Research Article

Yicen Xu*

Optimal trajectory planning and control of industrial robot based on ADAM algorithm of nonlinear data set

https://doi.org/10.1515/pjbr-2022-0126
received August 1, 2023; accepted November 6, 2023

Abstract: In order to better realize the optimal trajectory planning and trajectory control in industrial robots, a method based on ADAM algorithm is proposed. Taking PUMA 560 industrial robot as the research object, using nonlinear data sets and mathematical ADAM algorithm function planning, an optimal calculation method for time trajectory planning of industrial robot is explored. Finally, the programming, optimization, and simulation of the program code are implemented using MATLAB, and a standardized optimal trajectory planning is established. The experimental results show that the running time difference of the trajectory corresponding to the three joint points is small. In order to synchronize the position of each joint point in time, it is necessary to choose the optimal joint point position according to the time trajectory, so as to ensure the synchronization between each key node. Therefore, the joint node position is adjusted so that the total time and the final simulation results are basically synchronized in time, and both are 10.35 s. It proves that the improved ADAM algorithm realizes the trajectory optimization of industrial robots in terms of time planning, which can make the various joints of industrial robots basically synchronized in the time trajectory.

Keywords: ADAM algorithm function, trajectory optimization, PUMA560

1 Introduction

With the continuous development of modern science and technology, the application of robotics has become the most important achievement in the field of automation and control in this century. Since the United States developed the world's first industrial robot in the 1960s, robot technology and its products have made considerable progress and development, and its application has become more and more far-reaching and extensive, and has become a necessary tool in computer-integrated manufacturing systems, factory automation systems, and flexible manufacturing systems [1]. The industrial robot is not only a simple substitute for manual labor, but also an intelligent mechanical device formed by combining the unique expertise of humans and machines; it has a human-like operation, automatic control, and repeated programming, at the same time, electromechanical integrated production equipment that can complete various operations in three-dimensional space; therefore, it is especially suitable for flexible production of various types of variable batches and also plays a significant role in improving product quality, improving labor conditions, improving production efficiency and rapid product upgrading. In addition, in industrial production, it can also replace humans to complete some monotonous, repetitive, and even some high-risk and harsh environment-related work. In practical production applications, the motion of industrial robots should be stable, and the impact and vibration should be minimized, so as to improve the working accuracy and working life of industrial robots more effectively. Therefore, trajectory planning for robots is the key and foundation to solve such problems. With the continuous development of industrial robot trajectory planning, the optimal trajectory planning of multi-joint industrial robots has become a highly complex nonlinear optimization problem [2,3]. Time-optimal trajectory planning has been applied in many fields of industrial robotics and automation. At present, most industrial robots adopt the trajectory planning method based on trapezoidal speed to track the trajectory of a fixed path, which has some disadvantages such as large calculation amounts and long planning time. Although the problem of optimal time trajectory planning under geometric path constraints has been basically solved in theory and practice, the efficiency and adaptability of the algorithm need to be improved under dynamic constraints.

* **Corresponding author: Yicen Xu,** School of Mechatronic Engineering, Wuxi Institute of Commerce, Wuxi, Jiangsu 214153, China,
e-mail: YicenXu3@163.com

At present, there are many scholars at home and abroad who study the trajectory planning of such robots, and their research directions and starting points are also different. According to different optimization objectives, optimization is generally divided into the following three situations: Optimization based on time optimization, optimization based on energy minimization, and optimization based on pulsation minimization. Relatively speaking, the research on robot trajectory planning algorithms based on time optimization is the most. In the specific operation of trajectory planning, after the completion of specific optimization objectives, it is necessary to further select the planning space through application conditions, generally, according to the different planning space, the trajectory planning problems of robots can be divided into two categories: In joint space and Cartesian space [4], there are currently three ways to solve the time-optimal problem. One is dynamic planning, which divides the phase plane equally into grids, and then uses dynamic planning to find the minimum time trajectory on the phase plane. The second is the convex optimization method, which discretizes the position axis into n segments, and then transforms the temporal optimum problem into n variables, n equality, and inequality constraints, mainly using the existing efficient convex optimization computing package. The last category is the numerical integration method, which is based on the Pontryagin maximum principle: the optimal time trajectory is the "Bang-Bang" trajectory in the phase plane, and the velocity curve is obtained through the continuous integral acceleration. This method is theoretically faster than the previous two algorithms, but it is easy to cause dynamical singularity problems.

