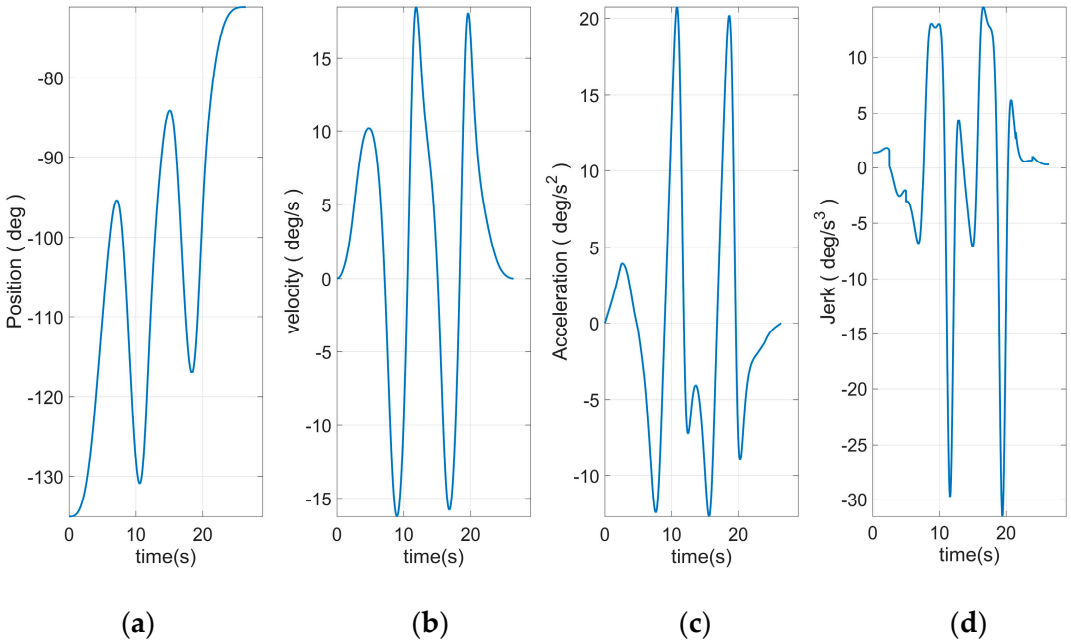


Figure 23. (**a**) Displacement of axis IV over time by using seven segments S-curve. (**b**) Velocity of axis IV over time by using seven segments S-curve. (**c**) Acceleration of axis IV over time by using seven segments S-curve. (**d**) Jerk of axis IV over time by using seven segments S-curve.

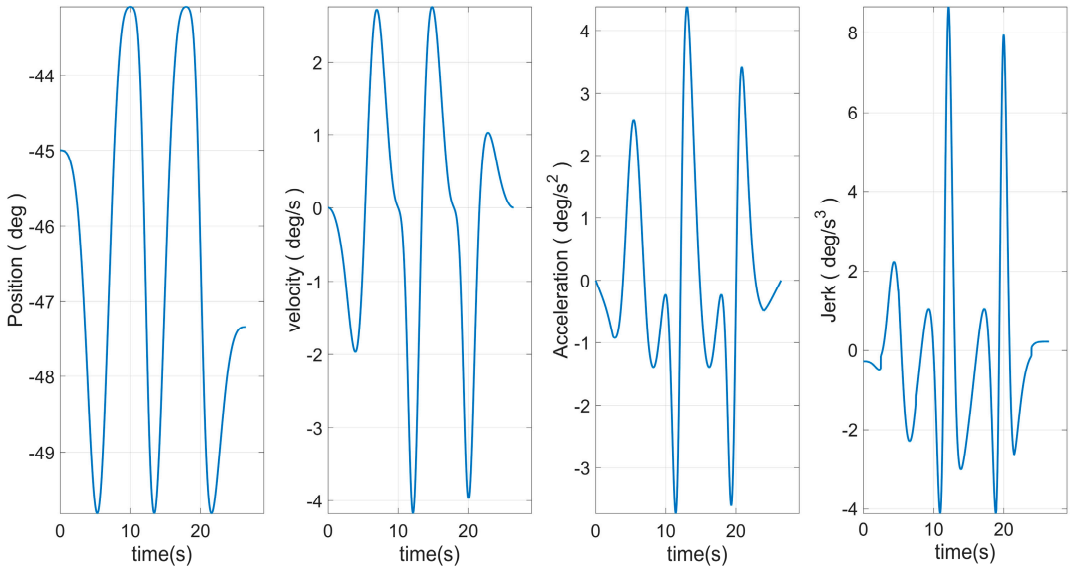


Figure 24. (**a**) Displacement of axis V over time by using seven segments S-curve. (**b**) Velocity of axis V over time by using seven segments S-curve. (**c**) Acceleration of axis V over time by using seven segments S-curve. (**d**) Jerk of axis V over time by using seven segments S-curve.

