The International Journal of Advanced Manufacturing Technology (2022) 119:4991–5009
https://doi.org/10.1007/s00170-021-08503-3

ORIGINAL ARTICLE

Design and implementation of multi-axis real-time synchronous look-ahead trajectory planning algorithm

Yanyang Liang[1] · Chaozhi Yao[1] · Wei Wu[1] · Li Wang[1] · Qiongyao Wang[1]

Received: 10 May 2021 / Accepted: 4 December 2021 / Published online: 14 January 2022

Abstract

Multi-joint industrial robots are widely used in many fields such as transportation, welding, and assembling. In order to meet the requirements of efficient and synchronous robot motion, a multi-axis real-time synchronous look-ahead trajectory planning algorithm is proposed based on dynamically given position and velocity sequences under the constraint of maximum velocity and acceleration of each joint axis. In addition, an efficient transition processing method is proposed to satisfy the smooth transition between adjacent trajectory segments. Furthermore, the trajectory planning methods of real-time velocity tuning is further investigated to meet the requirements in practical application of industrial robot. At last, the performance of the proposed algorithm is verified by simulation, and the feasibility of the algorithm in practical applications is demonstrated experimentally.

Keywords Multi axis synchronization · Look ahead · Trajectory planning · Real-time velocity tuning

1 Introduction

Trajectory planning, as the core technology of industrial robots and CNC machine tool control, has been studied by a large number of researchers and achieved certain results [1–3]. With the extensive demand and application of industrial robots and machine tools, trajectory planning technology has gradually become mature. Currently, most common research on trajectory planning technology focuses on achieving smooth transition at the corner of space trajectory, minimum contour error under various kinematic constraints or the shortest time of space trajectory planning, etc. By improving smoothness and continuity of machine trajectory, the stability and reliability of robot motion control can be improved.

Industrial robots account for more than 50% of the global robot market. Guan et al. planned the trajectory of the robot by combining the third-order and the fourth-order polynomials to smooth the robot trajectory [4]. The fourth-order polynomial was used to interpolate the trajectory segments at both ends, and the third-order polynomial was used to interpolate the middle trajectory segment. In order to improve the level of industrial manufacturing, the role of robot trajectory planning technology cannot be ignored. In order to realize fast and smooth motion of industrial robots, Wang et al. adopted four-order polynomial to plan the acceleration curve between adjacent two points under kinematic constraints [5]. Similarly, the method of trajectory planning for jerk curve of robot motion is quite common in relevant fields. Fang et al. have proposed two different trajectory planning algorithms successively by this method [6, 7]. One was to divide the jerk curve into acceleration section, constant section, and deceleration section. Under the constraints of velocity, acceleration, and jerk, the acceleration section and deceleration section of jerk curve were planned with *S*-curve, and a new jerk curve was established by the integration of *S*-curve and straight line of constant section. The other was to construct the jerk curve of robot motion using sine function and established a new constraint criterion, which enabled the proposed algorithm to achieve the adjustability of velocity and smoothness of trajectory. This kind of method avoids the chattering caused by the change of velocity during the movement of the robot and keeps the motion process with maximum velocity, acceleration, and jerk as possible, so as to ensure the stability of the motion process and improve the working efficiency of the robot. In order to improve the production efficiency of industrial

✉ Yanyang Liang
liangyanyang@163.com

[1] Wuyi University, Jiangmen 529020, China

manufacturing, optimization of the time-consuming of trajectory is also one of the hot topics in related fields. Liu et al. used spline curves to plan the trajectory in Cartesian space, then used B-spline curves to plan the trajectory in the joint space, and finally used the SQP method to plan the trajectory with minimum time [8]. Gao et al. optimized the trajectory with particle swarm optimization (PSO) and 4–3-4 polynomial interpolation [9]. Fares et al. constructed the trajectory of the robot with third-order polynomial and optimized it with genetic algorithm under three different constraints: kinematics, dynamics, and effective load, and the optimal time trajectory planning algorithm was proposed finally [10]. Huang et al. [11] constructed the optimal trajectory with fifth-order B-spline applying the NSGA-II algorithm proposed by Deb et al. [12]. The resulting motion trajectory algorithm also achieves the continuity of jerk curve. Unlike above literature, Paolo and Dario [13] carried out trajectory planning from the point of view of energy consumption. In addition, there are also some algorithms for specific robots, such as Liu et al. and Wu et al. also proposed different trajectory planning algorithms for Delta robot [14, 15].

