 CMES, vol.118, no.3, pp.583-598, 2019

Trajectory Planning of High Precision Collaborative Robots

Tuanjie Li[1,*], Yan Zhang[1] and Jiaxing Zhou[1]

Abstract: In order to satisfy the high efficiency and high precision of collaborative robots, this work presents a novel trajectory planning method. First, in Cartesian space, a novel velocity look-ahead control algorithm and a cubic polynomial are combined to construct the end-effector trajectory of robots. Then, the joint trajectories can be obtained through the inverse kinematics. In order to improve the smoothness and stability in joint space, the joint trajectories are further adjusted based on the velocity look-ahead control algorithm and quintic B-spline. Finally, the proposed trajectory planning method is tested on a 4-DOF serial collaborative robot. The experimental results indicate that the collaborative robot achieves the high efficiency and high precision, which validates the effectiveness of the proposed method.

Keywords: collaborative robot, high efficiency, high precision, velocity look-ahead control, trajectory planning, quintic B-spline.

1 Introduction

Collaborative robots are widely used in modern industrial manufacturing fields. With the development of industry, the demand for efficiency and precision of collaborative robots become higher and higher. The trajectory planning is one of the most important ways to improve the motion performance of robots. The trajectory planning is generally carried out in Cartesian space and/or in joint space.

In Cartesian space, aiming to create a smooth trajectory under kinematic constraints, the look-ahead interpolation techniques and nonlinear optimization algorithms [Pham and Nakamura (2015); Zhu, Hou, Wang et al. (2015)] are adopted in general. The former is usually used in computer numerical control (CNC) and mobile robots. Based on the smooth interpolation functions, such as jerk function, B-spline, non-uniform rational B-spline (NURBS) and S curve, the velocity look-ahead control algorithms have been adopted to obtain the smooth trajectory of robots [Wang, Yang, Gai R et al. (2015); Lin, Lee, Lee et al. (2016); Tsai, Nien and Yau (2008); Zhang, Yu, Hu et al. (2009); Huang (2018)]. Moreover, the look-ahead interpolation algorithm based on the micro-line was presented to construct the trajectory for CNC [Zhang, Sun, Gao et al. (2011); Shi and Ye (2011)]. These methods have effectively improved the kinematic performance of robot by estimating the curvature of trajectory and calculating the suitable velocity and acceleration. As for the industrial robots, the nonlinear optimization algorithms are

[1] School of Mechano-Electronic Engineering, Xidian University, Xi'an, 710071, China.

[*] Corresponding Author: Tuanjie Li. Email: tjli888@126.com.

adopted. The B-spline was used to interpolate trajectory, and the optimization algorithms, including the sequential quadratic programming (SQP) and genetic algorithm (GA), were adopted to optimize the trajectory [Li, Huang and Chetwynd (2018); Su, Cheng, Wang et al. (2018); Saravanan and Ramabalan (2008); Chettibi, Lehtihet, Haddad et al. (2004); Wang and Horng (1990)]. The trajectory planning in Cartesian space is easy to observe the motion trail and attitude of the end effector of the robot. However, because of the nonlinear relationship between the end-effector trajectory in Cartesian space and joint motion in joint space, and the kinematic singularity of robots, the final joint trajectory obtained by inverse kinematic solver cannot be guaranteed to meet the kinematic constraints. So, the efficiency and precision of collaborative robots may deteriorate obviously.

In joint space, the trajectories of joints can be constructed by using the polynomial, jerk function, Bezier curve or B-spline to fit the discrete points of each joint gained through inverse kinematic solver [Valente, Baraldo and Carpanzano (2017); Wu, Zhu and Liu (2009); Lin, Chang and Luh (1983)]. In order to meet the kinematic constraints and optimize the efficiency and accuracy of robots, the time [Kim and Kim (2011); Koblick, Xu, Foge et al. (2016)], jerk and energy optimal [Hashemiana, Hosseinib and Nabavib (2017)] techniques have been investigated in the past decades. For industrial robots, Huang et al. [Huang, Hu, Wu et al. (2018)] applied the quintic B-spline and non-dominated sorting GA (NSGA-II) to achieve the time-jerk synthetic optimal planning trajectory. Liu et al. [Liu, Lai and Wu (2013)] utilized B-spline in the joint trajectories interpolation to optimize the trajectory with SQP. For a 6-DOF PUMA560 robot, Zhang et al. [Zhang, Meng, Feng et al. (2018)] constructed time optimal trajectories by using cubic spline and NSGA-II. The CVX, which is convex optimization toolbox in MATLAB, was adopted to obtain time optimal trajectories [Mulik (2015)]. However, due to the robot system is nonlinear, the planning trajectory in joint space cannot guarantee that the final motion of the end-effector satisfies the kinematic constraints in Cartesian space. And the end-effector is easy to vibrate at large curvature of trajectory. Further, because the complex optimization algorithms are applied, the response time of the system is increased obviously.

This work presents a trajectory planning method based on the velocity look-ahead control algorithm. First, under the kinematic constraints of end-effector, the end-effector trajectory is constructed by combining a modified velocity look-ahead control algorithm and cubic polynomial. Then, the joint trajectories are derived through inverse kinematics of robot. Last, in order to improve the smoothness and stability in joint space, the modified velocity look-ahead control algorithm and quintic B-spline are combined to further adjust the joint trajectories. The rest of this paper is organized as follows. Section 2 describes a novel trajectory planning method in Cartesian space. And a novel trajectory planning method in joint space is proposed in Section 3. In Section 4, the experimental results are shown to validate the proposed method. Section 5 gives the main conclusions.

