Industrial robot trajectory planning based on improved pso algorithm

Han Shunjie1,a*, Shan Xinchao2,b, Fu Jinxin3,c, Xu Weijin4,d, Mi Hongyan5,e

1,2,3College of Electrical and Electronic Engineering, Changchun University of Technology, Changchun, Jilin, China
4,5State Grid Changchun Power Supply Company, Changchun, Jilin, China
a*email: hanshunjie@ccut.edu.cn, bemail: shin0921@foxmail.com, cemail: aries-fjx@foxmail.com, demail: 123757217@qq.com, eemail: 446249925@qq.com

Abstract. A particle swarm optimization (PSO) algorithm which can dynamically adjust learning factors is proposed to solve the problems of low efficiency and unstable operation of traditional industrial robots. The method uses piecewise polynomial interpolation to fit the trajectory, and uses an improved particle swarm algorithm to optimize the trajectory of industrial robots with time as a fitness function. This method effectively combines the piecewise polynomial interpolation function with PSO, avoids the complex process of particle swarm algorithm to construct the adaptation function, and improves the problem that the traditional PSO is more likely to fall into local extreme value in the early stage and convergence speed is slow in the later stage. Through experiments to obtain the motion pose, velocity and acceleration trajectory of each joint of the manipulator, we can know that this method can effectively realize the trajectory optimization of the industrial robot, and ensure the stability of the overall operation while improving the operating efficiency.

1. Introduction

With the increasing importance of intelligent manufacturing in social development, robots have become an indispensable key factor in the process of intelligent development, so how to make robots run more intelligently, safely and reliably is particularly important. Trajectory planning is an important part of robotics, and it is also a hot topic in robotics kinematics. Trajectory optimization can further improve the performance of the planned trajectory, meet actual needs, shorten the movement time of the robotic arm, reduce energy consumption, and at the same time can avoid unnecessary jitter and impact, and increase stability.

There are many research papers on trajectory planning by domestic and foreign scholars. Literature [1] and [2] use cubic polynomial functions for trajectory planning, which is simple in structure and easy to implement, but the discontinuous second derivative leads to unstable operation of the manipulator; Literature [3] proposes a combination of 5-7-5 polynomial and complex method, which improves efficiency and reduces energy consumption, but the calculation is relatively complicated, and higher-order orders are more likely to cause "The Runge Phenomenon"; Literature [4] aims to suppress mechanical vibration, combining genetic algorithm with polynomial trajectory planning; Literature [5] uses the weight coefficient method to form the objective function, and uses the particle swarm algorithm to plan the motion of the five-degree-of-freedom manipulator. Although the above article solves the problem that traditional algorithms are difficult to optimize high-order polynomials,
it does not consider the convergence rate of intelligent algorithms and the problem of falling into local optimum.

This paper takes the Mitsubishi RV-4FL-D manipulator as the research object, and proposes a 4-3-4 polynomial function to plan the motion trajectory of the manipulator based on the trajectory of the robot motion space and the actual working conditions of the sorting operation, so that the calculation process is simple, and The position, speed and acceleration of each joint are continuously controllable. The standard particle swarm algorithm has problems such as slow convergence speed in the early stage and easy to fall into local optimum in the later stage. In this paper, an improved particle swarm algorithm is used to change the learning factor in the standard particle swarm from a fixed value to a variable that can be dynamically adjusted according to the environment, and the improved particle swarm algorithm can converge quickly in the early stage and accurately converge to the optimal solution in the later stage. Experiments have proved that the trajectory planning method combining 4-3-4 polynomial function and improved particle swarm algorithm effectively improves the speed and accuracy of the solution, and ensures the comprehensive optimization of efficiency and stability.

2. The construction of polynomial interpolation function

In order to ensure overall continuous movement of the manipulator during operation and increase efficiency, according to the actual working conditions, the working interval of the manipulator is divided into three sections. In this paper, 4-3-4 polynomial interpolation function is used. In the first and third stages, a 4-degree polynomial is used for interpolation, and the second stage uses a third-degree polynomial interpolation.

The 4-3-4 polynomial interpolation function is:

\[ q_i(t) = a_{1i}t^4 + a_{2i}t^3 + a_{3i}t^2 + a_{4i}t + a_{5i} \]

\[ q_{2i}(t) = a_{1i}t^3 + a_{2i}t^2 + a_{3i}t + a_{4i} \]

\[ q_{3i}(t) = a_{1i}t^2 + a_{2i}t + a_{3i} \]  

(1)

Where \(i=1,2,3,...,n\), \(n\) is the number of joints, \(q_{1i}, q_{2i}, q_{3i}\) are the positions of the first, second, and third intervals corresponding to the \(i\)-th joint. All the coefficients in formula (1) can be calculated by the above constraints, as shown in formulas (2)-(4):

\[
A = \begin{bmatrix} B & 0 & 0 \\ C & D & 0 \\ 0 & E & F \end{bmatrix}
\]

