International Core Journal of Engineering Volume 11 Issue 8, 2025
ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

Kinematic Analysis and Trajectory Planning of Industrial Robots

Zechun Zhao*

Fudan University, Shanghai, 200433, China

*Corresponding author: 19300290043@fudan.edu.com

Abstract

As industrial automation progresses, industrial robots find extensive application across diverse sectors. However, due to the diversification of task requirements, higher and higher demands are put forward for the operating efficiency, impact resistance, and adaptability of industrial robots. Pertaining to this matter, the study focuses on the M - 710iC/50 multi-functional robot and develops its joint motion path using a 3 - 5 - 3 polynomial. The main research contents are as follows: Initially, the conventional D - H technique is employed to set up the robot's link coordinate system, followed by deriving the robot's forward kinematics equation based on the uniform transformation interplay among these coordinate systems. Analytical techniques are employed to compute the inverse kinematics, determining the angles at each robot joint, while the Robotics Toolbox confirms both the forward and reverse kinematics. Next, an examination is conducted on the industrial robot's algorithm for planning trajectories. An in-depth analysis of the polynomial interpolation algorithm is conducted to plan paths in combined space. Considering the shortcomings of cubic and quintic polynomial interpolation techniques, combined trajectory fitting utilizes the 3-5-3 polynomial interpolation method. Finally, the above two types of trajectory planning algorithms are implemented using MATLAB, and a comparative analysis is carried out according to the robot motion curves.

Keywords

Industrial Robots; Trajectory Planning; Kinematic Analysis; Polynomial Interpolation.

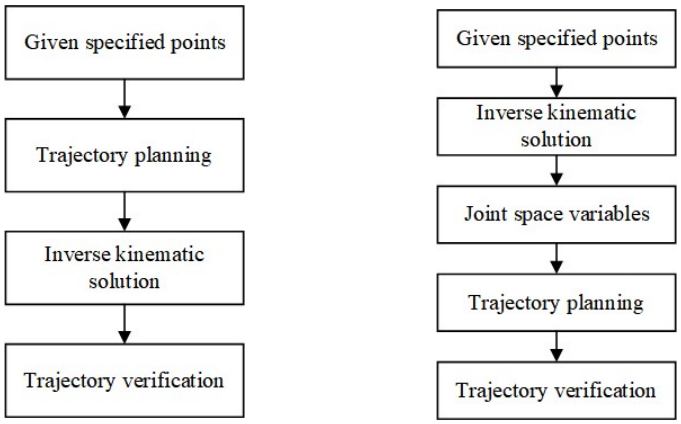
1. Introduction

A nation's economy's foundation lies in its manufacturing sector, mirroring its industrial prowess and production capacity[1]. As a key player in the manufacturing sector, China's advancement in this field significantly influences its economic expansion, technological advancements, and job market. In the "Made in China 2025" initiative[2,3], industrial robots play an important role and are significant drivers for the development of manufacturing automation. Consequently, enhancing the movement efficiency of industrial robots is a critical matter for independent domestic research and development in this field. Planning trajectories forms the cornerstone of robotic motion control in industry and is essential for optimizing trajectories. Therefore, for an industrial robot to complete its tasks with optimal motion performance, reasonable trajectory planning and optimization are particularly important.

The fundamental concept behind the trajectory planning technique is this: utilizing established path points, this method allows the industrial robot's end effector to sequentially navigate through these points, ensuring a seamless and uninterrupted path. Planning for pathways falls into two types: Cartesian space trajectory planning[4-6] and joint space trajectory planning.[7-9], based on the various planning areas involved. Utilizing the Cartesian space trajectory planning technique, the

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

trajectory of the industrial robot's end-effector can be precisely controlled, yielding easily observable results. It is usually applied to movements with specific path requirements. The method for planning trajectories in joint space directly orchestrates the joint variables, offering benefits like minimal computational effort and rapid processing speed. It is mainly applied to PTP (Point To Point) movements without specific path requirements. Figure 1 displays the flowcharts for methods for planning trajectories.



(a) Cartesian space trajectory planning (b) joint space trajectory planning

Figure 1. Flow chart of the trajectory planning method

2. Kinematics Analysis of Industrial Robots

Industrial robot kinematics form the theoretical foundation for planning and optimizing their trajectories. Robot kinematics primarily concentrates on D-H parametric modeling, conducting both forward and reverse kinematic assessments, and examining the robot's operational surroundings. This chapter focuses on the M-710iC/50, a multi-functional robot with six axes, as the subject of the research. Initially, the robot's parametric modeling is executed using the conventional D-H technique. Subsequently, utilizing the D-H parameter framework, the kinematic equations of the robot in both forward and reverse directions are deduced. Ultimately, the Robotics Toolbox serves to confirm the kinematic equations in both forward and reverse directions and to finalize the workspace examination of the industrial robot with a six-degree freedom.