Based on this result, this article presents a method based on the ADAM algorithm. This article takes the trajectory planning of industrial robot as the research direction, analyzes the working principle of the ADAM algorithm and the optimal time planning of industrial robots, takes the method of PUMA560 industrial robot as the research object, and uses MATLAB to implement program code programming, optimization, and simulation to establish a standardized optimal trajectory scheme. The joint node position is adjusted so that the total time, and the final simulation results are basically synchronized in time, both 10.35 s. The improved ADAM algorithm realizes the trajectory optimization of the industrial robot in terms of time planning, which makes each joint point of the industrial robot basically synchronized in the time track. From the practical point of view, the research on trajectory planning of industrial robots with the goal of time optimization has very important guiding significance for the practical application of industrial robot technology.

2 Research methods

2.1 Mechanical and manual mechanical model

After decades of research and development, many gratifying achievements have been made in the control field of manipulators (industrial robots), and control methods have also emerged in endlessly. The most classical PID control is still widely used in some simple systems, but its performance is relatively poor in complex systems. This requires an accurate mathematical model of the manipulator to improve its performance. The neural network algorithm is widely used in the control of manipulators, especially in the track tracking calculation of manipulators, integrating it with the corresponding control method can meet the requirements of trajectory and tracking of the manipulator. Therefore, the research of adaptive trajectory tracking algorithms based on RBF neural networks has great practical significance [5,6].

Without considering external interference, for a DOF manipulator, the Lagrange method is used to establish its dynamic equation as follows (1):

$$M(q)\ddot{q} + V_{\mathrm{m}}(q, \dot{q})\dot{q} + G(q) = \tau. \tag{1}$$

In the actual system, it is difficult to obtain (1) the dynamics model with medium and high accuracy. In the process of considering uncertain factors, it is necessary to cover a series of factors that are not taken into account or intentionally omitted when modeling. In the process of designing the actual mechanical manual control system, the above two factors must be considered to improve its working performance by improving the accuracy of the system [7]. This method belongs to the inverse dynamics control strategy. The complete robot dynamics model is as follows (2):

$$M(q)\ddot{q} + V_{\mathrm{m}}(q, \dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_{\mathrm{d}} = \tau. \tag{2}$$

Generally speaking, (1) and (2) are referred to as the nominal and actual systems of the manipulator, respectively.

2.2 Design of trajectory tracking controller

Let $q_{\mathrm{d}}(t) \in R^n$ be the ideal trajectory in the workspace, and define as follows (3) and (4):

$$e(t) = q_{\mathrm{d}}(t) - q(t), \tag{3}$$

$$s(t) = \dot{e}(t) + \wedge e(\dot{t}), \tag{4}$$

Where $\Lambda = \Lambda^t \in R^{n\times n}$ is a positive definite matrix. According to the above formula, the robot dynamic filter tracking error is the formula (5):

$$M\dot{s}(t) = -V_{\mathrm{m}}s - \tau + h(x) + F(\dot{q}) + \tau_{\mathrm{d}}. \tag{5}$$

2.2.1 Neural network

Adaptive control and robust control are used to improve the adaptability of the robot visual servo system to these uncertain factors. RBF neural network is used to compensate for the uncertain part, and the following formula (6) can be obtained:

$$\tilde{h}(x) = W^T\zeta(x) + \in(x). \tag{6}$$

When N is in an infinite state, the neural network reconstruction error will be in an infinitesimal state. For $\in_N > 0$, $\| \in (x)\| < \epsilon_N$. The vector field $\zeta(x)$ is a Gaussian function. The following formula (7) can be obtained by dividing the matrix points:

$$\zeta(x) = \exp\left(-\frac{\|x - c_i\|^2}{\sigma_i^2}\right)1, 2, \ldots, N. \tag{7}$$

The center position vector $c_i \in R^{5n}$ of the RBF Gaussian function and the width vector $\sigma_i \in R$ of the Gaussian function are predetermined, the local search ADAM algorithm can select c_i and σ_i. The mechanical and manual mechanical equation can be converted into the following equation (8):

$$M\dot{s}(t) = -V_{\mathrm{m}}s - \tau + \hat{h}(x) + F(\dot{q}) + \tau_{\mathrm{d}} + \in(x). \tag{8}$$

2.2.2 Adaptive constraints

According to formulas (2) and (4) and the reconstruction error constraint $\in_N$ of neural network, formula (9) can be obtained:

$$\|F(\dot{q}) + \tau_d + \epsilon(x)\| \le a + b\|\dot{q}\| + c + \epsilon_N. \tag{9}$$