2 Trajectory planning method in cartesian space

A paper for publication can be subdivided into multiple sections: a title, full names of all the authors and their affiliations, a concise abstract, a list of keywords, main text (including figures, equations, and tables)), acknowledgements, references, and appendix. Running title is optional.

2.1 Velocity calculation of break points

The curve path of end-effector consists of discrete break points, as shown in Fig. 1. When the manipulator passes through these break points, it will generate large acceleration. In order to improve the motion performance of manipulator, we propose the modified velocity look-ahead control algorithm to restrict the velocity and acceleration at break points.

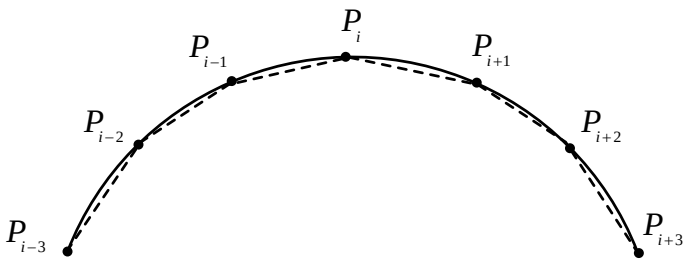


Figure 1: The discrete break points of curve path

The velocity at a break point must satisfy the following constraints.

$$\begin{cases} |v_i| \le v_m \\ \left|\dfrac{\Delta v_i}{\Delta t}\right| \le a_m \end{cases}, v_0 = 0,\ \ \Delta l_i = min\{l_{step},\ l_i\},\ \ \Delta t_i = \left|\frac{\Delta l_i}{v_i}\right|,\ \ i = 1,2,\ldots,n \tag{1}$$

where, v_i is the velocity of the i th break point, l_{step} is the step length of discrete path, l_i is the linear displacement from the $(i-1)$th to the i th break points, Δl_i and Δt_i are the distance and time for velocity direction changes from the vector $\boldsymbol{P}_{i-1}\boldsymbol{P}_i$ to $\boldsymbol{P}_i\boldsymbol{P}_{i+1}$, respectively. v_m and a_m are the maximum velocity and acceleration of end-effector, respectively. n is the number of break points, Δv_i is the velocity change between the $(i-1)$th and the i th break points, and

$$\Delta v_i = \sqrt{\Delta v_{ix}^2 + \Delta v_{iy}^2 + \Delta v_{iz}^2},\quad i = 1,2,\ldots,n \tag{2}$$

in which,

$$\begin{cases} \Delta v_{ix} = |v_i| \cdot (cos\theta_{ix} - cos\theta_{(i-1)x}) \\ \Delta v_{iy} = |v_i| \cdot (cos\theta_{iy} - cos\theta_{(i-1)y}) \\ \Delta v_{iz} = |v_i| \cdot (cos\theta_{iz} - cos\theta_{(i-1)z}) \end{cases} \tag{3}$$

where, θ_{ix}, θ_{iy} and θ_{iz} are the angles between the vector $\boldsymbol{P}_{i-1}\boldsymbol{P}_i$ and X-axis, Y-axis and Z-axis, respectively.

Substituting Eqs. (2) and (3) to Eq. (1), we can get the maximum permissible velocity of break points as follows.

$$v_{im} = sgn(\Delta l_i) \cdot min\left\{ v_m, \quad \sqrt{\frac{a_m \Delta l_i}{K}} \right\}, \quad i = 1, 2, \ldots, n \tag{4}$$

where,

$$sgn(\Delta l_i) = \begin{cases} 1, & \Delta l_i \geq 0 \\ -1, & \Delta l_i < 0 \end{cases} \tag{5}$$

$$K = \sqrt{(cos\theta_{ix} - cos\theta_{(i-1)x})^2 + (cos\theta_{iy} - cos\theta_{(i-1)y})^2 + (cos\theta_{iz} - cos\theta_{(i-1)z})^2} \tag{6}$$

For the high efficiency of robot motion, the velocity of break point takes the maximum permissible value.

$$v_i = v_{im} \tag{7}$$

Further, the displacement constraints of adjacent break points must be taken into account. If the velocities of break point *i*-1 (v_{i-1}) and *i* (v_i) are not the same, the v_{i-1} must change to v_i under the constraint conditions that the displacement and acceleration of this process are less than l_i and a_m respectively, so

$$l_{imin} = \frac{\left| v_i^2 - v_{i-1}^2 \right|}{2a_m} \leq l_i, \quad i = 1, 2, \ldots, n \tag{8}$$

where, l_{imin} is the minimum displacement required for v_{i-1} changing into v_i. According to Eq. (8), when $l_{imin} > l_i$ and $v_i \geq v_{i-1}$, v_i needs to be reduced as

$$v_i = \sqrt{2a_m l_i + v_{i-1}^2} \tag{9}$$

when $l_{imin} > l_i$ and $v_i < v_{i-1}$, v_{i-1} needs to be reduced as

$$v_{i-1} = \sqrt{2a_m l_i + v_i^2} \tag{10}$$

The traditional velocity look-ahead control algorithm only adjusts the velocity of break points once and cannot guarantee that all the velocities meet the displacement constraints shown in Eq. (8). For this, a novel velocity look-ahead control method is proposed to adjust the velocities of all break points and the adjustment flow chart is shown in Fig. 2.

2.2 Velocity calculation of linear points

The linear points, which are between the adjacent break points, are obtained by dispersing straight lines. To ensure the smoothness and stability of trajectory, we utilize the acceleration/deceleration (ACC/DEC) hybrid algorithm, based on the cubic polynomial, to calculate the velocity of these linear points. The velocity equation of ACC/DEC is described as

$$V(t) = \frac{V_1 - V_0}{(T_1 - T_0)^3}\left[-2t^3 + 3(T_0 + T_1) \cdot t^2 - 6T_0 T_1 \cdot t + T_0^3 (3T_1 - T_0) \right] + V_0, \quad t \in [T_0, T_1] \tag{11}$$

where, V_0, V_1, T_0 and T_1 are the initial velocity, terminal velocity, initial time and terminal time of ACC/DEC period, respectively.