(2)

\[
q = \begin{bmatrix} q_1 & 0 & 0 & q_2 & 0 & 0 & q_3 & 0 & 0 & q_4 & 0 & 0 \end{bmatrix}^T
\]

(3)

\[
a = A^Tq = \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix}^T
\]

(4)

3. Improved PSO algorithm

3.1. Kinematic constraints and fitness function

According to the physical limits of the kinematics and dynamic performance of the manipulator and the actual work requirements, the kinematic constraints of the manipulator can be determined. The maximum position of joint \(i\) that allows smooth operation is \(Q_{i,max}(\text{rad})\), and the maximum speed is \(V_{i,max}(\text{rad} / \text{s})\), then:

\[
\begin{align*}
|q_i(t)| & \leq Q_{i,max} \\
|v_i(t)| & \leq V_{i,max}
\end{align*}
\]

(5)

The purpose of the optimization in this paper is to increase the efficiency and stability of the manipulator on the basis of completing the task, so the shortest running time is used as the fitness function:
3.2. Improved PSO optimization algorithm

The speed represents the displacement of the particle in continuous iteration, and fitness value determines the quality of the solution. In the continuous iteration of particles toward better features, the individual extremum and global extremum are constantly updated. The update formula is as follows:

\[ v_{id}^{t+1} = \omega v_{id}^{t} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \]  

(7)

\[ x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \]  

(8)

Where \( x_{id} \) is the position of the particle, \( i \) represents an arbitrary particle in the particle swarm, and \( d \) represents the dimension; \( v_{id} \) represents the velocity of the above-mentioned particle at the corresponding position; \( p_{id} \) represents the optimal position of the above-mentioned particle under this condition; \( p_{gd} \) is the global optimal position under current conditions. \( \omega \) is inertia weight; the non-negative constants \( c_1 \) and \( c_2 \) are the learning factors, and \( r_1 \) and \( r_2 \) are uniformly distributed random numbers on [0,1].

In order to obtain the global optimal solution more efficiently, this paper adopts a dynamic method to assign values to \( c_1 \) and \( c_2 \). In the early stage of the search, the meaning of individual information is greater than global information, so the value of \( c_1 \) is greater than \( c_2 \). In the later stage of the search, in order to converge to the global optimum more quickly and accurately, the value of \( c_2 \) is greater than the value of \( c_1 \), and the constructed function is as follows:

\[ c_1 = 2 \sin^2 \left( \frac{\pi}{2} \left(1 - \frac{t}{T_{max}} \right) \right) \]  

\[ c_2 = 2 \sin^2 \left( \frac{\pi T}{2T_{max}} \right) \]  

(9)

Set the initial particle number \( M=50 \), the initial position is 0-1s random position, the flight speed is [-2,2], the inertia weight \( \omega=0.6 \), \( r_1, r_2 \) are random numbers on [0,1], the number of iterations is 50.

4. Modeling and simulation experiment

This article takes the Mitsubishi RV-4FL-D robot as the research object, and the experimental platform is shown in Figure 1. The standard D-H method is used to establish the kinematic coordinate system model: the axis direction of the rotation axis of the manipulator is the coordinate zi direction, the common perpendicular direction of the axis zi and zi-1 is the xi direction, and yi is determined by the right-hand rule. According to this method, the specific coordinate system is shown in Figure 2, and the D-H parameters obtained are shown in Table 1.

![Robot experimental platform](image1)

![Robot coordinate system](image2)

| Table 1. RV-4FL-D robot D-H parameters |
|-----------------------------------------|
| i | \( \alpha_i \) | \( \cos \alpha_i \) | \( \sin \alpha_i \) | \( a_i \) | \( d_i \) |
|---|---|---|---|---|---|
| 1 | -90° | 0 | -1 | 0 | \( L_1 \) |
In Table I, \( \alpha \) represents the degree of twist, \( a \) represents the length of the rod, and \( d \) represents the offset of the rod, where \( L_1=170\text{mm}, L_2=280\text{mm}, L_3=60\text{mm}, L_4=350\text{mm} \). Use Robotics Toolbox in MATLAB simulation software to model it, as shown in Figure 3:

![RV-4FL-D robot 3D model](image)

Given the initial position of the manipulator, the position of the path point 1, the position of the path point 2, and the end position in the Cartesian coordinate system, as shown in Table 2:

| Position angle | initial point | path point1 | path point2 | end point |
|----------------|---------------|-------------|-------------|-----------|
| J1             | -13.43        | -13.43      | -48.25      | -48.26    |
| J2             | -7.23         | -13.16      | 17.05       | 23.79     |
| J3             | 126.79        | 105.48      | 74.51       | 96.83     |
| J4             | 0             | 0           | 0           | 0         |
| J5             | 62.99         | 90.22       | 91.13       | 62.08     |
| J6             | 73.97         | 74.51       | 40.12       | 40.02     |