$\beta = a + b\|\dot{q}\| + c + \epsilon_N$ is defined as an adaptive constraint, or can be written as the following formula (10):

$$\beta = Q^T(\|\dot{q}\|)\varphi. \tag{10}$$

For the fixed positive parameter k, $Q \in R^k$ is a vector function of known joint velocity, and $\phi \in R^k$ is a parameter vector. According to formulas (8)–(10), the control torque input is proposed to achieve the desired trajectory. In adaptive control, the unknown parameters of the controlled object need to be estimated online and adjusted gradually to continuously improve the control performance of the system until the goal of gradual error convergence is achieved [8–10]. The following formula (11):

$$\tau = \hat{h} + Ks + \hat{W}^T\zeta(x) + \frac{\hat{\beta}^2 s}{\hat{\beta}\|s\| + \delta}. \tag{11}$$

2.3 CNN and Adam optimizer

2.3.1 CNN

CNN is essentially an input-output mapping. Through training, CNN will automatically obtain this mapping ability without the need to derive accurate algebraic expressions. Since the weights of neural modules on the same feature mapping surface are the same, the network can be learned in parallel mode, which is also a major advantage of convolutional neural network over other network models [11,12].

Using the back-propagation algorithm and supervised training method to train the convolutional neural network, compare the output result of the network with the preset label, and calculate and output the error term. According to the idea of back propagation, the error is transferred to each node layer by layer, and the weight value is updated. Through continuous iterative training, the error term of the network will be smaller and the weight update range will be smaller and smaller. When the weight value gradually becomes stable, the network training task is completed.

Images have their representative characteristics. After learning some features from a certain area of the image, these features can be used as detectors and extended to all areas to obtain the activation values of different areas. The purpose of the convolution operation is to extract the input features of the sample data, the first convolution layer usually only extracts some primary features, such as edges, lines, corners, and other basic levels, while the multi-layer convolution neural network will extract more complex and critical features.

The structure of convolution neural network model is shown in Figure 1.

In convolution neural network, image data can increase the number of training sets and improve the characteristic dimension of data after convolution processing, but it may lead to the occurrence of dimension disaster. In order to improve this problem, it is necessary to aggregate and pool the feature images obtained by convolution. Pooling can effectively reduce the resolution of the output feature map, while still maintaining the features at high resolution. After the pooled image data is processed through the fully connected network, the local features extracted previously

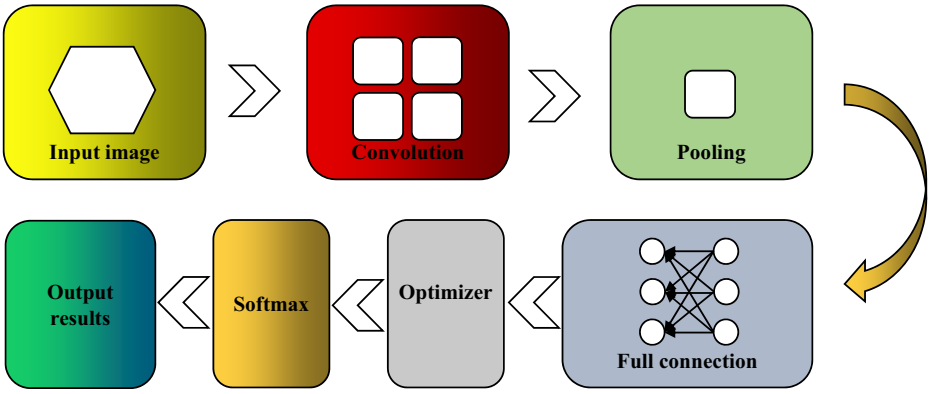

Figure 1: Structure of convolution neural network model.

can be recombined into a complete image using the weight matrix. Each nerve module node in the full connection layer is connected with the nerve module node in the feature map output from the previous layer. Second, the optimizer is used to process the processed image of the fully connected network to update and calculate the network parameters that affect the model training and model output, so as to make it approximate or reach the optimal value. Finally, the data are passed through the softmax classifier and output the corresponding classification results [13,14].

2.3.2 Adam optimizer

The Adam optimizer can still guarantee high accuracy for nonlinear separable and high-dimensional data sets. Because image data also have the characteristics of high dimension and nonlinear separability, Adam optimizer has obvious advantages over neural network in recognition applications. However, when the training sample dataset is large, the training time complexity of the Adam optimizer will become very high, and the traditional data processing architecture can no longer meet the requirements. The rise of distributed computing platform provides technical support for tasks requiring large amount of computation, and makes it possible for Adam optimizer to process large data sets.