2.1 D-H Modeling of Industrial Robots

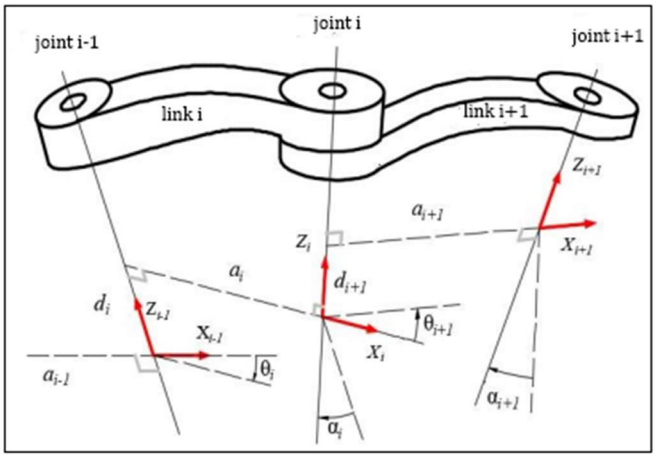


Figure 2. Diagrammatic representation for creating the coordinate frameworks of neighboring links

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

Analyzing the movement of industrial robots forms the foundational theory for designing and refining their trajectories. Establishing a consistent transformation link among neighboring links is crucial in the analytical and derivational stages of forward and inverse kinematics. Commonly, the D-H parameter approach is used to illustrate the consistent transformation patterns between adjacent links. The D-H parameter approach is categorized into the conventional D-H method(SDH)[10] and the altered D-H method(MDH)[11], based on various methods used to create coordinate systems. Figure 2 illustrates the method for setting up coordinate systems between two neighboring links.

Within the illustration, α_i denotes the angle of link twisting, a_i the length of the link, θ_i the angle of joint rotation, and d_i the offset of the joint. A pair of neighboring joints create a kinematic sequence via a connecting rod. Every joint is capable of both turning and moving. To derive the homogeneous transformation matrix T_i^{i-1} for two neighboring connecting rods, one must sequentially right-multiply the translational and rotational homogeneous transformation matrices:

$$ {}^{i-1}_{i}T = Rot(z_{i-1}, \theta_i)Trans(z_{i-1}, d_i)Trans(x_i, a_i)Rot(x_i, \alpha_i) \tag{1} $$

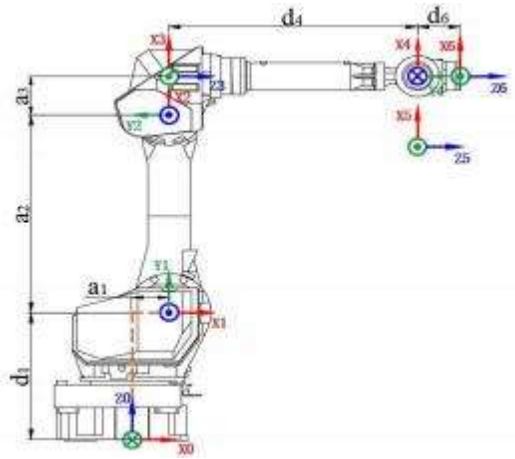
Within the equation, $Rot(z_{i-1}, \theta_i)$ symbolizes the rotation matrix for transformation, signifying angle rotation θ_i around the Zi-1 axis. The same applies to $Rot(x_i, \alpha_i)$. $Trans(z_{i-1}, d_i)$ is the translational transformation matrix, which represents a translation of distance di along the Zi-1 axis, The same applies to $Trans(x_i, a_i)$.

As can be seen from formula (1), the general formula for the uniform matrix for transformation T_i^{i-1} is:

$$ {}^{i}_{i-1}T = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i c\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

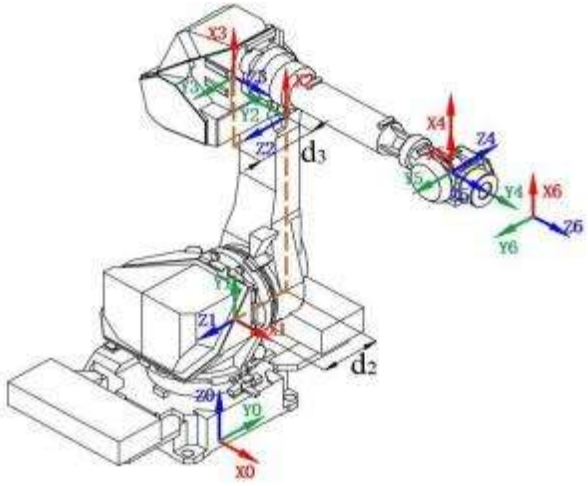
In the formula:$s\theta_i = sin\,\theta_i\,, c\theta_i = cos\,\theta_i$.The same applies hereinafter.

Taking the FANUCM-710/iC50 robot as the research object, the establishment of the robot's link coordinate system is shown in Figure 3.



(a) The front view of the M-710/iC50 robot

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018



(b) The axonometric drawing of the M-710/iC50 robot

Figure 3. Establishment of coordinate system for the M-710/iC50 robot link

Parameters D-H for the M-710iC/50 robot, following the robot D-H parameter model, are presented in Table 1.

Table 1. M-710/iC50 Standard D-H Parameters

Joint i	α_i (°)	a_i (mm)	d_i (mm)	θ_i (°)	θ_i range(°)
1	90	150	560	θ_1	-180~180
2	0	870	-180	θ_2	-90~135
3	90	170	177	θ_3	-160~280
4	90	0	1016	θ_4	-360~360
5	-90	0	0	θ_5	-125~125
6	0	0	175	θ_6	-360~360

2.2 Forward Kinematics of Industrial Robots

The key to resolving forward kinematics lies in determining the position of the industrial robot's end-effector in relation to the foundational coordinate system, using established joint angles. According to the formula, the joint pose matrices of adjacent link coordinate systems of the robot can be obtained, with a total of 6 matrices.

The robot's end-pose equation can be derived by sequentially multiplying the six homogeneous transformation matrices on the right, as per the coordinate transformation technique. The expression of the end-pose equation of the robot is:

$$ {}^{0}_{6}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3} $$

Address each component of the 4×4 matrix, derived sequentially from the formula (3), leading to the formation of forward kinematics equations for the robot that we have formulated. In summary, ${}^{0}_{6}T$ outlines the location of the robot's end-effector in comparison to the fundamental coordinate system.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

By integrating the parameters and angle θi of each joint into the equation, one can derive the arrangement matrix for the industrial robot's end-effector.