When the velocities of start and end points are V_s and V_e, respectively, the velocity of uniform phase is V_a, the maximum velocity is V_m, there exist seven profiles of velocity curves with the ACC/DEC hybrid algorithm. The velocity curves are shown in Fig. 3.

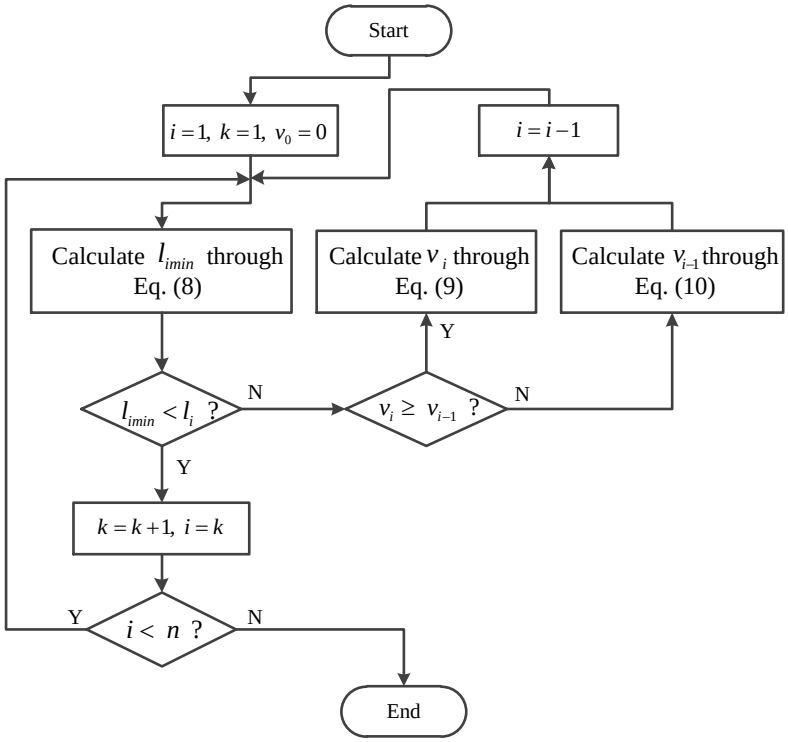


Figure 2: Flow chart of velocity adjustment

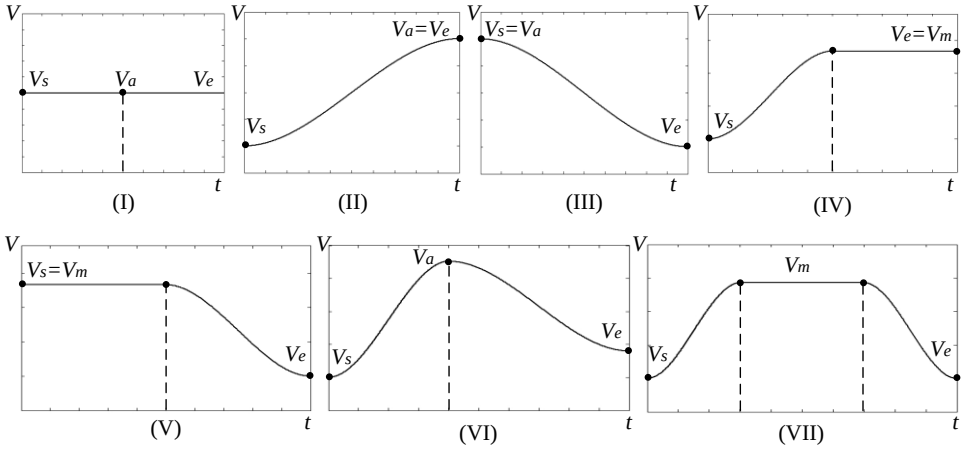


Figure 3: Seven profiles of velocity curves

In order to meet the kinematic constraint $V^{'}(t) \le a_m$ and reach the high efficiency of motion, the average acceleration of ACC/DEC is defined as

$$A_a = \frac{V_1 - V_0}{T_1 - T_0} = \frac{2}{3} a_m \tag{12}$$

The boundary conditions corresponding to the seven profiles of velocity curves are shown in Tab. 1.

Table 1: Boundary conditions corresponding to the seven profiles of velocity curves

Cases	I	II	III	IV	V	VI	VII
Boundary conditions	$L \le L_b$ $V_s = V_e$	$L \le L_b$ $V_e = V_a$	$L \le L_b$ $V_s = V_a$	$L > L_b$ $V_e = V_m$	$L > L_b$ $V_s = V_m$	$L \le L_b$ $V_e < V_m$ $V_s < V_m$	$L > L_b$ $V_e < V_m$ $V_s < V_m$

in which,

$$\begin{cases} L_b = \dfrac{1}{2a_m}(2V_m^2 - V_s^2 - V_e^2) \\ V_a = \sqrt{\dfrac{1}{2}(2a_m L + V_s^2 + V_e^2)} \end{cases} \tag{13}$$

where, L is the total displacement along a straight line, L_b is a displacement boundary condition, when $L > L_b$, there has a uniform phase, when $L \le L_b$, the uniform phase does not exist.

3 Trajectory planning method in joint space

3.1 Time calculation of discrete points

On the basis of trajectory planning in Cartesian space, we can get the angles and angular velocities by applying inverse kinematic transformation. In joint space, the joint trajectories also need to meet the constraints shown in Eq. (8). Therefore, the angular velocities must be adjusted, and the adjustment process is as follows.

