Paper Type: Original Article

Optimization of Joint Angle of Six-degree-of-freedom Robotic Arm Based on Multi-objective Genetic Algorithm

Huang Shuai[1*#], Chen Zipeng[1#], Lin Xiaoya[1##], Liu Xin[1##]

[1]College of Mechanical and Electrical Engineering、Quanzhou University of Information Engineering、Quanzhou、China

[#]Co-first Author: *Huang Shuai and Chen Zipeng are co-first authors.*

[##]Co-second Author: *Lin Xiaoya and Liu Xin are co-second authors.*

***Corresponding Author**: *Huang Shuai*, **E-mail**: 18093155093@163.com

abstract

In this paper, for the joint angle optimization problem of six-degree-of-freedom robotic arm, a path optimization method based on multi-objective genetic algorithm is proposed to enhance the robotic arm motion performance with the goal of minimizing the end error and energy consumption. Firstly, a forward kinematic model of the robotic arm is constructed based on the Denavit-Hartenberg (D-H) parameters, and the modeling and visualization in the zero-position state is realized through the MATLAB Robotics Toolbox, and the coordinates of the target point (1500mm, 1200mm, 200mm) are set. Then, the dual-objective optimization model including Euclidean distance of end position and motion energy consumption is established by combining the parameters of joint inertia (0.2-0.6 kg-m²) and average angular velocity (1.0-3.0 rad/s), and the non-dominated sorting genetic algorithm II (NSGA-II) is used to solve the Pareto optimal solution set. The optimization results show that the end error is as low as 0.0488 mm and the total energy consumption is reduced to 23.9282 J under the optimal configuration that takes into account both accuracy and energy consumption, which verifies the effectiveness of the method in robotic arm motion control. The research results provide theoretical reference and technical support for the optimization of robotic arm paths in industrial automation and precision operation.

Keywords: Six-degree-of-freedom Robotic Arm; Joint Angle Optimization; Multi-objective Genetic Algorithm; End Error; Energy Consumption Optimization

1 | Introduction

In the late 1960s, with the rise of industrial automation and the increasing demand for robots with higher flexibility and adaptability, the six-degree-of-freedom robotic arm, as a robotic structure capable of diversified motions and attitude control in three-dimensional space, soon attracted the interest of researchers. The earliest six-degree-of-freedom robotic arms were used in fields such as industrial assembly lines, welding and material handling, providing new possibilities for industrial production [1]. With the continuous development of technology and the improvement of mechanical structures, the application of six-degree-of-freedom robotic arms in the industrial and service sectors has been expanding and has become an important part of today's robotics. As an automated device consisting of multiple linkages and joints, robotic arms are widely used in industrial production, precision operations, hazardous environment operations, and logistics.

The motion state of the end of the robotic arm is a key factor affecting the quality of the robotic arm's work, which means that motion planning is a key component of robotic arm motion control [2]. Effective motion planning can improve the working efficiency and accuracy of the robotic arm, while mitigating the non-essential impacts in the motion, which can help to reduce the post maintenance cost and extend its service life. In this context, this paper aims to solve several key problems of robotic arms in practical applications by optimizing the design of joint angle paths of a six-degree-of-freedom robotic arm with the goal of minimizing the end error and energy consumption.

2 | Literature Review

The trajectory planning of the manipulator is carried out on the basis of kinematics. According to the different space where the trajectory is located, the trajectory planning algorithm can be divided into Cartesian space and joint space. Among them, in Cartesian space, the trajectory of the end of the manipulator is planned. The advantage of this algorithm is that the trajectory of the end of the manipulator can be observed intuitively, but its shortcomings are also more prominent. It needs to be calculated by inverse kinematics to transform it into the changes of each joint. When the manipulator needs to be controlled in real time, the trajectory planning in Cartesian space will take more time, and there are higher requirements for the performance of the computer, so it is not conducive to the realization of the real-time control of the manipulator. In the joint space, the angle, angular velocity and angular acceleration of each joint of the manipulator can be directly planned, which can directly control the rotation of each joint and ensure the synchronous operation of each joint motion. The calculation efficiency is high, and the real-time control of the manipulator can be better realized. By comparing and analyzing the advantages and disadvantages of the two plans, this paper mainly focuses on the trajectory planning of the joint space for the manipulator.