2.3 Inverse Kinematics of Industrial Robots

Reverse kinematics of an industrial robot with a liberty of six degrees involve a process of reverse computation. The process inversely computes every joint angle that aligns with a specific pose of the robot's end-effector, depending on that pose. The methods for solving inverse kinematics can be divided into two categories: the first approach involves mathematical analysis, while the second utilizes iterative numerical techniques like the Jacobian matrix method and the Newton-Raphson method. A mathematical analysis technique has been chosen to address the robot's reverse kinematic properties. In other words, to derive the equations for the joint variables, multiply the left-hand sides of the formula (3) by the inverse matrices that correspond to the joint variables. Subsequently, by balancing the components on each side of the equations, it's possible to separate each joint variable and determine the values for every joint angle in the robot. The angles formed at the joints are:

$$\theta_1 = a\,tan\,2\,(m,n) - a\,tan\,2\,(d_2 + d_3, \pm\sqrt{m^2 + n^2 - (d_2 + d_3)^2}) \tag{4}$$

$$\theta_2 = a\,tan\,2\,(k_3,k_4) - a\,tan\,2\,(k_5, \pm\sqrt{k_3^2 + k_4^2 - k_5^2}) \tag{5}$$

$$\theta_3 = a\,tan\,2\,(g, \pm\sqrt{a_3^2 + d_4^2 - g^2}) - a\,tan\,2\,(a_3.d_4) \tag{6}$$

When $s_5 \neq 0$,

$$\theta_4 = arcsin(\,(r_{23}c_1 - r_{13}s_1)/s_5) \tag{7}$$

At the point where $s_5 = 0$, the industrial robot assumes a unique arrangement. When the robot is in a unique arrangement, the value of θ_4 can be arbitrarily selected, and then the corresponding value of $\theta 6$ can be calculated.

$$\theta_5 = arccos(\,(r_{13}c_1 + r_{23}s_1)s_{23} - r_{33}c_{23}) \tag{8}$$

$$\theta_6 = \text{atan2}(-s_4s_{23}, \pm\sqrt{r_{31}^2 + r_{32}^2 - s_4s_{23}}) - \text{atan2}(r_{32},r_{31}) \tag{9}$$

In the formula:

$$m = d_6r_{23} - p_y, n = d_6r_{13} - p_x$$
$$k_1 = (d_6r_{13} - p_x)c_1 + (d_6r_{23} - p_y)s_1 + a_1$$
$$k_2 = p_z - d_1 - d_6r_{33}$$
$$k_3 = 2a_2k_1$$
$$k_4 = 2a_2k_2$$
$$k_5 = a_3^2 + d_4^2 - a_2^2 - k_1^2 - k_2^2$$
$$g = k_2s_2 - k_1c_2 - a_2.$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

The joint angles of this robotic arm have been calculated using the above formulas. As can be seen from the formulas, there are multiple solutions. The majority of robotic arms possess several reverse kinematic solutions, offering diverse choices for the location of the effector. Yet, when the automated limb is in motion, choosing the joint angles must take into account the limitations of the arm's parameters and the task's demands.

2.4 Kinematics Simulation Verification of Industrial Robots

Based on parameters D-H enumerated in Table 1, employ MATLAB's Robotics Toolbox to create a simulation model for the M - 710/iC50 robot. The simulation model of the robot is shown in Figure 4.

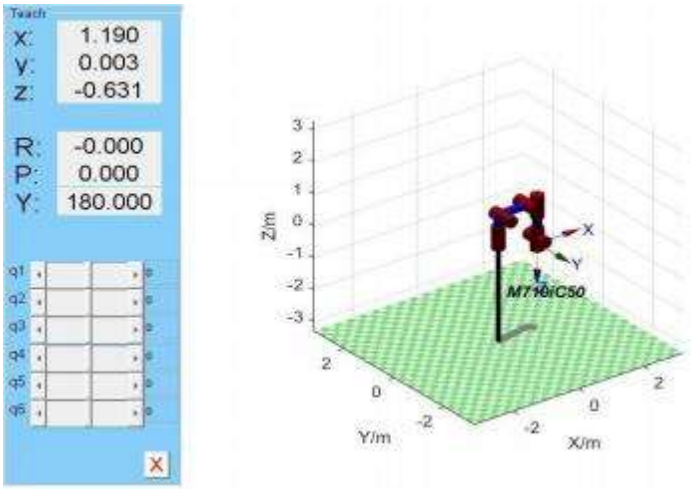


Figure 4. M-710/iC50 model diagram

Verify the robot's advancement by using the fkine feature in the Robotics Toolbox to compute its forward motion. Calculate using both the forward kinematics formula derived in this chapter and the fkine function. Given the initial joint angles, q1 = [0.5236, 2.0944, 0.5236, 0.5236, 0.5236, 0.5236], which means each joint rotates by 30°. Table 2 displays the computed outcomes for both techniques:

Table 2. Calculation results of two methods of forward kinematics

Calculation results of the derived formula	Calculation results of the *fkine* function
${}_6^0T = \begin{bmatrix} 0.2920 & 0.7643 & 0.5749 & 0.1648 \\ -0.7644 & -0.1744 & 0.6204 & 0.1490 \\ 0.5749 & -0.6204 & 0.5336 & 2.3721 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	${}_6^0T = \begin{bmatrix} 0.292 & 0.765 & 0.575 & 0.165 \\ -0.765 & -0.175 & 0.621 & 0.149 \\ 0.575 & -0.621 & 0.533 & 2.372 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

A comparison of the computational outcomes in Table 2 reveals consistency between the two approaches, indicating that the forward kinematics formula formulated in this chapter aligns with the varying dynamics among the robot links.

