Study on trajectory optimization algorithm of industrial robot in joint space

SHEN Luyang\textsuperscript{1,a}, PENG Yichao\textsuperscript{2,b}

\textsuperscript{1}Wuhan University of Technology, WuHan, Chinese
\textsuperscript{2}Wuhan University of Technology, WuHan, Chinese
\textsuperscript{a}601053243@qq.com, \textsuperscript{b}937033482@qq.com

Abstract—Aiming at the problem that the trajectory deviation of six-degree-of-freedom industrial robot is large, a robot model is established by Denavit-Hartenberg parameter method, and forward kinematics and inverse kinematics are analysed. Simulation is conducted on the Robotics Toolbox industrial robot model in MATLAB, and cubic polynomial interpolation and cubic polynomial interpolation are used to plan the terminal motion trajectory of the industrial robot in joint space. The results show that the error of the terminal motion trajectory is smaller in the fifth order polynomial interpolation.

1. INTRODUCTION

In this paper, the Denavit-Hartenberg method is used to solve the forward and inverse kinematics of the robot, which improves the control accuracy of the robotic arm, shortens the running time of the algorithm.

At present, on a given manipulator trajectory planning in joint space use three and five times more polynomial interpolation function, less than three function interpolation method of low order time, derivative get acceleration is constant, there is no guarantee that the acceleration the continuity of trajectory, and the practical application of all kinds of high order polynomial interpolation function, the higher order time can cause a dramatic interpolation interval exists on both ends of the longitudinal wave.

uses polynomial interpolation to perform trajectory planning, and intuitively shows the end trajectory of the industrial robot in joint space. According to the simulation experiments of MATLAB, the rationality of the algorithm is verified, which provides a theoretical basis for the practical application of the robotic arm motion system.

2. INDUSTRIAL ROBOT MODELING

Modeling of industrial is an important part of the research. Denavit-Hartenberg parameter method is used to model industrial robots and MATLAB software is used for simulation.

2.1. Denavit-Hartenberg parameter modeling

The Denavit-Hartenberg parameter is a mechanical arm mathematical model and coordinate system determination system that expresses the position and angle relationship between adjacent joint links with four parameters. It can be applied to any robot configuration. First of all, we need to establish the dh coordinate system.
Figure 1 Coordinate system of two adjacent links. $\theta_i$ is the angle from $X_{i-1}$ to $X_i$ about $Z_i$ Axis; $d_i$ is the distance from $X_{i-1}$ to $X_i$ along $Z_i$ Axis; $a_i$ is the distance from $Z_i$ to $Z_{i+1}$ along $X_i$ Axis; $\alpha_i$ is the angle from $Z_i$ to $Z_{i+1}$ about $X_i$ Axis.

The transformation from the coordinate system $\{i-1\}$ to $\{i\}$ is defined as the basic rotation and translation transformation. The transformation matrix between two adjacent coordinate systems can be written as

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

2.2 MATLAB simulation model

Determination of structural parameters.

| joint | $\theta$ (°) | d/mm | a/mm | $\alpha$ (°) |
|-------|-------------|------|------|-------------|
| 1     | $\theta_1$  | 0    | 0    | $\pi/2$     |
| 2     | $\theta_2$  | 0    | 0.4318 | 0           |
| 3     | $\theta_3$  | 0.15005 | 0.0203 | $-\pi/2$    |
| 4     | $\theta_4$  | 0.4318 | 0    | $-\pi/2$    |
| 5     | $\theta_5$  | 0    | 0    | $-\pi/2$    |
| 6     | $\theta_6$  | 0    | 0    | 0           |

Combining the parameters in Table 1 and using the simulation function of the Robotics Toolbox in MATLAB, the simulation model of the industrial robot when the joint angle is all 0° is shown in Figure 2.

Figure 2 Industrial robot simulation model when the joint angles are all 0°
3. **FORWARD AND INVERSE KINEMATICS ANALYSIS**

The analysis of the forward and inverse kinematics of the robot helps us to know more clearly how the robot moves, which lays a foundation for the future research on the trajectory planning algorithm.