The trajectory planning algorithm in joint space is mainly divided into multinomial interpolation and spline interpolation. In Reference [3], the method of cubic polynomial is used to plan the trajectory. Although this method is simple and fast, the obtained angular velocity and angular acceleration curves are not smooth, which leads to the instability of the manipulator during operation, which will have a serious impact on the use of the manipulator. Zhang et al. [4]

combined the improved GA algorithm with quintic polynomial interpolation to obtain time-optimal trajectory planning. Li Xiaowei [5] combined PSO optimization algorithm with 3-5-3 polynomial interpolation to realize trajectory planning. Machmudah et al. [6] combined GA algorithm and PSO algorithm and carried out six polynomial interpolation planning to obtain the time optimal trajectory. Li et al. [7] used 5-7-5 polynomial interpolation to make the manipulator have good stability when it rotates at a large speed. Wang [8] used the B-spline method to make the trajectory planning take less time and achieve better results. Zou et al. [9] interpolated the manipulator through non-uniform rational B-spline, and the final trajectory accuracy was high, which could meet the motion requirements. Zhao Hao et al. [10] carried out trajectory planning based on cubic non-uniform B-spline, and the results were better under continuous path. Shi et al. [11] combined B-spline interpolation with S-shaped curve to plan the trajectory of the manipulator in multiple segments. The final result is that the trajectory is smoother and more stable. Huang et al. [12] performed multi-objective optimization based on the quintic B-spline method combined with genetic algorithm (NSGA-II), and finally planned the time-jump optimal trajectory.

3 | Methodology

3.1 Zero state six degree of freedom robot arm sketch

Firstly, this paper analyzes the initial parameters of the robot arm and draws the sketch of its six-degree-of-freedom robotic arm according to the given zero state (initial angle of each joint). Then the D-H parameters are applied to establish the forward kinematic model of the robot arm and calculate the position of the end-effector. Finally, the coordinates of the target point (1500mm, 1200mm, 200mm) are determined, and the optimal joint angle path is solved with the objective of minimizing the end error (i.e., the Euclidean distance between the actual position of the end-effector and the target position) and using optimization algorithms (e.g., Particle Swarm Optimization, Genetic Algorithm, etc.). Visualizations such as trajectory plots can be given using MATLAB's Robotics Toolbox.

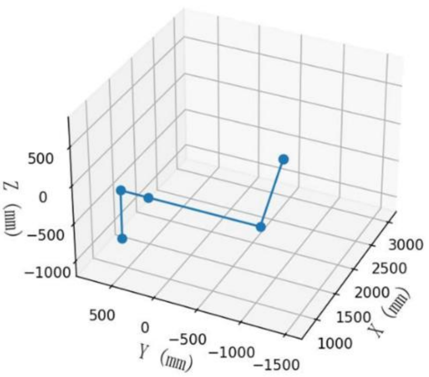


Figure 1 Zero state sketch of a

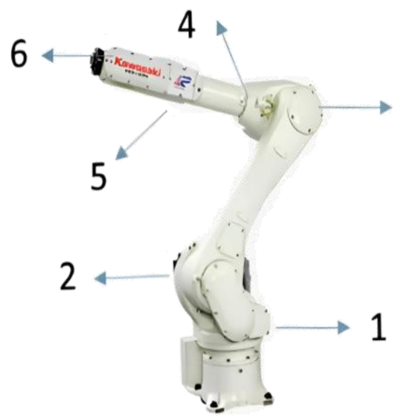


Figure 2 Six degree of freedom robot arm zero

six-degree-of-freedom robotic arm **position physical picture**

3.2 Mathematical modeling of robot arm motion

Modeling the kinematics of a six-degree-of-freedom robotic arm mainly involves defining the Denavit-Hartenberg (D-H) parameters, building the forward kinematics model, and solving the inverse kinematics model.