This document employs the *ikunc* inverse kinematics function in the Robotics Toolbox for validating the robot's inverse kinematics. As mentioned before, robot inverse kinematics is to obtain the joint angles of the robot by back-calculating from the target pose of the end-effector. Now, substitute the forward kinematics calculation results into the *ikunc* function to calculate the joint angles. The outcomes of the computations are presented in Table 3:

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

Table 3. Joint Position Sequence Calculated by ikunc Function (rad)

number	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
1	0.5236	2.0944	0.5236	0.5236	0.5236	0.5236

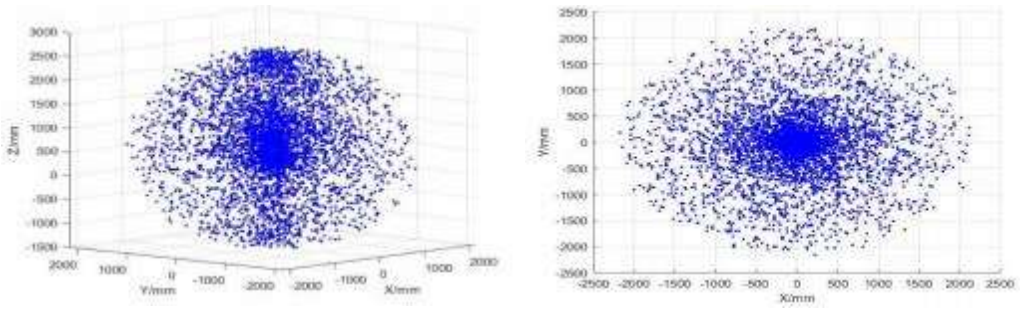
Table 3 illustrates that the joint angles determined by the *ikunc* function align with the initial angles specified in forward motion dynamics. Insert the outcomes of forward motion dynamics calculations into the previously derived equation of inverse kinematics to determine angles at the joints. The outcomes of the computations are presented in Table 4:

Table 4. Derived Formula Calculation of Joint Position Sequence(rad)

number	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
1	-2.5389	1.2394	0.55694	-0.25724	2.3255	1.1212
2	-2.5389	1.2394	0.55694	-0.25724	2.3255	3.6677
3	-2.5389	1.2394	2.2531	-0.3036	0.66868	0.70187
4	-2.5389	1.2394	2.2531	-0.3036	0.66868	-2.1961
5	-2.5389	2.1635	0.55694	-0.18856	1.4207	0.91438
6	-2.5389	2.1635	0.55694	-0.18856	1.4207	3.8745
7	-2.5389	2.1635	2.2531	-0.60199	0.33343	0.12858
8	-2.5389	2.1635	2.2531	-0.60199	0.33343	-1.6229
9	0.5236	1.1335	0.5236	0.49616	0.5529	0.22838
10	0.5236	1.1335	0.5236	0.49616	0.5529	-1.7227
11	0.5236	1.1335	2.2864	0.26437	1.2756	0.90865
12	0.5236	1.1335	2.2864	0.26437	1.2756	3.8803
13	0.5236	2.0944	0.5236	0.5236	0.5236	0.5236
14	0.5236	2.0944	0.5236	0.5236	0.5236	-2.0179
15	0.5236	2.0944	2.2864	0.31491	2.2023	1.1772
16	0.5236	2.0944	2.2864	0.31491	2.2023	3.6117

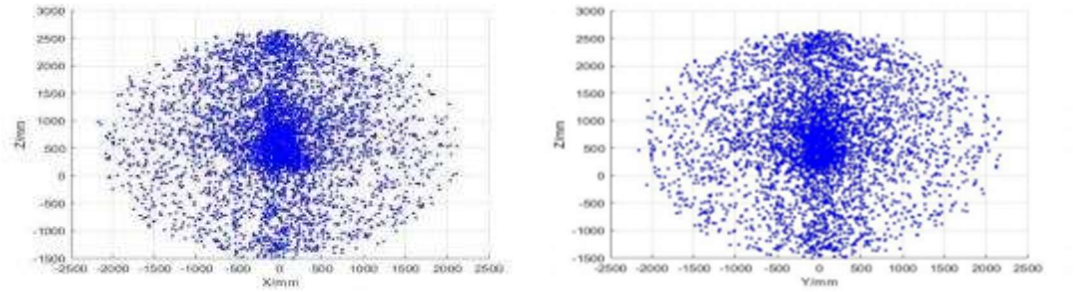
As shown in Table 4, the joint angles obtained by the formula are the same as those calculated by the `ikunc` function, which means that the derived inverse kinematics formula is correct.

The workspace is an important indicator for evaluating the motion performance of a robot. In this paper, the Monte Carlo method is used to simulate the robot's workspace. The range of the robot's workspace is shown in Figure 4.



(a) Robot movement area simulation (b) Robot movement area in the X-Y plane

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018



(c)Robot movement area in the X-Z plane (d) Robot movement area in the Y-Z plane

Figure 5. M-710/iC50 workspace

Figure 5 illustrates that the operational area of the FANUCM-710/iC50 six-axis robot takes a spherical form. Components X, Y, and Z fulfill the robot's operational needs, offering a solid theoretical foundation for future planning and fine-tuning the robot's path.