Step 1: detecting the break points in joint space and changing the angular velocities of break points to zero.

Step 2: using the points whose velocities are zero as boundaries to separate the entire trajectory to independent parts.

Step 3: defining ω_m and α_m as the maximum angular velocity and angular acceleration, respectively, and taking all discrete points as break points. Then, the angular velocities of each independent part can be adjusted through the method shown in Fig. 2.

The proposed ACC/DEC hybrid algorithm is applied to fit the adjacent points in joint space. According to the seven profiles of velocity curves shown in Fig. 3, the motion time between two adjacent points can be calculated by the equations described in Tab. 2.

Table 2: The motion time of adjacent points

Cases	Time
I	$\Delta T = \frac{L}{V_a}$
II	$\Delta T = \frac{V_e - V_s}{A_a}$
III	$\Delta T = \frac{V_s - V_e}{A_a}$
IV	$\Delta T = \frac{2A_aL + (V_m - V_s)^2}{2A_aV_m}$
V	$\Delta T = \frac{2A_aL + (V_m - V_e)^2}{2A_aV_m}$
VI	$\Delta T = \frac{2V_a - V_s - V_e}{A_a}$
VII	$\Delta T = \frac{2A_aL + (V_m - V_s)^2 + (V_m - V_e)^2}{2A_aV_m}$

where, $V_m = \omega_m$, $A_a = \frac{2}{3}\alpha_m$.

The time of discrete points is

$$t_i = t_{i-1} + max\left\{\Delta T_{1,i},\ \Delta T_{1,i},\ \ldots,\ \Delta T_{p,i}\right\}, \quad i = 1,2,\ldots,n \tag{14}$$

where, p is the number of joints, $\Delta T_{j,i}$ $(j = 1,2,\ldots,p)$ is the time that the j th joint moves from point $i-1$ to point i.

3.2 Quintic B-spline fitting

Through the above calculation, we have gained the discrete angles and time of every joint. In order to get the smooth trajectory of each joint, the multi-degree B-splines which have good smoothness and local support is adopted to fit these discrete joint points. Based on

the Cox-de Boor recursion formula, we can get the basis function of the *k*th-degree B-spline as follows.

$$N_{i,0}(u)=\begin{cases}1, & u\in[u_i,\ u_{i+1}]\\ 0, & other\end{cases},\quad i=0,1,\cdots,n+k \tag{15}$$

$$N_{i,k}(u)=\frac{u-u_i}{u_{i+k}-u_i}N_{i,k-1}(u)+\frac{u_{i+k+1}-u}{u_{i+k+1}-u_{i+1}}N_{i+k,k-1}(u)\quad ,define:\frac{0}{0}=1 \tag{16}$$

where, u_i is the knot and normalized to the interval $[0,\ 1]$. Using the accumulative chord length method, we normalize the time variable t_i to gain the knot variable u_i.

$$u_i=\begin{cases}0, & i=0,1,\cdots,k\\ u_{i-1}+\dfrac{t_{i-k}-t_{i-k-1}}{t_{n-k+1}-t_0}, & i=k+1,k+2,\cdots,n\\ 1, & i=n+1,\cdots,n+k+1\end{cases} \tag{17}$$

where, the knots have multiplicity $k+1$ at each end so that the parameters at the endpoints of trajectory can be controlled.

To interpolate $n+1$ discrete points $\boldsymbol{p}_i$ by the *k*th B-spline, the $n+1$ equations of trajectory can be written as follows.

$$\boldsymbol{P}(u_i)=\sum_{j=i-k}^{i}d_jN_{j,k}(u_i)=\boldsymbol{p}_{i-k},\ \ u_i\in\left[u_i,u_{i+1}\right],\ \ i=k,k+1,\cdots,n+k \tag{18}$$

where, d_j $(j=0,1,\ldots,n+k-1)$ is the control points that need to be solved, the vector $\boldsymbol{p}_i$ is

$$\boldsymbol{p}_i=\left(\theta_i,\ t_i\right),\quad i=0,1,2,\ldots,n \tag{19}$$

where, θ_i and t_i is the angle and time of point i, respectively.

According to Eqs. (18) and (19), we only have $n+1$ equations to solve $n+k$ control points, the extra $k-1$ equations must be provided by kinematic constraint conditions, including velocity and acceleration constraints at start and end points. The *r*th order derivatives for *k*th-degree B-spline can be expressed as

$$p^{(r)}(u)=\frac{\partial^r}{\partial u^r}\sum_{j=0}^{n}d_jN_{j,k}(u)=\sum_{j=i-k+r}^{i}d_j^{(r)}N_{j,k-r}(u),\quad u\in\left[u_i,\ u_{i+1}\right] \tag{20}$$

here,

$$d_j^{(l)}=\begin{cases}d_j, & l=0\\ (k-l+1)\dfrac{d_j^{(l-1)}-d_{j-1}^{(l-1)}}{u_{j+k-l+1}-u_j}, & l=1,2,\cdots,r\end{cases} \tag{21}$$

When the angular velocities and angular accelerations at starting and ending positions (t_0 and t_n) are $[\omega(t_0),\ \omega(t_n)]$ and $[\alpha(t_0),\ \alpha(t_n)]$, respectively, we can obtain the kinematic constraint conditions through Eqs. (20) and (21).

$$\begin{cases} \omega(t_0) = P^{(1)}(u_k) = \sum_{j=k-k+1}^{k} d_j^{(1)} N_{j,k-1}(u_k) = d_1^{(1)} \\ \omega(t_n) = P^{(1)}(u_{n+1}) = \sum_{j=n-k+1}^{n} d_j^{(1)} N_{j,k-1}(u_{n+1}) = d_n^{(1)} \\ \alpha(t_0) = P^{(2)}(u_k) = \sum_{j=k-k+2}^{k} d_j^{(2)} N_{j,k-1}(u_k) = d_2^{(2)} \\ \alpha(t_n) = P^{(2)}(u_{n+1}) = \sum_{j=n-k+2}^{n} d_j^{(2)} N_{j,k-1}(u_{n+1}) = d_n^{(2)} \end{cases} \tag{22}$$

According to Eqs. (18), (19) and (22), we can solve all control points of quintic B-spline ($k=5$). Then, substituting the control points to Eq. (18), we can obtain the continuous joint trajectories. Fig. 4 shows the whole process of trajectory planning.