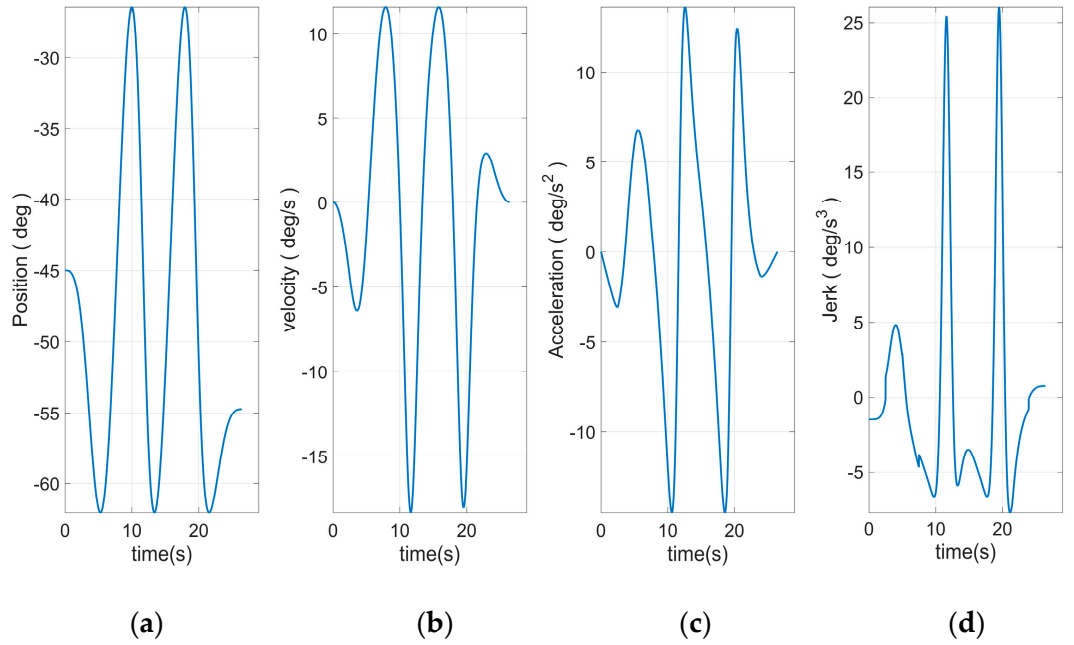


Figure 25. (**a**) Displacement of axis VI over time by using seven segments S-curve. (**b**) Velocity of axis VI over time by using seven segments S-curve. (**c**) Acceleration of axis VI over time by using seven segments S-curve. (**d**) Jerk of axis VI over time by using seven segments S-curve.

Figure 18 shows the actual run in Cartesian space. It could be seen that the actual motion trajectory coincided with the planned motion trajectory. The mean square error of the two tracks was calculated to be 7.8265×10^{-10} m. This data shows that the error between the actual trajectory and the target trajectory was extremely small. This shows that the inverse solution of the robot had higher accuracy.

6. Conclusions

Based on the DH model, this study proposed a universal algorithm for finding an inverse kinematics closed-form solution. This algorithm divided the inverse kinematics problem related to robots with a closed-form solution into three sub-problems assuming that the algebraic equation had a formula solution. The solvability conditions for the three types of sub-problems were analyzed to derive a formula solution. If a serial robot can be described with these three types of sub-problems, the robot will definitely have a closed-form solution. Meanwhile, the formula solution based on such a problem could quickly solve for all the joint angles. This method was not only applicable to six-DOF robots with a closed-form solution but also to low-DOF robots with the same solution form. In addition, establishing an algorithm based on the DH model provides a concise and efficient tool for selecting sub-problems and a real-time inverse kinematics solution for the motion controller of a serial robot. To verify the correctness and real-time performance of the algorithm, we presented four experimental designs. The first three experiments were conducted on MATLAB with a series of robot to verify the completeness, universality, and continuity of this algorithm.