3. Trajectory Planning of Industrial Robot in Joint Space

Within industrial settings, joint-space trajectory planning is typically chosen when there are no particular path prerequisites for an industrial robot's end-effector, meaning for point-to-point movement. Planning joint-space trajectories entails meticulously tracking the progression of each joint within an industrial robot over a period. Pathway planning techniques frequently employed are Interpolation of polynomials and B-spline techniques. Given that this technique primarily dictates the movement path of the terminal device of the industrial robot via the angles of joints and requires fewer variables, the computation is comparatively straightforward and the computational burden is comparatively minor. This part utilizes a polynomial interpolation algorithm for research, tailored to the specific needs of planning of trajectories in joint space.

3.1 Cubic Polynomial Interpolation Algorithm

Industrial robots frequently employ the cubic polynomial interpolation algorithm for planning their joint-space paths. Utilizing the cubic polynomial interpolation function to link neighboring points on the industrial robot's task path enables the robot to achieve a motion path characterized by a seamless and uninterrupted velocity curve. The mathematical expression of the cubic polynomial interpolation function is as follows:

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{10}$$

In the formula:a_0, a_1, a_2, a_3-coefficients of the cubic polynomial.

Presume knowledge of the starting and ending points of the terminal device of the robot, along with the initial moment $t_0 = 0$ and the final moment tf of the combined movement. Utilizing the robot's reverse kinematics, the initial joint angle can be determined as θ_0, and the final joint angle as θ_f. Typically, the robot's speeds at both the starting and ending points are zero. Subsequently,

$$\begin{aligned} \theta(0) = \theta_0, \theta(t_f) = \theta_f \\ \dot{\theta}(0) = 0, \dot{\theta}(t_f) = 0 \end{aligned} \tag{11}$$

Solve the system of linear equations to obtain

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

$$\begin{cases} a_0 = \theta_0 \\ a_1 = 0 \\ a_2 = \frac{3}{t_f^2}(\theta_f - \theta_0) \\ a_3 = -\frac{2}{t_f^2}(\theta_f - \theta_0) \end{cases} \tag{12}$$

Substitute the coefficients obtained from formula (12) into formula (10), and then the corresponding cubic polynomial expression can be obtained.

3.2 Quintic Polynomial Interpolation Algorithm

Within the previously developed cubic polynomial interpolation algorithm, continuity is assured solely for position and velocity curves, with no restrictions on acceleration. Moreover, there is a problem of sudden changes in acceleration. Consequently, in scenarios demanding stringent conditions for the operational path and acceleration limitations, employing a more complex polynomial interpolation function becomes necessary for computing trajectory interpolation. Usually, a quintic polynomial function is used for interpolation calculations. The mathematical expression of the quintic polynomial is:

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{13}$$

In the formula: $a_0, a_1, a_2, a_3, a_4, a_5$-coefficients of the quintic polynomial.Six coefficients remain undefined. Consequently, to determine the quintic polynomial's six coefficients, six specific conditions must be met. To ascertain the precise values of these coefficients, it's necessary to establish the constraint equations that rely on the boundary conditions. Based on established parameters like the starting location θ_0 , velocity v_0 , and acceleration a_0 , along with the final position θ_f , velocity v_f , and acceleration a_f , equations of limitations are outlined as follows:

$$\begin{gathered} \theta(t_0) = \theta_0, \theta(t_f) = \theta_f \\ \dot{\theta}(t_0) = v_0, \dot{\theta}(t_f) = v_f \\ \ddot{\theta}(t_0) = a_0, \ddot{\theta}(t_f) = a_f \end{gathered} \tag{14}$$

Taking the first- and second-order derivatives of formula (13) to obtain:

$$\begin{cases} \theta_0 = a_0 \\ \theta_f = a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_3^3 + a_4 t_f^4 + a_5 t_f^5 \\ \dot{\theta}_0 = a_1 \\ \dot{\theta}_f = a_1 + 2a_2 t_f + 3a_3 t_f^2 + 4a_4 t_f^3 + 5a_5 t_f^4 \\ \ddot{\theta}_0 = 2a_2 \\ \ddot{\theta}_f = 2a_2 + 6a_3 t_f + 12a_4 t_f^2 + 20a_5 t_f^3 \end{cases} \tag{15}$$

Solving the system of linear equations shown in formula (15) to obtain:

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

$$\begin{cases} a_0 = \theta_0 \\ a_1 = \dot{\theta}_0 \\ a_2 = \frac{1}{2}(\ddot{\theta}_0) \\ a_3 = \frac{20\theta_f - 20\dot{\theta}_0 - (8\dot{\theta}_f + 12\dot{\theta}_0)t_f - (3\ddot{\theta}_0 - \ddot{\theta}_f)t_f^2}{2t_f^3} \\ a4 = \frac{30\theta_0 - 30\theta_f + (14\dot{\theta}_f + 16\dot{\theta}_0)t_f + (3\ddot{\theta}_0 - 2\ddot{\theta}_f)t_f^2}{2t_f^4} \\ a5 = \frac{12\theta_f - 12\dot{\theta}_0 - (6\dot{\theta}_f + 6\dot{\theta}_0)t_f - (\ddot{\theta}_0 - \ddot{\theta}_f)t_f^2}{2t_f^5} \end{cases} \quad (16)$$

Substituting the coefficients obtained from formula (16) into formula (13), the corresponding quintic polynomial can be obtained. In contrast to the method of interpolating cubic polynomials, interpolation of quintic polynomials has been shown to facilitate a more fluid movement path. However, the quintic polynomial interpolation algorithm requires more constraint conditions, and the computational complexity also increases accordingly. Similarly, if a polynomial of a higher order is needed for interpolation in some special situations, more boundary conditions are required. For example, the seventh-order polynomial interpolation algorithm also needs to add the jerk (third-derivative of the joint angle) constraint conditions. This particular derivation method bears resemblance to the cubic and quintic polynomial interpolation techniques.