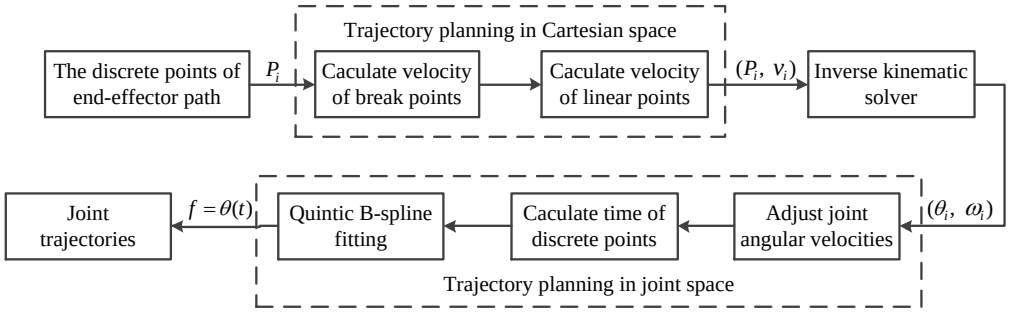


Figure 4: The whole process of trajectory planning

4 Experimental results

To verify the effectiveness of the proposed trajectory planning method, the experiments are carried out on a 4-DOF serial collaborative robot, as shown in Fig. 5. Fig. 5(a) shows the simulated model of robot including four joints with its standard D-H link coordinate shown in Fig. 5(b). The D-H parameters of the robot are described in Tabs. 3, and the initial value of θ_i $(i=1,2,3,4)$ is zero. The kinematic constraints in Cartesian space (the maximum velocity and acceleration of end-effector) are given as in Tab. 4. Tab. 5 shows the working range and kinematic constraints of each joint (the maximum angular velocity and angular acceleration of each joint).

The task of experiments is to track a closed path in the XY plane (Z=0), as shown in Fig. 6. The closed path is made up of 686 discrete points. The end effector starts from point S and moves along the closed path anticlockwise. The velocity and acceleration at the

initial and the ending moments are configured as zero. We adopt the proposed velocity look-ahead control algorithm and ACC/DEC algorithm to gain the velocities of all discrete points in Cartesian space. The displacement and velocity of discrete points are described in Fig. 7. From Fig. 7, we can find that the velocity of end effector increases from zero quickly at the beginning because the curvature of line $\overline{SP_b}$ shown in Fig. 6 is zeros. But after the point P_b is an arc and the curvature rise suddenly. In order to guarantee the motion of end-effector meets the kinematic constraints when passing the arc with large curvature, the velocity drops steadily in advance when approaching the point P_b. Moreover, the acceleration period reduces with the linear distance decreases between adjacent break points, like the line $\overline{P_cP_d}$ shown in Fig. 6.

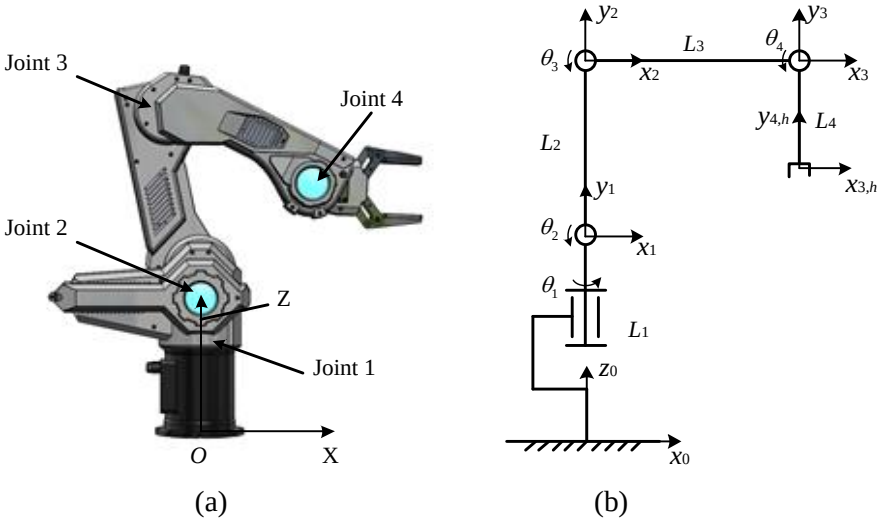


Figure 5: The 4-DOF serial collaborative robot. (a) Simulation model. (b) D-H link coordinate system

Table 3: Standard D-H parameters

Parameters	d_i(mm)	a_i(mm)	α_i(rad)	θ_i(rad)
1	255	0	0.5π	θ_1
2	0	257	0	$\theta_2+0.5\pi$
3	0	250	0	$\theta_3-0.5\pi$
4	0	140	0	$\theta_4-0.5\pi$

Table 4: Kinematic constraints for end-effector

Constraints	Sup $\lvert v_m \rvert$ (mm / s)	Sup $\lvert a_m \rvert$ (mm/ s^2)
Values	200	100