Trajectory planning of CNC machine tool with high velocity and precision promotes the efficiency and quality of industrial manufacturing. Zhao et al. put forward a two-stage velocity planning algorithm, reducing the motion time and velocity fluctuation by secondary optimization of a minimum time trajectory planning [16]. Zhang et al. attempted to solve the problem of excessive fluctuation of velocity, acceleration, and jerk at the corner between adjacent straight segments. Under the jerk constraint, to realize smooth transition of motion path, a transition curve with smooth curvature between adjacent straight segments was constructed using fifth-order B-spline curve with constraint on kinematic parameters of the robot. In the subsequent study, a new trajectory planning algorithm with the optimal velocity was proposed based on the linear programming algorithm [17, 18]. Constructing smooth transition at the corner between adjacent straight segments is one of the main methods to improve the smoothness of machine tool movement, and the searchers in related fields have also given many different solutions [19–21]. Besides, the machining accuracy is also a frequent consideration in trajectory planning. In order to reduce the chord height error caused by trajectory planning, Liu et al. constructed a smooth transition trajectory with NURBS curves at the sharp corners of the continuous path [22]. Li et al. and Chen et al. also proposed different solutions to reduce contour error [23, 24].

Most of the algorithms mentioned in the above literatures are trajectory planning for the current segment or adjacent two segments of the robot motion path. Although the planned trajectory can satisfy the constraints on velocity, acceleration, and jerk in the current segment, for multi-segment continuous trajectory planning, the current segment planning is affected by all the previous and afterward segments. In recent years, look-ahead trajectory planning technology has gradually become a research focus, such as Tajima and Sencer [25] and Huang [26], respectively, carried out corresponding research. In this paper, a real-time multi-axis synchronous look-ahead trajectory planning algorithm for a large number of sequentially connected path segments under the kinematic constraints. The planned trajectory is further processed with *S*-shape-type filter, and a real-time velocity tuning algorithm is proposed to meet the requirements of industrial applications. Finally, simulation is carried out in MATLAB environment to verify the feasibility of the algorithm, and experiments are conducted to verify the performance of the algorithm.

2 Multi-axis synchronous look-ahead trajectory planning

2.1 Problem description

In order to control the industrial robot with multiple joints, the path likes line or arc in task space should be interpolated with a large number of sampling points which lie in the line or arc path, and then these points are converted into multi-axis position sequence in the joint space by inverse kinematic of the industrial robot. All the successively adjacent two positions constitute a segment sequence, and the velocity of each segment can be set by the robot operator or simply general distance dividing sampling period.

Suppose there are m positions in the position sequence Q, which can be defined as $Q = [\vec{q}_1, \vec{q}_2, \cdots, \vec{q}_m]$, and each position is a n-dimensional vector, namely, $\vec{q}_i = [q_{i1}, \cdots, q_{in}]^T$, where n is the number of robot's joints, that is, n axes. The adjacent positions $\vec{q}_i$ and $\vec{q}_{i+1}$ constitute the segment i (where $1 \le i \le m-1$), and the generalized setting velocity of the segment i is V_{F_i}. The constraints on maximum velocity and acceleration of each axis are $V^j_{\max}$ and $A^j_{\max}$, respectively $(1 \le j \le n)$.

Briefly, given m positions Q, which constitute $m-1$ segments, and the setting velocity sequence of segments V_F, under the maximum velocity $V^j_{\max}$ and acceleration $A^j_{\max}$ $(1 \le j \le n)$, n axis synchronous look-ahead trajectory planning algorithm is required, and the algorithm must support real-time planning applying the dynamic sliding window technique while the robot is operating and real-time velocity tuning.

2.2 Transition between two adjacent segments

As shown in Fig. 1, the adjacent two segments consist of three adjacent positions, and the generalized length of the two segments is defined as $L_i = \|\vec{q}_{i+1} - \vec{q}_i\|$ and $L_{i+1} = \|\vec{q}_{i+2} - \vec{q}_{i+1}\|$.

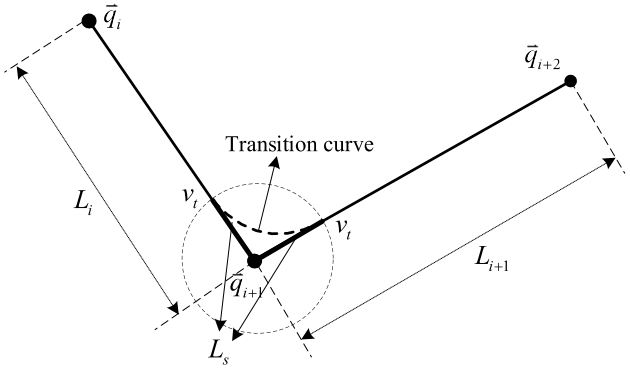

Fig. 1 Transition curve construction between two adjacent segments

The radius of transition zone is $L_s \le \frac{1}{2}\min\left(L_i, L_{i+1}\right)$ and the transition velocity is v_t.