The Adam optimizer algorithm iteratively updates the weights of the neural network based on the training data, and performs a stepwise optimization of the random objective function. The diagonal scaling of the gradient of Adam algorithm is invariable, which is suitable for solving the non-stationary problems with large data or parameters and large noise and sparse gradient. The basic algorithm of the Adam optimizer can be described as follows.

Set the noise target function $f_t(\theta)$, which is a random function of the parameter θ in the t period (the t iteration). In order to reduce the expected size of the function, it is necessary to use randomness to describe the noise of the small batch sample function and calculate the gradient of the objective function with respect to parameters, as shown in formula (12):

$$g_t = \nabla_\theta f_t(\theta). \tag{12}$$

The expressions of the exponential moving mean m_t and the square gradient v_t of the gradient in the t period are as follows (13):

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2. \end{aligned} \tag{13}$$

The parameter $\beta_1, \beta_2 \in [0,1)$ represents the decay rate of the moving average index. When the initial time and decay rate are very small, the moment estimation will be biased to 0.

In order to eliminate the initialization deviation, it is generally necessary to correct the deviation of the exponential moving mean value and the square gradient respectively during the attenuation process, the expressions of the corrected exponential moving mean $\hat{m}_t$ and the square gradient $\hat{v}_t$ are as follows (14):

$$\begin{aligned} \hat{m}_t &= m_t/(1-\beta_1^t), \\ \hat{v}_t &= v_t/(1-\beta_2). \end{aligned} \tag{14}$$

$\hat{m}_t/\sqrt{\hat{v}_t}$ is the signal noise ratio (SNR), which represents the ratio of signal to noise in the system. When the SNR value is small and Δt tends to infinity, the objective function will converge to the extreme value.

If the initial mean square gradient is 0, the update expression of the mean square gradient in the first phase is as follows (15):

$$\begin{aligned} v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \\ &= (1-\beta_2) \sum_{i=1}^{t} \beta_2^{t-i} g_i^2. \end{aligned} \tag{15}$$

At each iteration step, the value of parameter θ should be updated, and the update expression of θ is as follows (16):

$$\theta_t = \theta_{t-1} - \eta \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon), \tag{16}$$

where η is the learning rate, representing the magnitude of the effective step in the parameter space; $\varepsilon = 10^{-8}$ represents a constant parameter. Through parameter updating, the algorithm iteration is realized, and the objective function gradually converges to the optimal value. In the Adam optimizer algorithm, the estimation of the first moment to the non-central second moment is modified to reduce the offset; however, in the classification processing of complex and large-scale electron microscope images, the iterative curve of the algorithm oscillates violently and the convergence performance is poor [15].

2.4 Industrial robot trajectory planning based on optimal time angle

2.4.1 Constraint planning

The motion trajectory of the industrial robot is based on the specific key node of the robot motion, the Cartesian space coordinates are defined for the key point (q), the time series corresponding to the key point is defined as t, let h be the time interval between two time nodes, then the following formula (17):

$$\begin{gathered} h_i = t_{i+1} - t_i \\ T(h_i) = h_1 + h_2 + \cdots + h_{m-1} = \sum_{i=1}^{m-1} h_i. \end{gathered} \tag{17}$$

$T(h_i)$ here refers to the total running time of the industrial robot from the starting point to the end point, and the industrial robot needs to achieve time synchronization at each key node during its re-movement. Therefore, in order to achieve the optimal time planning of the robot, all joints of the robot need to be synchronized, so as to reduce the movement energy consumption caused by the complex trajectory.

Therefore, it is necessary to carry out kinematic constraints on each joint of the industrial robot. In order to bring convenience to each key motion path of industrial robot, it is necessary to describe the upper and lower limits of joint position in the way of joint constraint; the absolute value of the maximum acceleration of the joint is calculated according to the maximum speed of the joint. The absolute value of the maximum second order velocity of the industrial robot joint is calculated according to the maximum acceleration of the joint. Therefore, when solving the trajectory position corresponding to the robot joint point, it is necessary to calculate the limit value of the *i*th segment of the *j*th joint node and compare it with the end point of the segment to obtain the maximum and minimum limit values, save value. In the Buffer array, after the cyclic calculation of ADAM algorithm, the maximum and minimum values of the trajectory position of the *j*th joint node are obtained [16]. The specific process of solving the trajectory position is shown in Figure 2.