In the first experiment, the completeness of the algorithm was verified. The experiment used a low-degree-of-freedom robot with closed inverse solutions to prove that the algorithm could solve its closed inverse solution, and the algorithm would be terminated for the robot without closed inverse solutions to avoid infinite loop. This was used to prove the completeness of the algorithm. In the second experiment, two uncommon robots were constructed through using some basic problems, but the closed inverse solution could still be solved. Therefore, it could be known that the closed inverse solution of the series robots constructed by basic problems could be solved. Meanwhile, for these robots whose structure was not changed but the link parameter was changed, the closed inverse solution of them could also be solved correctly, which indicates that the algorithm had certain versatility in solving the closed inverse solution problem. In the third experiment, the common Puma560 robot was used as the experimental object. In the experiment, a spatial curve was planned. The correct joint angle sequence was obtained and the changes of the joint angle curve were continuously without jumping after been inversely solved by the algorithm. Which indicates that the algorithm could guarantee the continuity of its mapping on the inverse solution problem.

The fourth experiment was conducted on a six-DOF robotic platform built using a Beckhoff-C6920 controller and an RGM motor to verify the real-time performance of the algorithm using the above hardware platform. The spatial and joint positions were computed with a close-loop cycle of 2 ms and were transmitted to the motor via CanOpen. The results demonstrated the ability of the algorithm to solve the three sub-problems within a specified cycle and to compute the appropriate joint angles. Moreover, the curve was continuous. This paper used a completely new approach to study the inverse kinematics of serial robots. In this study, the conditions for the existence of the closed-form inverse kinematic solution of a serial robot were completed and the basic theory of robotics was perfected. Based on this method, a general closed-form inverse kinematics algorithm based on the DH model was implemented, which had not been done in previous studies. Importantly, this method is a general-purpose algorithm that can be used in industrial fields in real time. This increases the versatility of the universal serial robot motion controller.

However, the algorithm still has shortcomings. First, the algorithm only targets serial robots and the moving joints are not included in the solution model. This limits the scope of application of the algorithm, due to its inability to solve the cases of SCARA robots or other parallel robots. In addition, the algorithm cannot solve the problem when the robot passes through a singular point; singular points are avoided via restrictions implemented in the software. Moreover, because the algorithm is a purely analytical method, it is difficult to combine with other numerical methods to solve the problem of singular points.

Author Contributions: Conceptualization, W.S.; methodology, W.S. and L.X.; software, W.S.; writing—original draft preparation, W.S.; writing—review and editing, W.S., L.X., H.B., L.Q.; supervision, L.Q., funding acquisition, L.X.

Funding: Supported by: National Key R&D Program of China (2016YFC0803000, 2016YFC0803005).

Conflicts of Interest: The authors declare there is no conflict of interest regarding the publication of this paper.