In the first step, we define the Denavit-Hartenberg (D-H) parameters. The D-H parameter method is a standard way of describing the relationship between the connecting rod and the joints of a robot arm, as defined in Table 1 of :

Table 1 Parameter specific definitions

parameters	hidden meaning	Specific definitions
a_i	Connecting rod length	Distance along the Xi axis from the Zi axis to the Zi+1 axis
a_j	Torsion angle of connecting rod	Angle of rotation from Zi axis along Xi axis to Zi+1 axis
d_i	linkage misalignment	Distance along the Zi axis from the Xi-1 axis to the Xi axis
θ_i	joint angle	Angle of rotation along the Zi axis from the Xi-1 axis to the Xi axis

In the second step, we model the positive kinematics. Positive kinematics modeling means that given each joint angle θ_i of the robot arm, the position and orientation of the end-effector are solved. The position and orientation of each joint can be passed to the end-effector through a transformation matrix. For each joint i, the following transformation matrix A_i can be used.

$$A_i = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & sin\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

By multiplying the transformation matrices for each joint, the end-effector's position matrix with respect to the base can be obtained T_0^n :

$$T_0^n = A_1 A_2 \ldots\ldots A_n$$

Where T_0^n is a 4x4 chi-square transformation matrix describing the position and orientation of the end-effector.

Finally, we determine the end-effector position. The end-effector position coordinates (x,y,z) are extracted from the T_0^n matrix:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} T_0^n[0,3] \\ T_0^n[1,3] \\ T_0^n[2,3] \end{bmatrix}$$

In the third step, we solve the inverse kinematics model. The inverse kinematics model is to solve for the angle θ_i of each joint, given the position and orientation of the end-effector. Inverse kinematics problems are usually complex and involve a system of nonlinear equations, which can be solved numerically or analytically.

3.3 Joint angle path optimization for robotic arms

The joint angle path optimization of the robotic arm is stepwise disassembled to obtain Table 2 :

Table 2 Steps and code description of the joint angle path optimization algorithm

algorithmic step	Code Description
Setting Chinese fonts	Use FontProperties to set the Chinese font path to ensure that the chart title and labels can be displayed correctly in Chinese;
Define the D-H parameters and the transformation matrix	The transformation matrix for each joint is defined using the D-H parameter;
Forward Kinematics Calculations	Calculate the position of the actuator at the end of the robotic arm;
objective function	Calculate the Euclidean distance between the target position and the end-effector position;
Optimization process	Optimize using the minimize function in scipy.optimize;
Results mapping	Plot the position of the optimized arm and update the title to "State sketch of the optimized six-degree-of-freedom robotic arm".

Running the code implementing the above algorithmic steps will generate a 3D graphic that can show the sketch of a six-degree-of-freedom robotic arm under the optimal joint angle path with correctly displayed Chinese titles and labels.

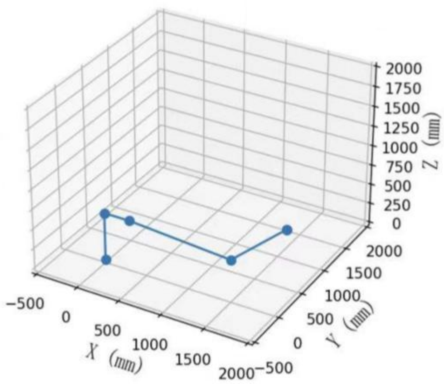


Figure 3 Sketch of the optimized state of the six-degree-of-freedom robotic arm

By visualizing the results obtained from Figure 3 , not only the optimal joint angle and the minimum end error are obtained, but also the position of the robotic arm under the optimal joint angle configuration can be seen through the 3D graphics. The 3D graphic shows the movement path of the robotic arm from the base to the end-effector to visualize and understand the movement state of the robotic arm.

3.4 Minimizing End Error and Energy Consumption Models

Combining the rotational inertia and average angular velocity of each joint mentioned above, the joint angle path of the robot arm is further optimized with the objective of minimizing the end error and energy consumption. In this paper, genetic algorithm is chosen to solve the model, where two objectives need to be considered simultaneously as minimizing the end error and minimizing energy consumption, so it is a multi-objective optimization problem.