Through inverse kinematics, the initial point, path point, and end point are transformed from Cartesian space representation into joint space representation, as shown in Table 3:

| Position angle | initial point | path point1 | path point2 | end point |
|----------------|---------------|-------------|-------------|-----------|
| J1             | -13.43        | -13.43      | -48.25      | -48.26    |
| J2             | -7.23         | -13.16      | 17.05       | 23.79     |
| J3             | 126.79        | 105.48      | 74.51       | 96.83     |
| J4             | 0             | 0           | 0           | 0         |
| J5             | 62.99         | 90.22       | 91.13       | 62.08     |
| J6             | 73.97         | 74.51       | 40.12       | 40.02     |

In order to illustrate the advantages of this method, the optimal particle fitness curves of the improved PSO algorithm (IPSO) and the basic PSO algorithm (BPSO), GA algorithm and DE algorithm are compared as shown in Figure 4:

![Comparison of fitness curves](image)
It can be seen from Figure 4 that BPSO converges to 1.9707 at about 40 steps; the optimization result of GA is 1.7079, and the convergence step is about 30; the optimization result of DE is 1.7789, and the convergence is about 35; the optimization result of IPSO is 1.5973 and converged at about 25 steps. The improved PSO algorithm is better than other algorithms in both convergence speed and fitness value in the early stage, which shows that the improved algorithm has a better effect.

The improved PSO algorithm is used to optimize the 4-3-4 piecewise polynomial function with time as the objective. The iterative process of the optimal particle position of joint 1 is as follows:

For other joints, the above method is also used to optimize, so as to obtain the optimal running time of each joint, as shown in the following table:

| Joint | t1 | t2 | t3 |
|-------|----|----|----|
| J1    | 0.5698 | 0.6803 | 0.3445 |
| J2    | 0.7643 | 0.4851 | 0.5296 |
| J3    | 0.2067 | 0.7035 | 0.3561 |
| J4    | 0   | 0   | 0   |
| J5    | 0.2943 | 0.5160 | 0.4923 |
| J6    | 0.4831 | 0.5243 | 0.2493 |

In order to ensure that each joint of the manipulator reaches the target position at the same time, the maximum value of the interpolation time of all the joints is selected, so the manipulator running time is T=1.9974. The position, velocity and acceleration motion trajectory curves of joint 1 to joint 6 are shown in Figure 6-8 respectively.
From the analysis of the above simulation results, it can be known that the improved trajectory planning method can shorten the running time of the manipulator and ensure that the continuity of the planned displacement, velocity, and acceleration curves is within the ideal range. Therefore, the stable operation of the mechanical arm can be continuously ensured under the premise of shortening the time.

5. Conclusions
The trajectory optimization method proposed in this paper comprehensively considers the efficiency and stability of the manipulator, and proposes 4-3-4 polynomial interpolation for fitting, which greatly reduces the calculation while ensuring the continuous position, speed and acceleration of the manipulator. It solves the problems of the impact caused by the discontinuous acceleration in the literature [1] and [2] and the problem in the literature [3] that the calculation is complicated and "The Runge Phenomenon" is easy to occur. Compared with the use of genetic algorithm [4] and basic particle swarm algorithm [5], the improved PSO algorithm effectively solves the problem that high-order polynomials are difficult to optimize with traditional methods, and the dynamically adjusted learning factor can quickly and accurately converge to the optimal solution. The results show that using 4-3-4 piecewise polynomial combined with improved PSO algorithm to optimize manipulator motion trajectory, the trajectory running time is shortened by 40% compared with polynomial trajectory planning alone, and 20% compared with basic PSO trajectory optimization. The operation process of each joint is stable and the expected optimization goal is achieved.

Acknowledgments
Fund Project: Youth Science Foundation Project (No.61503045)

References
[1] Kim K W, Kim H S, Choi Y K. (2012) Optimization of cubic polynomial joint trajectories and sliding mode controllers for robots using evolution strategy. In: International Conference on Industrial Electronics. New Orleans. PP. 1465-1477.
[2] Klemens S, Hubert G, Peter S. (2013) On time-optimal trajectory planning for a flexible link robot. Journal of Systems and Control Engineering, 227(10): 752-763.
[3] Chen H, Li L. (2019) Trajectory Planning of Time Optimal Manipulator based on Complex Method. Mechanical Transmission, 43(03): 72-75+94.
[4] Lou J, Wei Y, Li Gl. (2016) Optimal trajectory planning of a flexible manipulator for its vibration suppression using genetic algorithm. Journal of Vibration and Shock, 35(11): 1-6.
[5] Rubio F, Valero F, Suner J L. (2009) A comparison of algorithms for path planning of industrial robots. In: 2nd European Conference on Mechanism Science. Berlin. PP. 247-254.