Define the cosine ratio of each axis of each segment as

$$\begin{aligned}\vec{R}_i &= \left[R_{i1}, \cdots, R_{in}\right]^T \\ &= [\cos\theta_{i1}, \cdots, \cos\theta_{in}]^T \\ &= \frac{\vec{q}_{i+1} - \vec{q}_i}{L_i}\end{aligned} \tag{1}$$

$$\vec{R}_{i+1} = \frac{\vec{q}_{i+2} - \vec{q}_{i+1}}{L_{i+1}} \tag{2}$$

The maximum generalized velocity and the maximum generalized acceleration in the segment i are shown in Eq. (3).

$$V_{\max,j} = \min_{1\le j\le n}\{\frac{V^j_{\max}}{R_{i,j}}\}, A_{\max,j} = \min_{1\le j\le n}(\frac{A^j_{\max}}{R_{i,j}}) \tag{3}$$

Then the cosine ratio difference between two adjacent segments is calculated and normalized. Equation is as follows: $\overrightarrow{\Delta R} = \vec{R}_{i+1} - \vec{R}_i$, and normalize $\overrightarrow{\Delta R}$ to $\overrightarrow{\Delta DR} = \overrightarrow{\Delta R}/\left\|\overrightarrow{\Delta R}\right\|$.

The generalized acceleration of the transition zone is $A_p = \min_{1\le j\le n}\{\frac{A^j_{\max}}{\Delta DR_j}\}$; then, the acceleration vector of the transition zone is $\vec{A}_t = \overrightarrow{\Delta DR}\cdot A_p$.

Let the transition time be t_m, and the generalized velocity at the start and end point of the transition zone remains unchanged, and it leads to Eq. (4).

$$v_t\vec{R}_{i+1} = v_t\vec{R}_i + \vec{A}_t t_m \tag{4}$$

That is, $v_t\overrightarrow{\Delta R} = \vec{A}_t t_m = \overrightarrow{\Delta DR}\cdot A_p t_m$, so we have Eq. (5).

$$v_t = \frac{1}{\left\|\overrightarrow{\Delta R}\right\|}A_p t_m \tag{5}$$

Thus, there is $v_t \propto t_m$.

The trajectory equation of the transition curve is shown in Eq. (6).

$$\begin{aligned}L_s\vec{R}_{i+1} &= -L_s\vec{R}_i + \int_0^{t_m}(v_t\vec{R}_i + \vec{A}_t t)dt \\ &= -L_s\vec{R}_i + v_t\vec{R}_i t_m + 0.5\vec{A}_t t_m^2\end{aligned} \tag{6}$$

And then there is

$$\begin{aligned}L_s(\vec{R}_{i+1} + \vec{R}_i) &= v_t\vec{R}_i t_m + 0.5\vec{A}_t t_m^2 \\ &= 0.5(\vec{R}_{i+1} + \vec{R}_i)A_p t_m^2/\left\|\overrightarrow{\Delta R}\right\|\end{aligned}$$

Therefore, Eq. (7) can be deduced.

$$2L_s = A_p t_m^2/\left\|\overrightarrow{\Delta R}\right\| \tag{7}$$

Thus, there are $L_s \propto t_m^2$.

According to the above analysis, if the transition radius L_s is given, then there is $t_m \propto \sqrt{L_s}$ and $v_t \propto \sqrt{L_s}$.

2.3 Determination of maximum allowable transition velocity

- Special situation $\overrightarrow{\Delta R} = \vec{0}$

  In this case, it means that the cosine ratio of the two adjacent segments is the same, and $L_s = 0$. There is no transition zone, and the maximum transition velocity can be set as $v_t = \min(V_{\max,i}, V_{\max,i+1}, V_{F_i}, V_{F_{i+1}})$.
- $\overrightarrow{\Delta R} \ne \vec{0}$

First, the transition radius is given as $L_s = 1/2\min(L_i, L_{i+1})$, and calculate the transition time by Eq. (8).

$$t_m = \sqrt{\frac{2\left\|\overrightarrow{\Delta R}\right\|L_s}{A_p}} \tag{8}$$

Then the maximum transition velocity is calculated as Eq. (9).

$$v_t = \frac{A_p t_m}{\left\|\overrightarrow{\Delta R}\right\|} = \sqrt{\frac{2A_p L_s}{\left\|\overrightarrow{\Delta R}\right\|}} \tag{9}$$

If $v_t > \min(V_{F_i}, V_{F_{i+1}})$, then take $v_t = \min(V_{F_i}, V_{F_{i+1}})$, and the transition radius is calculated by Eq. (10).

$$L_s = v_t\left\|\overrightarrow{\Delta R}\right\|/(2A_p) \tag{10}$$

2.4 Reverse velocity planning

Suppose k ($k < m$) segments need to be looked ahead, that is the length of sliding window is k. In the reverse velocity