Table 5: Kinematic constraints for each joint

Joint number	Working range (rad)	Sup $\lvert \omega_m \rvert$ (rad / s)	Sup $\lvert \alpha_m \rvert$ (rad / s^2)
1	$[-0.75\pi,\ 0.75\pi]$	2.0	1.0
2	$[-0.75\pi,\ 0.75\pi]$	2.5	1.2
3	$[-0.25\pi,\ 1.25\pi]$	3.0	1.5
4	$[-0.25\pi,\ 1.25\pi]$	3.0	1.8

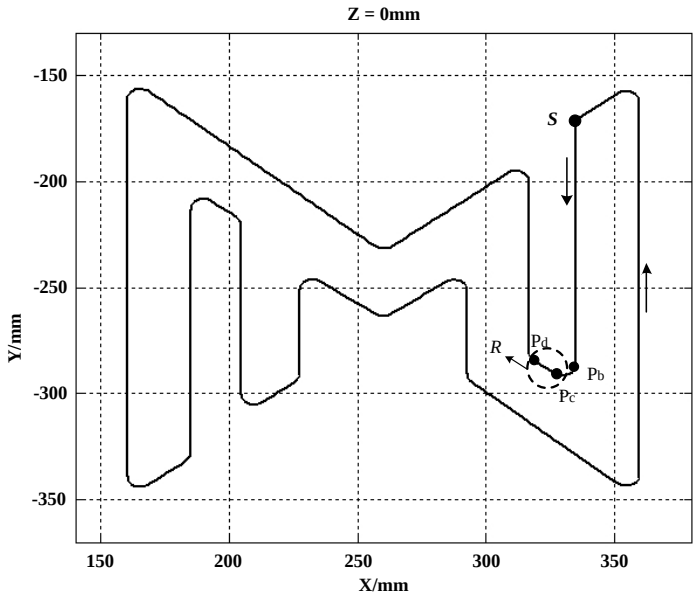


Figure 6: The closed path in the XY plane

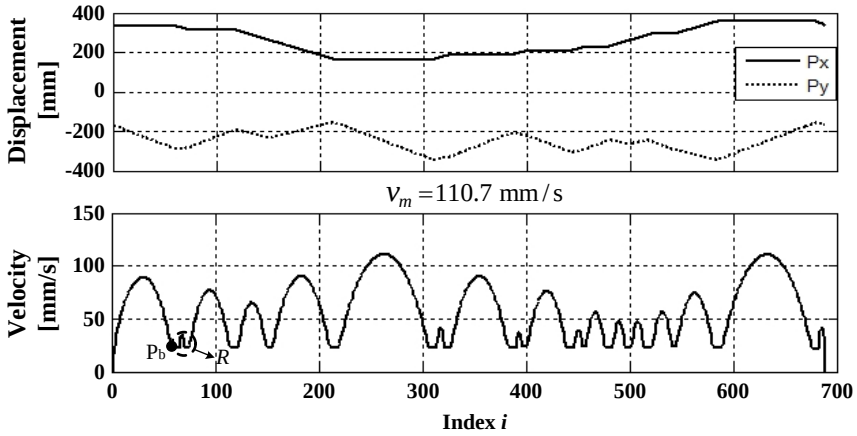
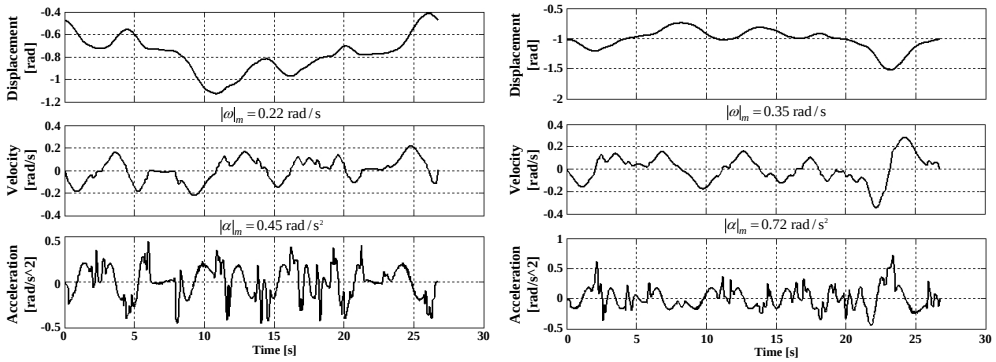
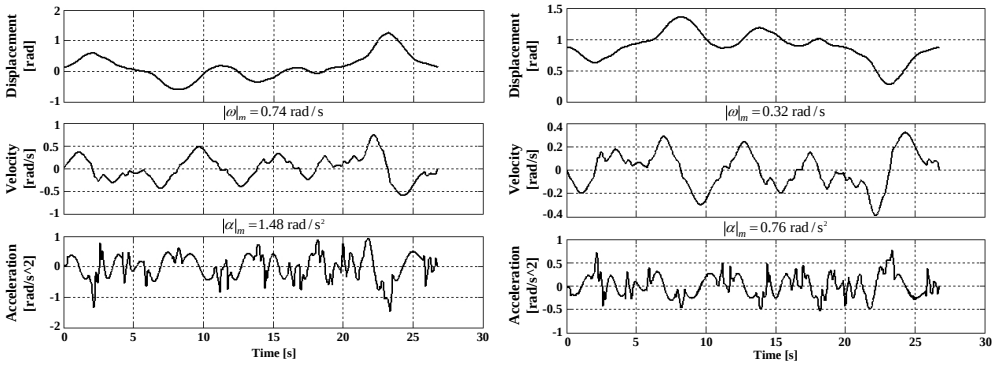


Figure 7: The displacement and velocity of discrete points

After obtaining the position and velocity of end effector, the joint angle and angular velocity can be gained by the inverse kinematic solver which is constructed through the standard D-H method based on the D-H parameters shown in Tab. 3. Then, the trajectory planning method in joint space is adopted to construct the smooth joint trajectories which are shown in Fig. 8. From Fig. 8, we can see the angular velocity and angular acceleration of each joint satisfy the kinematic constraints illustrated in Tab. 4, and the execution time of robot is 26.8 s. Moreover, because the joint trajectories are the quintic B-spline, the movement of each joint is continuous and smooth which is useful to reduce the flexible impact caused by the joint actuator and improve the fluency performance.