3.3 3-5-3 Hybrid Polynomial Interpolation Algorithm

In actual industrial production, the working environment and motion trajectories of robots are complex and variable. When conducting trajectory planning, the 3 - 5 - 3 hybrid polynomial interpolation algorithm is usually selected.

Initially, the algorithm of hybrid polynomial interpolation known as 3-5-3 serves as a seamless and ongoing approach for organizing paths of movement. Based on the assumption that the robot can navigate the path with ease, it is capable of efficiently addressing the issue of abrupt acceleration shifts in the algorithm for interpolating cubic polynomials and the significant computational intricacy in the autonomous quintic polynomial interpolation technique.

Additionally, the algorithm of hybrid polynomial interpolation known as 3-5-3 exhibits significant versatility and adaptability. It can be adjusted according to different path requirements and constraint conditions to meet the trajectory planning needs in various practical application scenarios. Concurrently, by modifying the polynomial coefficients, this technique enhances the robot's control over its movement, thereby managing the trajectory's speed and acceleration.

Assume that the poses of the initial interpolation point p_0, the intermediate interpolation points p_1, p_2 and the terminal interpolation point p_3 in the motion trajectory are known. The robot's inverse kinematic equation allows for solving the angular measurements at each joint that align with the quartet of points for interpolation. Define θij as the interpolated angle for joint *i*, with *i* indicating the joint number (*i* = 1, 2,..., 6) and j as the serial number for the interpolation point (*j* = 1, 2, 3).

For the *i*-th joint, the standard equation of the algorithm of hybrid polynomial interpolation known as 3-5-3 is as follows:

$$\begin{aligned} \theta_{i,1}(t_1) &= a_{10} + a_{11}t_1 + a_{12}t_1^2 + a_{13}t_1^3 \\ \theta_{i.2}(t_2) &= a_{20} + a_{21}t_2 + a_{22}t_2^2 + a_{23}t_2^3 + a_{24}t_2^4 + a_{25}t_2^5 \\ \theta_{i,3}(t_3) &= a_{30} + a_{31}t_3 + a_{32}t_3^2 + a_{33}t_3^3 \end{aligned} \quad (17)$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

The formula includes: $\theta_{i1}(t_1)$, $\theta_{i2}(t_2)$, $\theta_{i3}(t_3)$ as the polynomial angular displacement function; aij as the polynomial's j-th coefficient for the i-th (i = 1, 2, 3) segment, with j varying based on the polynomial's degree; t as the variable for interpolation time.

The formula (3) reveals that the 3 - 5 - 3 polynomial interpolation function possesses 14 indeterminate coefficients a_{ij}. Therefore, 14 boundary equations need to be established to calculate the specific values of the undetermined coefficients. Limitation equations are formulated based on the starting points' positions x_{j0}, points of the central path x_{j1} and x_{j2}, and the endpoint x_{j3} of each part of the i-th joint's trajectory, along with established conditions like the consistency of position, speed, and acceleration among these points, and the initial velocity (typically 0) and acceleration. Based on the boundary conditions and constraint equations, the relationship matrix A between the joint interpolation points and the polynomial coefficients can be derived, and the conditions and limits of the constraints are solely connected to the time t. The relationship expression between the relationship matrix A and the coefficient vector a is as follows:

$$Aa = \theta \tag{18}$$

In formula (18), the expression for the polynomial coefficient vector a is:

$$\begin{cases} a_1 = [a_{13} \quad a_{12} \quad a_{11} \quad a_{10}] \\ a_2 = [a_{25} \quad a_{24} \quad a_{23} \quad a_{22} \quad a_{21} \quad a_{20}] \\ a_3 = [a_{33} \quad a_{32} \quad a_{31} \quad a_{30}] \\ a = [a_1 \quad a_2 \quad a_3] \end{cases} \tag{19}$$

The formula includes: a_1, a_2, a_3, representing the polynomial coefficients for each part of the trajectory curve at the i-th joint. a_{ij} - undetermined coefficients of the polynomial function.

The expression for the relationship matrix A is as follows:

$$A = \begin{bmatrix} t_1^3 & t_1^2 & t_1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 3t_1^2 & 2t_1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6t_1 & 2 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & t_2^5 & t_2^4 & t_2^3 & t_{22}^2 & t_2 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 5t_2^4 & 4t_2^3 & 3t_2^2 & 2t_2 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 20t_2^3 & 12t_2^2 & 6t_2 & 2 & 0 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_3^3 & t_3^2 & t_3 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3t_3^2 & 2t_3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6t_3 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{20}$$

The expression for the joint angle θ is as follows:

$$\theta = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad x_{i3} \quad 0 \quad 0 \quad x_{i0} \quad 0 \quad 0 \quad x_{i2} \quad x_{i1}]^T \tag{21}$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

Within the equation, x_{ij} symbolizes the position of the j-th interpolation point on the i-th joint.

In summary, as can be seen from formula (21), the 3-5-3 piecewise polynomial trajectory planning is only related to the time variable t.

3.4 Simulation and Result Analysis of Polynomial Interpolation Algorithm

3.4.1 Simulation of Cubic Polynomial Interpolation

Utilizing the second joint as the simulation subject, a cubic polynomial function facilitates interpolation fitting within the joint space. Table 5 displays the sequence of positions for each joint.