2.4.2 Solve the optimal solution of the trajectory function

In the process of solving the trajectory of industrial robots, the maximum value of different trajectory functions can be solved according to the trajectory operation function, and the maximum value of the trajectory function can be determined according to the complex trajectory function using the extremum trajectory solution method. The solution process of solving the trajectory function with Matlab is as follows:

2.4.2.1 Solution of function extremum locus

Because the solution function of the trajectory belongs to the category of artificial intelligence, and the definition of its mathematical function belongs to the high-order polynomial, the limit value of the application position can be obtained by using Matlab to analyze the solve() function.

2.4.2.2 Solution of the trajectory velocity of function extremum

The speed function of industrial robot's trajectory is also a high order polynomial, so when it is solved by Matlab, the limit value expression (three extreme points) of the trajectory speed function is obtained by using three root expressions.

2.4.2.3 Solution of acceleration of function extreme point trajectory

The second-order function of the motion trajectory of the articulated arm of the industrial robot is established, namely, Parabolic function, and the function vertex expression corresponding to the extreme point is obtained. Before determining the extreme value of the joint point function, the relevant extreme points were checked and brought them into the extreme value of the track function [17].

2.4.3 Coding mode

Without the commonly used binary encoding, the binary encoding of adjacent integers may have a large Hamming

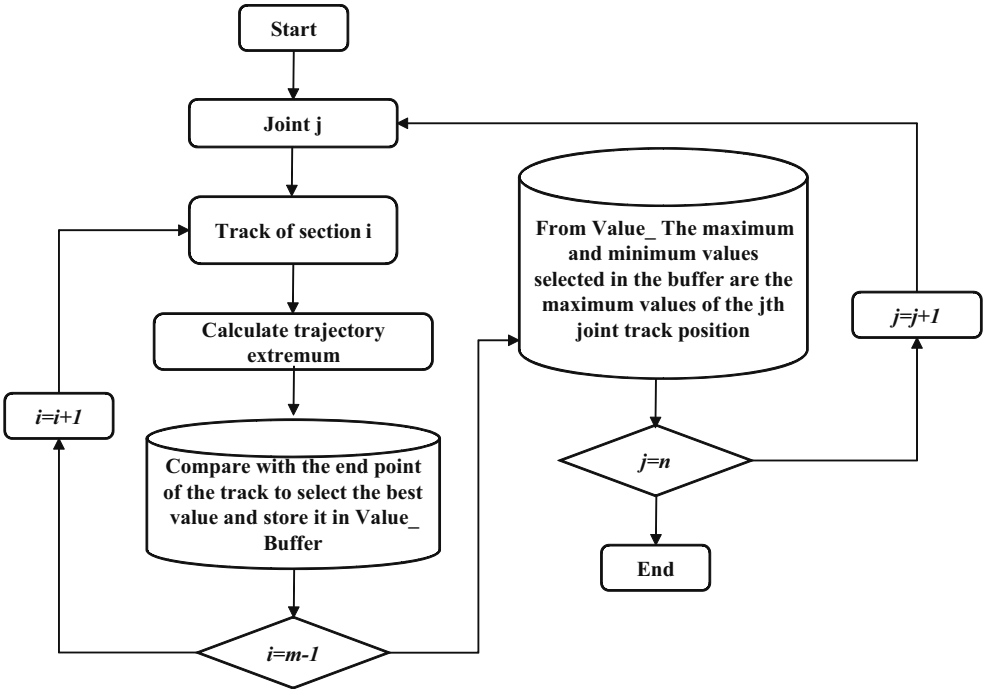

Figure 2: Specific process for solving the trajectory position.

distance, and the real number encoding method is adopted, that is, each gene value of an individual is represented by the real number within a specific range. This method is easy to deal with multi-dimensional and high precision problems, but also improves the computational complexity and improves the operation efficiency of the algorithm.

2.4.4 Optimal individual retention

The basic method is to choose according to the proportion of individual fitness. In this paper, the individual with the highest fitness is directly copied to the next generation, so that the individual with the highest fitness, namely the best individual, does not exchange and vary in this generation. This method accelerates the search speed and improves the local search ability.

3 Result analysis

3.1 Stability analysis

The robot dynamics of equation (2) is the control input of equation (11), and the whole system is asymptotically stable, when $t \to \infty$, the tracking error $s(t)$ and the subsequent tracking error $e(t)$ tend to zero. The following formulas (18) and (19):

$$\dot{\hat{W}} = \Gamma_W \zeta(x) s^T, \tag{18}$$

$$\dot{\hat{\varphi}} = \Gamma_\varphi Q \|s\|, \tag{19}$$

where $\Gamma_W = \Gamma_W^T \in R^{N\times N}$ and $\Gamma_\varphi = \Gamma_\varphi^T \in R^{k\times k}$ are positive definite matrices.