Table 3 Joint rotation energy consumption parameters

Joint i	Moment of inertia (kg · m²)	Average angular velocity (rad/s)
1	0.5	2.0
2	0.3	1.5
3	0.4	1.0
4	0.6	2.5
5	0.2	3.0
6	0.4	2.0

Firstly, the energy consumption of the six-jointed robot arm moving to the end is calculated, taking into account the rotational inertia of each joint, angular velocity and the dynamics of the whole movement process. Then the kinetic energy of each joint is calculated, and the kinetic energy of the joints is calculated by the following equation.

$$E_{ktnettc} = \frac{1}{2} I \omega^2$$

Where I is the moment of inertia of the joint andω is the angular velocity of the joint. Then the kinetic energy of the whole arm is calculated and the kinetic energy of all the joints are added together to get the total kinetic energy of the whole arm:.

$$E_{total} = \sum_{i=1}^{6} \frac{1}{2} I_i a q^2$$

Next consider the change in gravitational potential energy, which is calculated based on the path of motion of the robotic arm. The equation for gravitational potential energy is.

$$E_{potential} = mgh$$

Where m is the total mass of the robotic arm and end load (5kg), g is the gravitational acceleration of 9.81m/s^2, and$h$ is the change in height. Finally, the total energy consumption is calculated to calculate the total energy consumption, which includes not only the changes in kinetic and potential energy, but also other factors such as friction, loss of control system and so on. In the ideal case, the energy consumption can be reduced to the sum of the changes in kinetic and potential energy.

$$E_{total} = E_{kinetic} + E_{potential}$$

Assuming that the rotational moment of inertia of the ith joint is I and the angular velocity is ωi, the kinetic energy of the ith back-off is.

$$E_{kindic}, i = \frac{1}{2} I_i a q^2$$

Total Kinetic Energy Calculation.

$$E_{total,\, kinetic} = \sum_{i=1}^{6} \frac{1}{2} I_i \omega_i^2$$

3.5 Optimized design of joint angle paths

The goal of optimization in this model is to find a balance between these two objectives. The error objective function goal is to minimize the Euclidean distance between the end-effector position and the target point.

$$error = \sqrt{\left(x(\theta) - x_{target}\right)^2 + \left(y(\theta) - y_{target}\right)^2 + \left(z(\theta) - z_{target}\right)^2}$$

Where $\left(x(\theta), y(\theta), z(\theta)\right)$ is the end-effector position obtained by forward kinematics calculation and $\left(x_{target}, y_{target}, z_{target}\right)$ is the target point position. The goal of the energy consumption objective function objective is to minimize the total energy consumption of the robotic arm motion, including the changes in kinetic and gravitational potential energy.

$$energy = \sum_{i=1}^{6} \frac{1}{2} I_i \omega_i^2 + mg\left(z_{final} - z_{initial}\right)$$

Where I_i is the rotational inertia of the individual joints,ω_i is the average angular velocity of the individual joints, m is the total mass of the robotic arm and the end-station load, g is the gravitational acceleration, and z_{final} and $z_{initial}$ are the heights of the robotic arm at the initial and final positions, respectively.

Combining these two objective functions, the formulation of the multi-objective optimization model can be expressed as.

$$\min_{\theta}\left(error(\theta), energy(\theta)\right)$$

Where θ is the angle vector of each joint of the robotic arm. This optimization problem is solved by a genetic algorithm, which ultimately yields a set of Pareto optimal solutions, each finding a trade-off between error and energy consumption. Then there is the joint angle range constraint:the angle of each joint must be within its allowable range.