3.1. **Forward kinematics analysis**

Robot positive kinematics refers to the parameters of each joint, and to find the pose of the end effector relative to the base coordinate system. Find the transformation matrix from the end coordinate system to the base coordinate system. Substituting the Denavit-Hartenberg parameter into equation (1) goes here.

\[
T_1^0 = \begin{bmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 & 0 \\
\sin \theta_1 & \cos \theta_1 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
T_2^1 = \begin{bmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 & 0 \\
0 & 0 & 1 & d_2 \\
\end{bmatrix}
\]

\[
T_3^2 = \begin{bmatrix}
\cos \theta_3 & -\sin \theta_3 & a_2 & 0 \\
\sin \theta_3 & \cos \theta_3 & 0 & 0 \\
0 & 0 & 1 & d_3 \\
\end{bmatrix}
\]

\[
T_4^3 = \begin{bmatrix}
\cos \theta_4 & -\sin \theta_4 & 0 & a_3 \\
0 & 0 & 1 & d_4 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
T_5^4 = \begin{bmatrix}
\cos \theta_5 & -\sin \theta_5 & 0 & 0 \\
0 & 0 & 1 & d_5 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
T_6^5 = \begin{bmatrix}
\cos \theta_6 & -\sin \theta_6 & 0 & 0 \\
0 & 0 & 1 & d_6 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
T_6^0 = T_1^0 T_2^1 T_3^2 T_4^3 T_5^4 T_6^5 = \begin{bmatrix}
n_x & a_x & a_x & p_x \\
n_y & a_y & a_y & p_y \\
n_z & a_z & a_z & p_z \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

3.2. **Inverse kinematics analysis**

Knowing the expected position and posture of the joint end, the rotation angle of each joint is obtained by the variable separation method.

\[
T_1^0 = T_1^0(T_2^1 T_3^2 T_4^3 T_5^4 T_6^5) = \begin{bmatrix}
n_x & a_x & a_x & p_x \\
n_y & a_y & a_y & p_y \\
n_z & a_z & a_z & p_z \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Substituting the robot end pose into (4), the following equation can be derived.

\[
\theta_1 = \tan^{-1} \frac{p_y}{p_x} \text{ and } \theta_1 = \theta_1 + 180^\circ
\]

\[
\theta_2 = \tan^{-1} \frac{(C_3a_3 + a_2)(p_x - S_{234}a_4) - S_3a_3(p_x + p_2S_1 - C_{234}a_4)}{(C_3a_3 + a_2)(p_x + p_2S_1 - C_{234}a_4) + S_3a_3(p_x - S_{234}a_4)}
\]
\[ \theta_3 = \tan^{-1} \frac{S_3}{C_3} \]

\[ \theta_4 = \theta_{234} - \theta_2 - \theta_3 \]

\[ \theta_5 = \tan^{-1} \frac{C_{234}(C_1a_x + S_1a_y) + S_{234}a_z}{S_1a_x - C_1a_y} \]

\[ \theta_6 = \tan^{-1} \frac{-S_{234}(C_1n_x + S_1n_y) + C_{234}n_z}{-S_{234}(C_1a_x + S_1a_y) + C_{234}a_z} \]

Where \( S_i = \sin \theta_i \); \( C_i = \cos \theta_i \); \( S_{ABC} = \sin(\theta_A + \theta_B + \theta_C) \); \( C_{ABC} = \cos(\theta_A + \theta_B + \theta_C) \);

\[ \theta_{234} = \tan^{-1} \left( \frac{a_z}{C_1a_x + S_1a_y} \right) \]

Use the \texttt{fkine} function and \texttt{ikine6s} function in MATLAB’s Robotics Toolbox for verification. The \texttt{fkine} function uses the format: "T=fkine" ("robot,q") " ukine function call format is q=robot.ikine6s(T). Where robot is the established robot model, q is the joint angle vector, and T is the positive solution change matrix corresponding to q. Take a random point \( Q_1 \), the corresponding joint angle vector \( q = [0 \pi / 4 -\pi 0 \pi / 4 0] \). Use \texttt{fkine} function to solve for T to get

\[ T = \begin{bmatrix} 0 & 0 & 1 & 0.5963 \\ 0 & 1 & 0 & -0.1501 \\ -1 & 0 & 0 & -0.01435 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]  

The joint angle vector is substituted into (4) to solve, and the result is completely consistent with the matrix (6), which verifies the correctness of the positive kinematics equation. \( q=\text{robot.ikine6s}(T) \).

4. JOINT SPACE TRAJECTORY PLANNING

The space formed by all joint vectors is called joint space. In the point-to-point motion control of a robotic arm, the joint variables of the six joints at the starting point are known, and the joint variables corresponding to the ending point can be calculated by inverse kinematics according to the end position of the given point. Therefore, the motion trajectory of each joint can be represented by a smooth \( \theta(t) \) function passing through the start point and the end point. According to the different constraints, it can be divided into a cubic polynomial interpolation function and a quartic polynomial interpolation function.