After summarizing and analyzing the above simulation model and simulation results, a 2-joint robot system is shown in Figure 3.

After a period of time, the characteristic points of the end effector converge to the ideal trajectory. For trajectory planning, the length of simulation time has no effect on the final planned trajectory, as shown in Figure 4. It can be seen that they all converge to the true value, which proves the effectiveness of the algorithm [18].

3.2 Time optimal trajectory planning simulation

In order to determine the effectiveness of the ADAM algorithm encoded in this project, Matlab simulation is used to realize the data analysis of the ADAM algorithm results, and the optimal trajectory information of the key nodes of the industrial robot trajectory is calculated through the data value obtained from the simulation. The optimal trajectory was obtained from the above optimization results, and the motion was analyzed in ADAMS to derive kinematic parameter curves to test whether

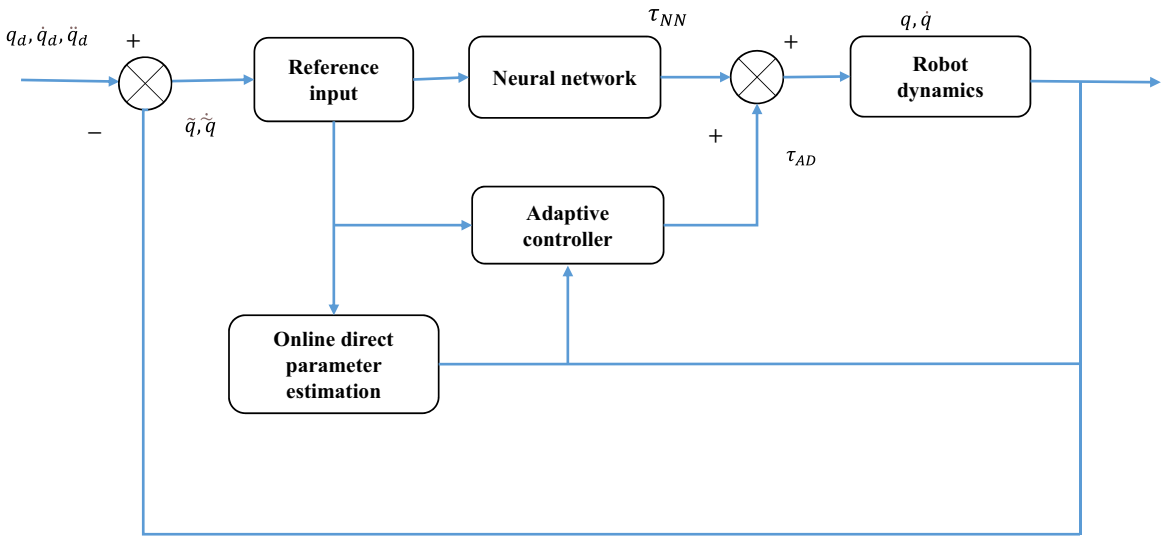

Figure 3: Schematic diagram of neural network adaptive control system.

they satisfy the constraints. If the constraint is satisfied, the result is optimal; if not, the optimization result is reselected and simulated again until the constraint is satisfied. Among them, the speed of the joint key starting point of the PUMA560 industrial robot is set to 0, the speed of the end point is also set to 0, and the acceleration is set to 1, after filling in the initial data, the time-optimal trajectory planning simulation of the industrial robot is carried out. The specific ADAM algorithm simulation process is as follows:

(1) Set all joint kinematics constraints of the industrial robot, and write the source code of the adaptive function, and optimize the writing statement of the source code to achieve time optimization.

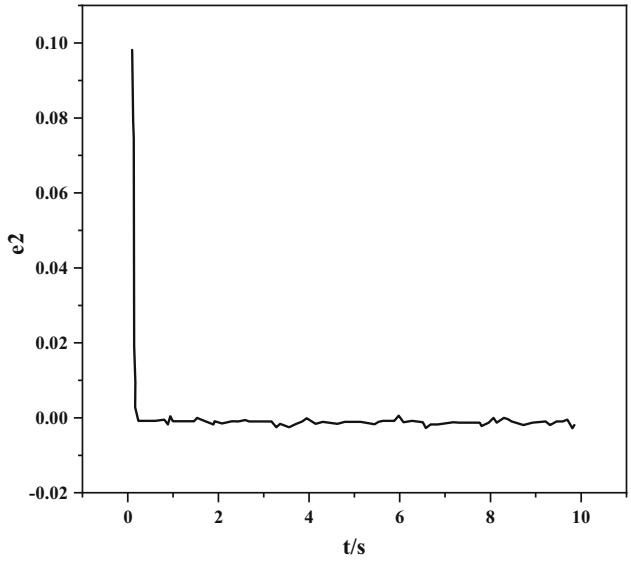

Figure 4: Tracking error.