(a) (b)



(c) (d)

Figure 8: The trajectory of each joint. (a) Joint 1, (b) Joint 2, (c) Joint 3, (d) Joint 4

To display the experimental result that the manipulator tracks the closed path shown in Fig. 6, we use a black pen as the end effector and the pen length is 60 mm. The experimental result is shown in Fig. 9, which indicates that the 4-DOF collaborative robot achieves the expected requirements successfully. The trajectory of end effector is shown in Fig. 10, and the equations of velocity and acceleration are as follows.

$$\begin{cases} V = \sqrt{V_x^2 + V_y^2} \\ A = \sqrt{A_x^2 + A_y^2} \end{cases} \tag{23}$$

where, V_x , V_y , A_x and A_y are the projections of velocity and acceleration on the X-axis and Y-axis, respectively.

From Fig. 10, we can find that the manipulator trajectory meets the kinematic constraints shown in Tab. 4. Because of the nonlinear relationship between the end-effector and joint and the kinematic singularity of joints, the final trajectory of end-effector is properly adjusted at some of break points, like the stage *R*. Fig. 11 indicates the period that the acceleration is greater than specified value of boundary. From Fig. 11 we can see that the period that the acceleration is greater than $0.8a_m$ is more than 15 s, the period that the acceleration is greater than $0.5a_m$ is more than 20 s in the whole movement. The average acceleration is $66.8\,\mathrm{mm/s^2}$ which is greater than 60% of the maximum acceleration of end-effector obviously. Therefore, the robot achieves the high efficiency and high precision.

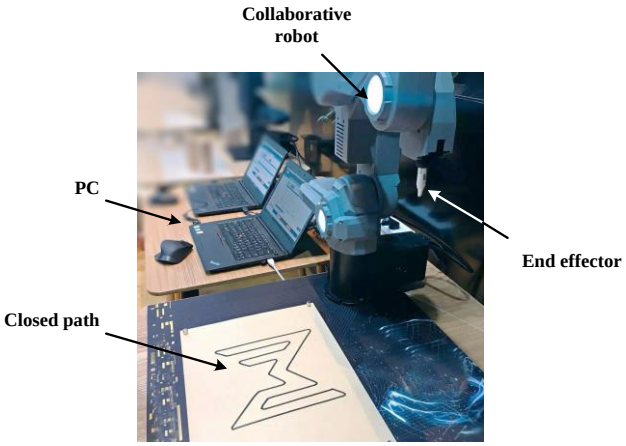


Figure 9: The experimental result

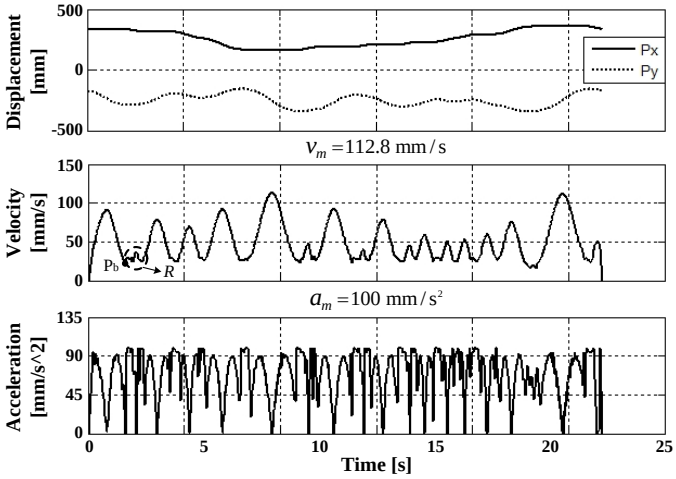


Figure 10: The trajectory of end effector

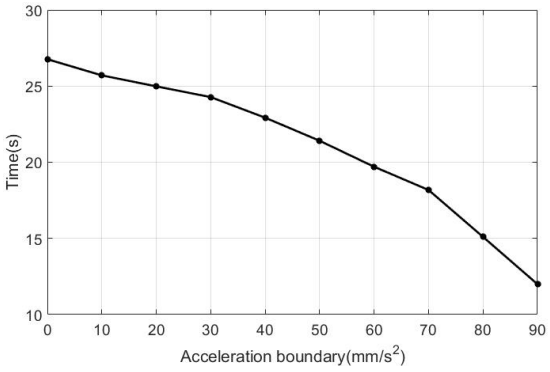


Figure 11: The period that the acceleration is greater than its boundary

5 Conclusions

This paper proposed a novel trajectory planning method for collaborative robots. Compared with the traditional methods based on the complex optimization algorithm, the proposed method has higher computational efficiency. Furthermore, through the cubic polynomial, the trajectory in Cartesian space are twice continuous differentiable. When using the quintic B-spline in joint space, the angular acceleration and jerk of joints are continuous. Because the kinematic constraints and minimum-time problem are all taken into account in both Cartesian space and joint space, the precision and efficiency of robot are improved effectively. The experimental results on a 4-DOF serial collaborative robot show that the robot maintains the high efficiency for about 75% of whole movement under kinematic constraints. Therefore, the effectiveness and practicability of the proposed method is verified.