Table 5. Joint position sequence(°)

i	Joint 1	Joint 2	Joint 3	Joint 4	Joint5	Joint6
1	0	90	0	0	90	0
2	45	60	-20	0	40	120
3	0	110	-40	0	-110	240
4	0	90	-10	0	-100	360

As per the outcomes of the simulations, the motion curve of joint 2 is plotted, as shown in Figure 6.

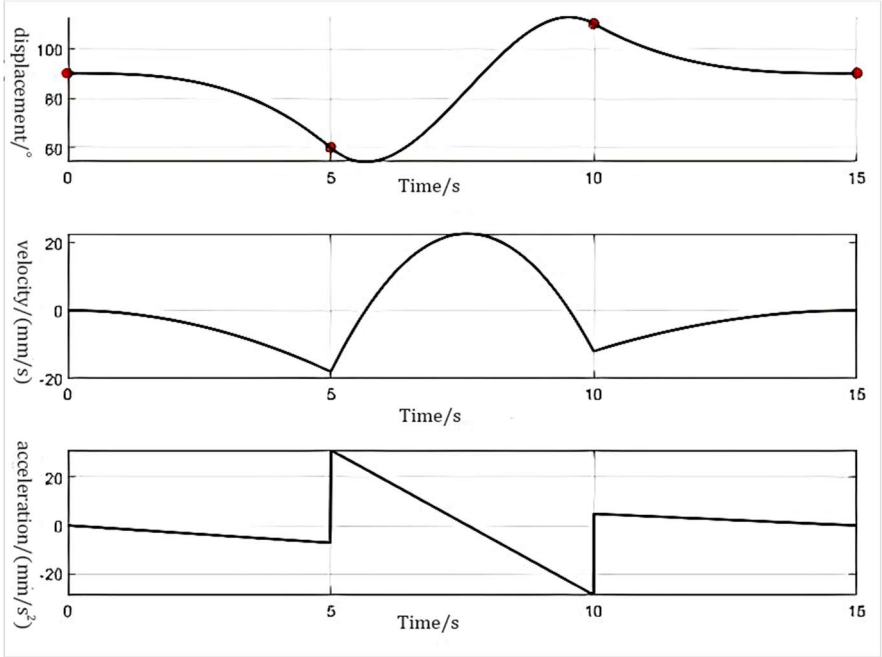


Figure 6. Pathway of interpolating cubic polynomials

As can be seen from Figure 6, the angular displacement and angular velocity curves of the cubic polynomial are smooth and continuous, but there is a sudden change in the angular acceleration. Although the cubic polynomial algorithm is simple and easy to implement, the sudden change in acceleration causes a significant impact on the robot during its operation.

3.4.2 Simulation of Quintic Polynomial Interpolation

Utilizing the industrial robot's second joint as the simulation subject, cubic polynomial interpolation is applied to align identical interpolation points. The simulation outcomes reveal that the movement trajectory of joint 2 is graphically represented in Figure 7.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

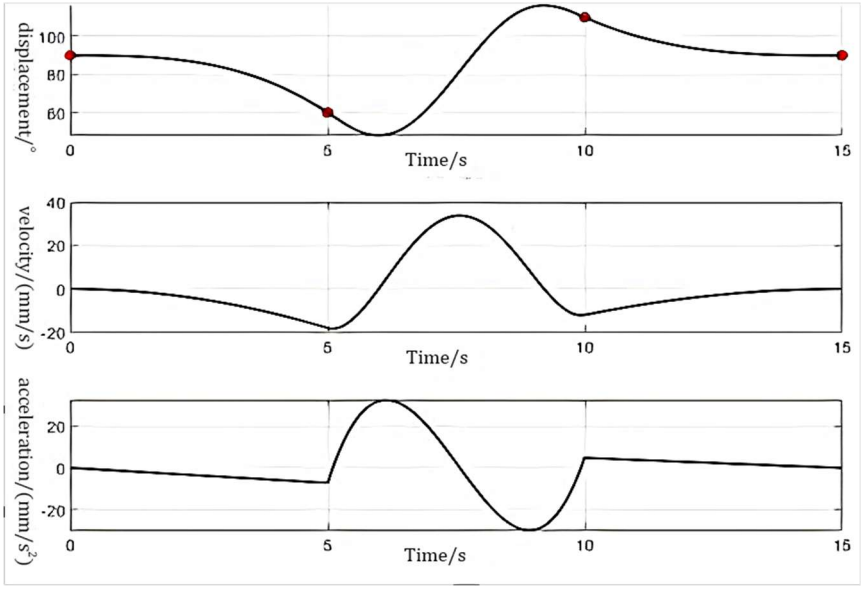


Figure 7. Quintic polynomial interpolation trajectory

Figure 7 illustrates that the quintic polynomial's angular displacement, velocity, and acceleration graphs are seamless and unbroken, successfully circumventing the abrupt alteration issue in angular acceleration during cubic polynomial interpolation. Differing from Interpolation of cubic polynomials, the quintic polynomial interpolation algorithm's computation is notably intricate and demands extensive computational effort.

3.4.3 Simulation of 3-5-3 Polynomial Interpolation Trajectory

Utilizing the second joint as the simulation subject, the 3-5-3 polynomial function facilitates interpolation fitting within the joint space. The simulation outcomes reveal that the movement trajectory of joint 2 is graphically represented in Figure 8.