References

1. Xiao, W.; Strauß, H.; Loohß, T.; Hoffmeister, H.W.; Hesselbach, J. Closed-form inverse kinematics of 6R milling robot with singularity avoidance. *Prod. Eng.* **2011**, *5*, 103–110. [CrossRef]
2. Wang, L.; Fallavollita, P.; Zou, R.; Chen, X.; Weidert, S.; Navab, N. Closed-form inverse kinematics for interventional C-arm X-ray imaging with six degrees of freedom: Modeling and application. *IEEE Trans. Med. Imaging* **2012**, *31*, 1086–1099. [CrossRef] [PubMed]
3. Khan, A.; Cheng, X.; Zhang, X.; Quan, W.L. Closed form inverse kinematics solution for 6-DOF underwater manipulator. In Proceedings of the 2015 International Conference on Fluid Power and Mechatronics (FPM), Harbin, China, 5–7 August 2015; pp. 1171–1176.
4. Bunathuek, N.; Laksanacharoen, P. Inverse kinematics analysis of the three-legged reconfigurable spherical robot II. In Proceedings of the 2017 3rd International Conference on Control, Automation and Robotics (ICCAR), Nagoya, Japan, 22–24 April 2017; pp. 31–35.
5. Bai, L; Yang, J.; Chen, X.; Jiang, P.; Liu, F.; Zheng, F.; Sun, Y. Solving the Time-Varying Inverse Kinematics Problem for the Da Vinci Surgical Robot. *Appl. Sci.* **2019**, *9*, 546. [CrossRef]
6. Hartenberg, R.S. A Kinematic Notation for Lower-Pair Mechanism Based on Matrices. *Trans. ASME J. Appl. Mech.* **1955**, *22*, 215–221.
7. Raghaven, M.; Roth, B. Kinematic analysis of the 6R manipulator of general geometry. In Proceedings of the International Symposium on Robotics Research, Hanoi, Vietnam, 6–10 October 2019; pp. 263–269.
8. Penrose, R. On Best Approximate Solutions of Linear Matrix Equations. *Proc. Camb. Philos. Soc.* **1956**, *52*, 17. [CrossRef]
9. Siciliano, B. A Closed-loop Inverse Kinematic Scheme for On-line Joint-based Robot Control. *Robotica* **1990**, *8*, 231–243. [CrossRef]
10. Wampler, C.W.I. Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods. *IEEE Trans.Syst. Man Cybern.* **1986**, *16*, 93–101. [CrossRef]
11. Kelemen, M.; Virgala, I.; Lipták, T.; Miková, Ľ.; Filakovský, F.; Bulej, V. A Novel Approach for a Inverse Kinematics Solution of a Redundant Manipulator. *Appl. Sci.* **2018**, *8*, 2229. [CrossRef]
12. Reiter, A.; Muller, A.; Gattringer, H. On Higher Order Inverse Kinematics Methods in Time-Optimal Trajectory Planning for Kinematically Redundant Manipulators. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1681–1690. [CrossRef]
13. Feng, Y.; Wang, Y.; Wei, S. A novel hybrid electromagnetism-like algorithm for solving the inverse kinematics of robot. *Ind. Robot* **2011**, *38*, 429–440. [CrossRef]
14. Yin, F.; Wang, Y.N.; Wei, S.N. Inverse Kinematic Solution for Robot Manipulator Based on Electromagnetism-like and Modified DFP Algorithms. *Acta Autom. Sin.* **2011**, *37*, 74–82. [CrossRef]
15. Paul, R.P.; Shimano, B. Kinematic Control Equations for Simple Manipulators. In Proceedings of the IEEE Conference on Decision and Control Including the 17th Symposium on Adaptive Processes, San Diego, CA, USA, 10–12 January 1979.
16. Pieper, D.L. The Kinematics of Manipulators under Computer Control. Ph.D. Thesis, Stanford University, Stanford, CA, USA, 1968.
17. John, J.C. Inverse kinematics of the manipulator. In *Introduction to Robotics: Mechanics and Control*; Pearson: London, UK, 2005; pp. 87–88.
18. Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. *Robotics: Modelling, Planning and Control*, 2nd ed.; Springer Publishing Company, Incorporated: London, UK, 2010; pp. 76–82.
19. Cui, H.-X.; Feng, K.; Li, H.-L.; Han, J.-H. Singularity avoidance of 6R decoupled manipulator using improved Gaussian distribution damped reciprocal algorithm. *Ind. Robot* **2017**, *44*, 324–332. [CrossRef]
20. Murray, R.M.; Sastry, S.S.; Li, Z. *A Mathematical Introduction to Robotic Manipulation*; CRC Press, Inc.: Florida, FL, USA, 1994; p. 292.
21. Kahan, W. *Lectures on Computational Aspects of Geometry*; University of California: Berkeley, CA, USA, 1983.
22. Paden, B. Kinematics and Control Robot Manipulators. Ph.D. Thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA, 1986. Available online: https://10.1109/ACSSC.1985.671441 (accessed on 8 October 2018). [CrossRef]

23. Wang, H.; Lu, X.; Cui, W.; Zhang, Z.; Li, Y.; Sheng, C. General inverse solution of six-degrees-of freedom serial robots based on the product of exponentials model. *Assem. Autom.* **2018**, *38*, 361–367. [CrossRef]
24. An, H.S.; Seo, T.W.; Lee, J.W. Generalized solution for a sub-problem of inverse kinematics based on product of exponential formula. *J. Mech. Sci. Technol.* **2018**, *32*, 2299–2307. [CrossRef]
25. Corke, P.I.J.I.R.; Magazine, A. A robotics toolbox for MATLAB. *IEEE Robot. Autom. Mag.* **1996**, *3*, 24–32. [CrossRef]
26. Jazar, R.N. Inverse kinematics. In *Theory of Applied Robotics*; Springer: Berkeley, CA, USA, 2010; pp. 222–234.