The principle of solving the multi-objective optimization problem using genetic algorithm is then as follows:

A population is first initialized and each individual (chromosome) is a candidate solution. For a multi-objective optimization problem, each chromosome consists of a set of decision variables (e.g., the joint angle θ of the robotic arm).

$$P(0) = \{\theta_1, \theta_2, \dots\dots\dots, \theta_N\}$$

where N is the population size and each 0; is within the constraints of the decision variables. The fitness of each individual in the population is then calculated. For a multi-objective optimization problem, the fitness function is a vector representation of the objective function.

$$fitness(\theta) = [f_1(\theta), f_2(\theta), \dots\dots, f_M(\theta)]$$

where M is the number of objective functions. Then the individuals with high fitness (usually non-dominated solutions) are selected for reproduction based on the fitness value. Commonly used methods include Roulette Wheel Selection (RWS), Tournament Selection (TSS), and so on.

$$P_{selected} = Selection(P)$$

Afterwards, crossover operations are performed on the selected individuals to generate new individuals. Commonly used methods are Single-point Crossover, Two-point Crossover chips.

$$\theta_{new} = Crossover(\theta_{perent1}, \theta_{parent2})$$

The newly generated individuals are then subjected to mutation operations to increase the diversity of the population. Commonly used methods include Unifomn Mutation (Uniform Mutation), Non-uniform Mutation (Non-uniform Mutation), and so on.

$$\theta_{matated} = Mutation(\theta_{new})$$

Finally, the newly generated individuals are added to the population to form a new generation of population.

$$P(t+1) = \theta_{mutated} \cup P_{salacted}$$

Repeat the above steps until the termination conditions (e.g., maximum number of generations, convergence of fitness values, etc.) are satisfied.

4 | Results and Discussion

4.1 Joint angle path optimization for robotic arms

The position of the end-effector can be calculated by the matrix multiplication method in this paper, which can complete the kinematic modeling of the robotic arm and obtain the position and attitude of the end-effector in the base coordinate system.

Table 4 Robotic arm initial Denavit-Hartenberg (D-H) parameters

Joint i	$a_{i\text{-}1}$/(mm)	$a_{i\text{-}1}$/(。)	d_i/(mm)	Range of variation of θ_i joints/(。)
1	0	0	600	-160~160

2	300	-90	0	-150~15
3	1200	0	0	-200~80
4	300	-90	1200	-180~180
5	0	-90	0	-120~120
6	0	-90	0	-180~180

The results of the optimal joint angles are shown in Table 5 below, which enables the end-effector of the six-degree-of-freedom robotic arm to reach the target positions (1500mm, 1200mm, 200mm):

Table 5 Optimum angle for each joint

Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
45.9989839°	-45.99598394°	4.68158374°	84.54076205°	-90.0°	0.0°

These angles are optimized by an optimization algorithm to bring the end-effector position as close as possible to the target position. With a minimum end error of 0.00281491582969362mm, the end-effector reaches the target position almost exactly (1500mm,1200mm,200mm) with the optimized joint angle configuration. The optimal joint angles represent the best angle configurations required to bring the robotic arm end-effector to the target position. With these angular adjustments, the robotic arm can accurately accomplish the gripping task. The minimum end error indicates that the optimized configuration almost completely meets the target requirements, validating the effectiveness of the optimization process.

4.2 Minimization of end-of-pipe errors and energy consumption optimization

The genetic algorithm optimization iteration process is firstly given as a diagram:

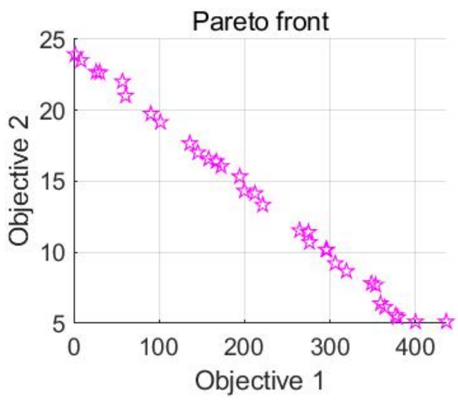


Figure 4 Pareto Frontier Diagram

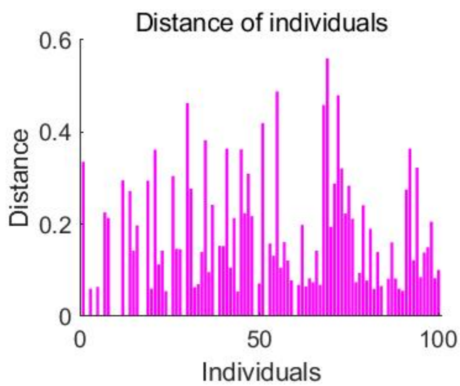
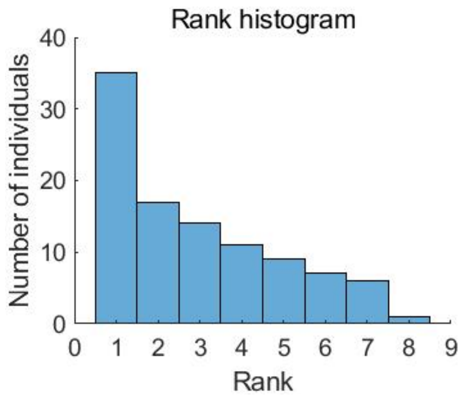


Figure 5 Pareto Distance Chart



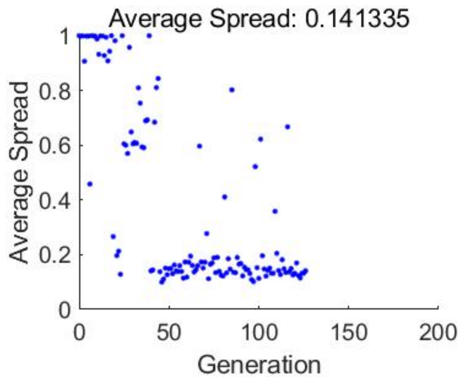


Figure 6 Pareto Ranking Histogram　　**Figure 7 Pareto distribution map**

It can then be derived that the Pareto frontier graph (gaplotpareto`) shows the variation of the Pareto frontier during the optimization process. Each point represents a solution, and the horizontal and vertical coordinates correspond to the values of the two objective functions (error and energy consumption). As the number of iterations increases, the number of points on the Pareto front increases and the distribution becomes progressively denser, indicating that the algorithm finds a set of well-balanced solutions while exploring the solution space. These solutions achieve different trade-offs between error and energy consumption, forming a Pareto-optimal set of solutions.The Pareto distance graph (gaplotparetodistance`) shows the distance between an individual in the population and its nearest neighbor in each generation. The distances shown in the graph reflect the diversity of solutions in the population. Larger distances indicate that individuals in the population are more dispersed, which helps to avoid locally optimal solutions; smaller distances indicate that individuals are more concentrated, which may mean that the algorithm is converging.

By observing the change in distance, the population diversity and the speed of convergence of the algorithm can be assessed. The ranking histogram (gaplotrankhist") shows the distribution of the non-dominated ranking of individuals in the population. The horizontal coordinates indicate the different non-dominated ranking ranks and the vertical coordinates indicate the number of individuals in each rank.

Ideally, as the number of iterations increases, the number of individuals in the high-ranking (low-numbering) class gradually increases, which means that the algorithm is gradually finding more non-dominated solutions that are of higher quality. The distribution graph (gaplotspread') shows the distribution of solutions in the population in the goal space. By calculating the distance between the maximum and minimum values of the solutions in the population in the target space, the distribution range of the solutions can be evaluated. A larger distribution range indicates that the solutions in the population cover a larger objective space, which helps to find the global optimal solution; a smaller distribution range can indicate that the algorithm is converging and the diversity of solutions is reduced.

The final optimized minimized end error is 0.048823mm; while the minimum energy consumption is: 23.9282J; the optimal joint angles are shown in Table 6 :

Table 6　Optimum angle for each joint

Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
11.8092°	-11.8096°	4.78363°	84.4236°	-10.6945°	47.5971°

4| Conclusion

This paper focuses on the joint angle optimization problem of a six-degree-of-freedom robotic arm, which is investigated with the dual objectives of minimizing the end error and energy consumption. Firstly, the forward kinematic model of the robotic arm is constructed based on the D-H parameter, and the modeling and visualization of the robotic arm in the zero-position state is realized through the MATLAB robotics toolbox. Following that, a joint angular path optimization method based on multi-objective genetic algorithm (NSGA- II) is proposed, taking the Euclidean distance of the end position and the motion energy consumption as the optimization objectives, and combining the parameters of rotational inertia and angular velocity of each joint to establish a comprehensive optimization model. The Pareto optimal solution set is obtained through iterative solving of the algorithm, and finally a joint angle configuration that takes into account both precision and energy consumption is obtained, and the end error after optimization is as low as 0.0488mm, and the energy consumption is reduced to 23.9282J, which verifies the effectiveness of the method. The results provide theoretical references for the efficient motion control of robotic arms in precision operation, industrial automation and other scenarios.