References

Chettibi, T.; Lehtihet, H. E.; Haddad M.; Hanchi, S. (2004): Minimum cost trajectory planning for industrial robots. *European Journal of Mechanics*, vol. 23, pp. 703-715.

Hashemiana, A.; Hosseinib, S. F.; Nabavib, S. N. (2017): Kinematically smoothing trajectories by NURBS reparameterization an innovative approach. *Advanced Robotics*, vol. 31, no. 23-24, pp. 1296-1312.

Huang, H. (2018): An adjustable look-ahead acceleration/deceleration hybrid interpolation technique with variable maximum federate. *International Journal of Advanced Manufacturing Technology*, vol. 95, pp. 1521-1538.

Huang, J.; Hu, P.; Wu, K.; Zeng, M. (2018): Optimal time-jerk trajectory planning for industrial robots. *Mechanism & Machine Theory*, vol. 121, pp. 530-544.

Kim, K. B.; Kim, B. K. (2011): Minimum-time trajectory for three-wheeled omnidirectional mobile robots following a bounded-curvature path with a referenced heading profile. *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 800-808.

Koblick, D.; Xu, S.; Fogel, J.; Shankar, P. (2016): Low thrust minimum time orbit transfer nonlinear optimization using impulse discretization via the modified picard-chebyshev method. *Computer Modeling in Engineering & Sciences*, vol. 24, no. 1, pp. 1-27.

Li, Y.; Huang, T.; Chetwynd, D. G. (2018): An approach for smooth trajectory planning of high-speed pick-and-place parallel robots using quintic B-splines. *Mechanism & Machine Theory*, vol. 126, pp. 479-490.

Lin, C. S.; Chang, P. R.; Luh, J. Y. S. (1983): Formulation and optimization of cubic polynomial joint trajectories for industrial robots. *IEEE Transactions on Automatic Control*, vol. 28, no. 12, pp. 1066-1074.

Lin, M. T.; Lee, M. C.; Lee, J. C.; Lee, C. Y.; Jian, Z. W. (2016): A look-ahead interpolator with curve fitting algorithm for five-axis tool path. *IEEE International Conference on Advanced Intelligent Mechatronics*, pp. 12-15.

Liu, H.; Lai, X.; Wu, W. (2013): Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints. *Robotics & Computer Integrated Manufacturing*, vol. 29, pp. 309-317.

Mulik P. B. (2015): Optimal trajectory planning of industrial robot with evolutionary algorithm. *IEEE International Conference on Computation of Power, Energy Information and Communication*, pp. 256-263.

Pham, Q.; Nakamura, Y. (2015): A new trajectory deformation algorithm based on affine transformations. *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 1054-1063.

Saravanan, R.; Ramabalan, S. (2008): Evolutionary minimum cost trajectory planning for industrial robots. *Journal of Intelligent & Robotic Systems*, vol. 52, pp. 45-77.

Shi, C.; Ye, P. (2011): The look-ahead function-based interpolation algorithm for continuous micro-line trajectories. *International Journal of Advanced Manufacturing Technology*, vol. 54, pp. 649-668.

Su, T.; Cheng, L.; Wang, Y.; Liang, X.; Zheng, J. et al. (2018): Time-optimal trajectory planning for delta robot based on quintic pythagorean-hodograph curves. *IEEE Access*, vol. 6, pp. 28530-28539.

Tsai, M. S.; Nien, H. W.; Yau, H. T. (2008): Development of an integrated look-ahead dynamics-based NURBS interpolator for high precision machinery. *Computer-Aided Design*, vol. 40, pp. 554-566.

Valente, A.; Baraldo, S.; Carpanzano, E. (2017): Smooth trajectory generation for industrial robots performing high precision assembly processes. *CIRP Annals-Manufacturing Technology*, vol. 66, pp. 17-20.

Wang, C. H.; Horng, J. G. (1990): Constrained minimum-time path planning for robot manipulators via virtual knots of the cubic B-spline functions. *IEEE Transactions on Automatic Control*, vol. 35, no. 5, pp. 573-577.

Wang, Y.; Yang, D.; Gai, R.; Wang, S.; Sun, S. (2015): Design of trigonometric velocity scheduling algorithm based on pre-interpolation and look-ahead interpolation. *International Journal of Machine Tools & Manufacture*, vol. 96, pp. 94-105.

Wu, W.; Zhu, S.; Liu, S. (2009): Smooth joint trajectory planning for humanoid robots based on B-splines. *International Conference on Robotics and Biomimetics*, pp. 475-479.

Zhang, J.; Meng, Q.; Feng, X.; Shen, H. (2018): A 6-DOF robot-time optimal trajectory planning based on an improved genetic algorithm. *Robotics & Biomimetics*, vol. 5, pp. 3-4.

Zhang, L. X.; Sun, R. Y.; Gao, X. S.; Li, H. B. (2011): High speed interpolation for micro-line trajectory and adaptive real-time look-ahead scheme in CNC machining. *Science China Technological Sciences*, vol. 54, no. 6, pp. 1481-1495.

Zhang, X.; Yu, D.; Hu, Y.; Hong, H.; Sun, W. (2009): Development of a NURBS curve interpolator with look-ahead control and feedrate filtering for CNC system. *IEEE Conference on Industrial Electronics and Applications*. pp. 2755-2759.

Zhu, B. J.; Hou, Z. X.; Wang, X. Z.; Chen, Q. Y. (2015): Long endurance and long distance trajectory optimization for engineless UAV by dynamic soaring. *Computer Modeling in Engineering & Sciences*, vol. 106, no. 5, pp. 1799-1816.