(2) Through the ADAM algorithm provided in Matlab, the upper and lower limits of the specified individuals are set in the defined population function, and other relevant parameters and variables are set by default to expand the calculation of the time optimal trajectory.

(3) Optimize the trajectory calculation process to obtain the final time optimization simulation results of each trajectory.

Through the calculation of the optimization results and the simulation curve data value, the simulation results of the time optimal trajectory planning of industrial robots based on the ADAM algorithm can be calculated, as shown in Table 1: From the simulation results, it can be seen that

Table 1: Matlab simulation results of this project

Joint point	h1	h2	h3	h4	h5	h6	h7	Total time
1	0.24	1.643	1.634	1.04	1.796	1.264	1.786	9.42
2	0.475	1.421	2.866	1.105	1.202	2.104	1.196	10.37
3	0.943	1.495	1.663	1.157	1.358	1.986	1.973	10.57

Table 2: Matlab simulation results after optimization

Joint point	h1	h2	h3	h4	h5	h6	h7	Total time
1	0.24	1.667	1.636	1.10	1.798	1.287	1.189	8.9
2	0.475	1.421	2.838	1.105	1.202	2.107	1.108	10.25
3	0.933	1.475	1.653	1.145	1.838	1.855	1.353	10.25

the track running time difference corresponding to the three joint points is small, in order to make the position of each joint point basically synchronized in time; it is necessary to select the optimal node position for the time track, so as to ensure the synchronization between key nodes [19]. Therefore, adjust the joint node position so that the total time and time of the final simulation results are basically synchronized, both 10.35 s, as shown in Table 2.

At this time, the time required for the three joint nodes to complete the time-optimal trajectory is less than the trajectory planning time optimized by the ADAM algorithm in terms of total time and shows a better synchronization result. The simulation results prove that the ADAM algorithm realizes the trajectory optimization of industrial robots in terms of time planning, which can make each joint point of the industrial robot basically synchronize in the time trajectory [20].

4 Conclusion

In the process of mechanized production and application of industrial robots, time efficiency, and work quality are important indicators to evaluate their own performance. Taking the time trajectory planning of the PUMA560 industrial robot as the research object, the author uses the ADAM algorithm to optimize the running time of each joint node of the industrial robot and takes the time optimization as the goal on the basis of setting constraints, expanding Matlab trajectory planning coding and simulation. Improving ADAM improves the operation efficiency, local search ability, and real-time performance of the algorithm. The improved ADAM optimization objective function is used to find the optimal result, which obtains the optimal trajectory as the motion trajectory of the robot simulation model. The kinematic parameters and dynamic parameters of each joint of the robot meet the constraints in ADAMS, which verifies the rationality of the optimization trajectory. The simulation results show that the orbital time difference corresponding to the three joint points is small. To substantially synchronize the temporal locations of each joint point, selecting the optimal node locations for the temporal track is required to ensure synchronization between key nodes. The position of the joint nodes was adjusted so that the total time and time of the final simulation results were basically synchronized, both at 10.35 s. PUMA 560 Industrial robots can effectively complete industrial manufacturing tasks under a specific ADAM algorithm. The content studied in this article does not involve mechanical dynamics, but the current motor cannot meet the demand, so the motor with torque sensor drives the robot joint and realizes closed-loop control, which can achieve better control effect and effectively reduce the vibration of the robot movement. The content of the study of trajectory planning did not consider the dynamics of continuous; in some high-end robots, industrial applications are needed to consider it, achieve higher precision control requirements, and can further reduce the potential impact of the robot arm structure, so the next work is to consider the degree of a continuous excellent solution.

Funding information: This work was sponsored in part by the Jiangsu Qing Lan Project.

Author contributions: The author made significant individual contributions to this manuscript. Yicen Xu: writing and performing surgeries; data analysis and performing surgeries; article review and intellectual concept of the article.

Conflict of interest: The author declares that they have no competing interests.

Ethical approval: The conducted research is not related to either human or animals use.

Informed consent: Informed consent was obtained from all individuals included in this study.

Data availability statement: The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

References

[1] D. Malyuta, Y. Yu, P.Elango, and B. Açıkmeşe, “Advances in trajectory optimization for space vehicle control,” *Annu. Rev. Control.*, vol. 52, pp. 282–315, 2021.