Although this paper improves the comprehensive performance of the robotic arm through multi-objective optimization, there are still some limitations. On the one hand, the study only considered the kinetic and potential energies under ideal conditions, and did not fully incorporate the effects of actual factors such as friction and mechanical loss on the energy consumption model; on the other hand, the optimization process assumed a constant joint angular velocity, which is at variance with the variable-speed characteristics in the dynamic motion scenario. Future research can be expanded in the following directions: first, introducing a more accurate dynamics model, combining the friction coefficient, transmission efficiency and other parameters to improve the accuracy of the energy consumption calculation; second, exploring the dynamic path planning method, and investigating the impact of joint angular velocity changes on the multi-objective optimization results; third, applying the algorithm to the physical robotic arm for experimental validation, and further optimizing the parameter configurations to enhance the applicability of the project. In addition, we can try to combine intelligent algorithms such as reinforcement learning to improve the robustness and adaptive ability of the robotic arm movement in complex environments.

References

[1] Yue Jian; Duan Luck;. Application of industrial robots in automated production line of machinery manufacturing[J]. Internal Combustion Engines and Accessories,2024,No.399(03).

[2] Song Y ; Zhang Lei ; Tian Rong ; Wang Xiaohua ;. Research on obstacle avoidance motion planning method for robotic arm under complex multi-scenes[J]. Journal of Northwestern Polytechnical University,2023,v.41;No.201(03).

[3] Kim K W, Kim H S, Choi Y K, et al. Optimization of cubic polynomial joint trajectories and sliding modecontrollers for robots using evolution strategy [C]/. /Proceedings of the

IECON'97 23rd InternationalConference on Industrial Electronics, Control, and Instrumentation, November 14,1997, New Orleans, LA, USA. New York:IEEE,c2002:1444-1447.

[4] Zhang J, Meng Q, Feng X, et al. A 6-DOF Rotbot-time Optimal Trajectory Planning Based on anImproved Genetic Algorithm[J].Robotics and Biomimetics,2018 ,5(1):1-7.

[5] LI Xiaowei,HU Likun,WANG Kun. Time-optimal trajectory planning of six-degree-of-freedom robotic arm with PSO under velocity constraint [J]. Journal of Intelligent Systems,2015,10(03):393-398.

[6] Machmudah A, Parman S, et al. Polynomial joint angle arm robot motion planning in complexgeometrical obstacle [J].Applied Soft Computing,2013,13(3). 1099-1109

[7] Li C, Qu Y, Shan L, et al. Research on Path optimization algorithm of the 6-DOF manipulator[C]//2019Chinese Automation Congress (CAC).IEEE,2019:3135- 3140.

[8] Wang, You-Min. B-spline curve trajectory planning for robotic arm joint space[J]. Journal of Anhui Institute of Mechanical and Electrical Engineering,2000(02):21-26.

[9] Zou, Qingxiao. A Novel Robot Trajectory Planning Algorithm Based on NURBS Velocity AdaptiveInterpolation[C].Advances in Mechanical Design- Proceedings of the 2017 International Conference onMechanical Design,ICMD 2017,1191-1208.

[10] ZHAO Hao,LIU Manlu,ZHANG Hua. 6DOF robotic arm trajectory planning based on cubic non-homogeneous B-splines[J]. Chinese and foreign technology application,2016,(6):43-48.

[11] SHI Buhai,SUN Huihui. Research on B-spline curve algorithm based on new S-type speed planning[J]. Machine Tools and Hydraulics,2016,44(15):72-79.

Huang J, Hu P, Wu K, et al. Optimal time-jerk trajectory planning for industrial robots [J].Mechanismand Machine Theory, 2018,121:530-544.