4.1. Cubic polynomial interpolation

The cubic polynomial interpolation method has four constraints, the angle "q" "0" at the initial movement of the manipulator, and the angle "q" "t" at the end point. To avoid damage to the manipulator, we also set its initial speed to 0 and the end speed to 0.

\[ \theta(t_0) = q_0; \quad \theta(t_f) = q_f; \quad V(t_0) = 0; \quad V(t_f) = 0; \]

Four constraints can finally get a cubic polynomial, which is the joint angle:

\[ \theta(t) = c_0 + c_1t + c_2t^2 + c_3t^3 \]

Find the first-order derivative of this formula to get the joint velocity:

\[ V(t) = c_1 + 2c_2t + 3c_3t^2 \]

Derive it to get joint acceleration:

\[ a(t) = 2c_2 + 6c_3t \]
Substitute the four constraints into the solution:

\[ c_0 = q_0; \ c_1 = 0; \ c_2 = \frac{3(q_f - q_0)}{t_f^2}; \ c_3 = \frac{-2(q_f - q_0)}{t_f^3}; \]

(11)

4.2. Quintic polynomial interpolation

The quintic polynomial has six constraints, which are added to the four constraints of the cubic polynomial interpolation method, and the starting and ending accelerations are 0.

\[ \theta(t_0) = q_0; \ \theta(t_f) = q_f; V(t_0) = 0; V(t_f) = 0; a(t_0) = 0; a(t_f) = 0; \]

Six constraints can finally get a fifth degree polynomial, which is the joint angle

\[ \theta(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 + c_5 t^5 \]

(13)

Find the first-order derivative of this formula to get the joint velocity

\[ V(t) = c_1 + 2c_2 t + 3c_3 t^2 + 4c_4 t^3 + 5c_5 t^4 \]

(14)

Derive it to get joint acceleration

\[ a(t) = 2c_2 + 6c_3 t + 12c_4 t^2 + 20c_5 t^3 \]

(15)

Substitute the six constraints into the solution

\[ c_0 = q_0; \ c_1 = 0; \ c_2 = 0; \ c_3 = \frac{20(q_f - q_0)}{2t_f^3}; \ c_4 = \frac{30(q_f - q_0)}{2t_f^4}; \ c_5 = \frac{12(q_f - q_0)}{2t_f^5} \]

(16)

4.3. Matlab simulation results

In the Cartesian coordinate system, we choose (0.4, 0.2, 0.2) as the starting point and (0.4, -0.2, 0) as the end point, and use MATLAB to plan the trajectory of the robot. Choose joint one, joint three, and joint six among the six joints to be compared and analyzed, and the simulation are shown in the following figure.

Figures 3 Joint 1 plans the rear Angle, angular velocity and angular acceleration
In the above picture, (a), (b) and (c) are Cubic polynomial interpolation. (d) (e) (f) is after Quintic polynomial interpolation. From Figures 3, 4, it can be seen that the cubic polynomial, fifth polynomial joint angle curve, and joint speed curve are smooth, and only the fifth polynomial acceleration curve is smooth on the joint acceleration. The movement of the robotic arm is first accelerated and then decelerated. In the process, the trajectory planned by the cubic polynomial interpolation method does not have an acceleration of 0 when it reaches the target point, indicating that it is not stable. The trajectory planned by the quintic polynomial interpolation method, the stability of the robot arm in motion is better.

5. CONCLUSION
This paper uses the MATLAB Robotics Toolbox to build a robot simulation model using the Denavit-Hartenberg parameter method, and analyzes its forward and inverse kinematics, and uses polynomial interpolation to perform trajectory planning in joint space to generate different joint angles, joint angular velocities, and joint angular accelerations. The results show that the robot model established is correct, and the trajectory planned by the fifth-degree polynomial interpolation method is more stable through simulation analysis, which lays a theoretical foundation for subsequent robot-related research.

REFERENCES
[1] Peter Corke.(2016) Robotics, Machine Vision and Control. Beijing. Electronic Industry Publishing, Beijing
[2] LIU Qiang. YANG Daoguo. HAO Weidong. (2019) Kinematic Analysis and Trajectory Planning of UR10 Robot. MACHINE TOOL & HYDRAULICS, 47: 22–28.
[3] Saeed B, Niku.(2018) Introduction to Robotics . Beijing. Electronic Industry Publishing, Beijing.
[4] Li Xianhua, Fan Kaijie, Zhang Jun. (2018) Trajectory planning and simulation of six degrees of freedom modular manipulator. Design and Research, 9: 73–77.

[5] Li Linlin, Kou Ziming, Wu Juan. (2019) Research on Design and Trajectory Planning of Six Degrees of Freedom Manipulator. Coal Mine Machinery, 4