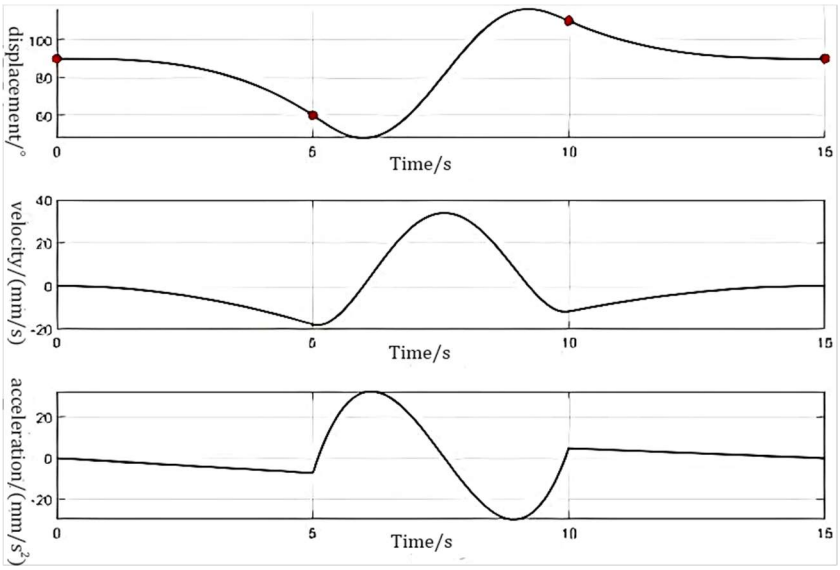


Figure 8. 3-5-3 Path of polynomial interpolation, 3-5-3

Figure 8 illustrates the application of the 3-5-3 polynomial function for executing interpolation fitting on the combined variables within the combined area. The curves representing displacement in angular terms and velocity are seamless and unbroken, with neither angular velocity nor acceleration registering as zero at the commencement nor cessation points. Utilizing the 3-5-3 polynomial

ISSN: 2414-1895 DOI: 10.6919/ICJE.202508_11(8).0018

interpolation algorithm successfully circumvents issues related to abrupt changes in acceleration. The trajectory of the motion closely resembles that found in the quintic polynomial interpolation algorithm. Differing from the quintic polynomial interpolation method, the 3-5-3 polynomial interpolation algorithm shows a significant reduction in computational complexity and demand.

In summary, through the comparative analysis of the motion curves of various interpolation algorithms and considering multiple aspects such as the robot's stability, processing quality, and production efficiency, when conducting multi - index comprehensive optimal trajectory optimization, the 3-5-3 polynomial function can be selected for joint trajectory interpolation fitting.

4. Conclusion

The study in this document delves into the strategic planning of industrial robot trajectories, focusing on the extensive optimization of time, shock, and agility. This finalizes the study on delineating rigid-body motion states, the conventional D-H modeling technique, developing both forward and inverse kinematics, and conducting kinematic simulations. The Robotics Toolbox is utilized to create a model for robot parameters. Utilizing the positional data of path points, computational simulations are conducted on both forward and reverse kinematic derivation methods to derive the robot's forward kinematic solutions and joint angles that align with all inverse solutions. An analysis is also conducted on the operational area of the M - 710/iC50 Industrial robot with a liberty of six degrees. The study focuses on the 3 - 5 - 3 polynomial interpolation method, with MATLAB-based numerical simulations of various trajectory planning techniques. The movement trajectories of every joint derived from the simulations undergo comparison and examination. Taking into account various factors like the operational effectiveness and shock intensity of the industrial robot, the 3 - 5 - 3 polynomial interpolation technique is chosen for combined path adjustment, laying a theoretical groundwork for subsequent trajectory optimization.

References

[1] Jin, B. (2023). Made in China: Sharing China's Opportunities with the World. China Press Release (Practical Edition), (08), 19–22.

[2] Wu, Z., Teng, X., & Xia, W. (2023). New Developments in Mechanical Design under the Background of "Made in China 2025". Mechanical & Electrical Engineering Technology, 52(06), 46–49.

[3] Cui, H., & Chen, L. (2022). "Intelligent Manufacturing" in the "Made in China 2025" Strategy. Science - Technology & Economic Market, (04), 7–9.

[4] Dobiš, M., Dekan, M., Sojka, A., et al. (2021). Cartesian constrained stochastic trajectory optimization for motion planning. Applied Sciences, 11(24), 11712.

[5] Li, Z., Wang, T., Wang, B., et al. (2019). Cartesian space trajectory planning of manipulator based on constrained S - type velocity curve. CAAI Transactions on Intelligent Systems, 14(04), 655–661.

[6] Xu, W., Li, C., Liang, B., et al. (2008). The Cartesian path planning of free - floating space robot using particle swarm optimization. International Journal of Advanced Robotic Systems, 5(3), 27.

[7] Wang, M., Luo, J., Yuan, J., et al. (2018). Coordinated trajectory planning of dual - arm space robot using constrained particle swarm optimization. Acta Astronautica, 146, 259–272.

[8] Liu, H., Lai, X., & Wu, W. (2013). Time - optimal and jerk - continuous trajectory planning for robot manipulators with kinematic constraints. Robotics and Computer - Integrated Manufacturing, 29(2), 309–317.

[9] Xiao, P., Ju, H., & Li, Q. (2021). Point - to - point trajectory planning for space robots based on jerk constraints. Review of Scientific Instruments, 92(9).

[10] Cai, J., Deng, J., Zhang, W., et al. (2021). Modeling method of autonomous robot manipulator based on DH algorithm. Mobile Information Systems, 2021, 1–10.

[11] Wen, H. (2019). Analysis of D - H Modeling Method for Serial Robots. Modern Manufacturing Technology and Equipment, (04), 117–118 + 121.