[2] Z. Wang, X. Zhou, C. Xu, and F. Gao, “Geometrically constrained trajectory optimization for multicopters,” *IEEE Trans. Robot.*, vol. 38, no. 5, pp. 3259–3278, 2022.

[3] Y. Cao, Z. Zhang, F. Cheng, and S. Su, “Trajectory optimization for high-speed trains via a mixed integer linear programming approach,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17666–17676, 2022.

[4] M. Tajalli and A. Hajbabaie, “Traffic signal timing and trajectory optimization in a mixed autonomy traffic stream,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6525–6538, 2021.

[5] W. Lu, Y. Ding, Y. Gao, S. Hu, Y. Wu, N. Zhao, et al., “Resource and trajectory optimization for secure communications in dual unmanned aerial vehicle mobile edge computing systems,” *IEEE Trans. Ind. Inform.*, vol. 18, no. 4, pp. 2704–2713, 2021.

[6] B. Li, Q. Li, Y. Zeng, Y. Rong, and R. Zhang, "3D trajectory optimization for energy-efficient UAV communication: A control design perspective," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 6, pp. 4579–4593, 2021.

[7] A. T. Khan, S. Li, and X. Zhou, "Trajectory optimization of 5-link biped robot using beetle antennae search," *IEEE Trans. Circuits Syst. II: Express Briefs*, vol. 68, no. 10, pp. 3276–3280, 2021.

[8] L. Zhang, A. Celik, S. Dang, and B. Shihada, "Energy-efficient trajectory optimization for UAV-assisted IoT networks," *IEEE Trans. Mob. Comput.*, vol. 21, no. 12, pp. 4323–4337, 2021.

[9] A. Azarfar, B. Azarfar, and M. Vahedi, "Self-tuning fuzzy task space controller for puma 560 robot," *J. Electr. Eng. Technol.*, vol. 16, pp. 579–589, 2021.

[10] G. Singh and V. K. Banga, "Trajectory planning and obstacle avoidance of puma 560 by using soft computing techniques," *Harbin Gongye Daxue Xuebao/J. Harbin Inst. Technol.*, vol. 53, no. 9, pp. 125–137, 2021.

[11] H. Rahali, S. Zeghlache, and L. Benyettou, "Fault tolerant control of robot manipulators based on adaptive fuzzy type-2 backstepping in attendance of payload variation," *Int. J. Intell. Eng. Syst.*, vol. 14, no. 4, pp. 312–325, 2021.

[12] C. Wang, D. Liu, Q. Sun, and T. Wang, "Analysis of open architecture 6R robot forward and inverse kinematics adaptive to structural variations," *Math. Probl. Eng.*, vol. 2021, pp. 1–11, 2021.

[13] A. A. Raji, O. S. Asaolu, and T. T. Akano, "Joint space robot arm trajectory planning using septic function," *ABUAD J. Eng. Res. Dev.*, vol. 5, pp. 110–123, 2022.

[14] A. Rout, D. BBVL, B. B. Biswal, and G. B. Mahanta, "Optimal trajectory planning of industrial robot for improving positional accuracy," *Ind. Rob: Int. J. Rob. Res. Appl*, vol. 48, no. 1, pp. 71–83, 2021.

[15] P. Mallahi Kolahi and M. Mosayebi, "Optimal trajectory planning for an industrial mobile robot using optimal control theory," *J. Mod. Process. Manuf. Prod.*, vol. 10, no. 3, pp. 25–34, 2021.

[16] J. Zhao, X. Zhu, and T. Song, "Serial manipulator time-jerk optimal trajectory planning based on hybrid iwoa-pso algorithm," *IEEE Access*, vol. 10, pp. 6592–6604, 2022.

[17] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, vol. 9, pp. 119310–119342, 2021.

[18] L. Zhang, Y. Wang, X. Zhao, P. Zhao, and L. He, "Time-optimal trajectory planning of serial manipulator based on adaptive cuckoo search algorithm," *J. Mech. Sci. Technol.*, vol. 35, pp. 3171–3181, 2021.

[19] T. Zhang, M. Zhang, and Y. Zou, "Time-optimal and smooth trajectory planning for robot manipulators," *Int. J. Control Autom. Syst.*, vol. 19, no. 1, pp. 521–531, 2021.

[20] H. Touzani, H. Hadj-Abdelkader, N. Séguy, and S. Bouchafa, "Multi-robot task sequencing & automatic path planning for cycle time optimization: Application for car production line," *IEEE Rob. Autom. Lett.*, vol. 6, no. 2, pp. 1335–1342, 2021.