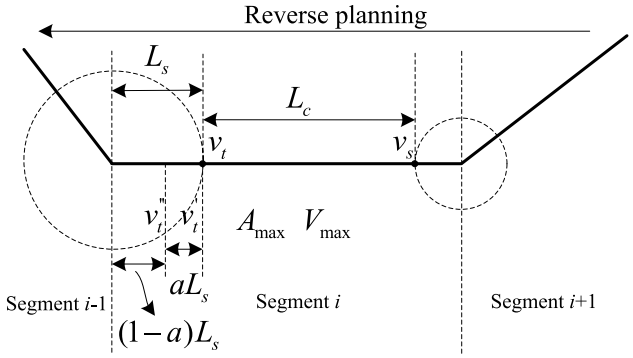

Fig. 2 Reverse look-ahead trajectory planning

planning stage, the final velocity of the last segment (segment k) is set to be zero, and then velocity planning is conducted from segment k to segment $k-1$, …, until from segment 2 to segment 1. Take segment $i+1$ and segment i as example for detailing the reverse velocity planning algorithm, as shown in Fig. 2.

If $v_s \geq v_t$, then v_t is reachable and no further processing is required. If $v_s < v_t$, then it is necessary to determine whether v_t can be reached. Firstly, $v_t' = \sqrt{v_s^2 + 2A_{\max}L_c}$ is calculated. If $v_t' \geq v_t$, then v_t is reachable. Otherwise, it is necessary to calculate the maximum transition velocity that can be reached and revise transition radius.

Let the maximum allowable transition velocity be v_t''. According to Fig. 2, there is Eq. (11).

$$v_t''^2 - v_t'^2 = 2A_{\max}aL_s = 2aA_{\max}L_s \tag{11}$$

Since $v_t'' = v_t\sqrt{1-a}$, Eq. (12) is obtained.

$$\begin{aligned} a &= (v_t^2 - v_t'^2)/(2A_{\max}L_s + v_t^2) \\ &= (v_t^2 - v_s^2 - 2A_{\max}L_c)/(2A_{\max}L_c + v_t^2) \end{aligned} \tag{12}$$

Then the transition length and transition velocity can be updated by Eq. (13).

$$L_{sn} = (1-a)L_s, v_{tn} = v_t\sqrt{1-a} \tag{13}$$

Taking v_{tn} as v_s of the previous segment (segment $i-1$), the reverse velocity planning continues until to the segment 1.

2.5 Forward velocity planning

Forward velocity planning is carried out for the first segment, as shown in Fig. 3, which is mainly divided into two cases.

1. $v_s \geq v_t$

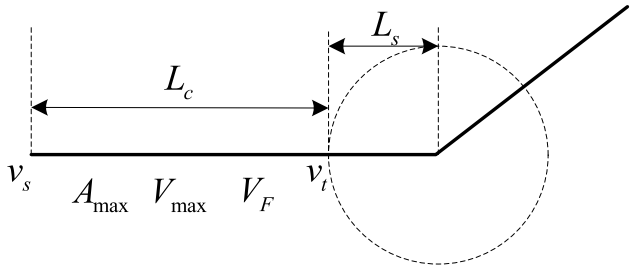

Fig. 3 Forward velocity planning

Reverse velocity planning already ensures that v_s can slow down to v_t and $v_s \leq V_F$. Firstly, the critical velocity v_c is calculated by the following equation:

$$v_c^2 - v_s^2 + v_c^2 - v_t^2 = 2A_{\max}L_c$$

Thus Eq. (14) can be obtained.

$$v_c = \sqrt{(v_s^2 + v_t^2 + 2A_{\max}L_c)/2} \tag{14}$$

If $v_c \leq V_F$, the velocity curve is shown in the first figure in Fig. 4. If $v_c > V_F$, the velocity curve is shown in the second figure in Fig. 4. If $v_s = v_c$, the velocity curve is shown in the third figure in Fig. 4.

2. $v_s < v_t$

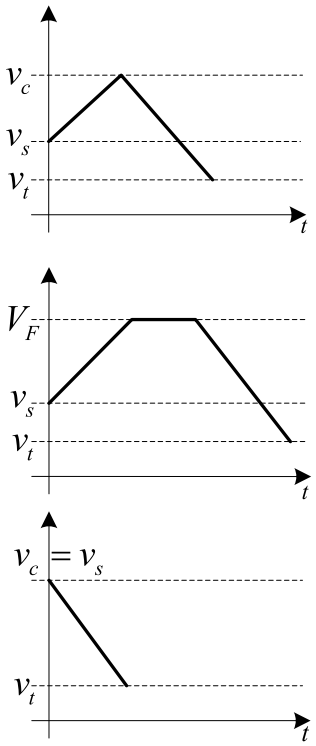

Fig. 4 Deceleration case

This situation can be further divided into two cases.

2.1 $v_t^2 - v_s^2 < 2A_{\max}L_c$

This situation indicates that the velocity can be raised to v_t, and the critical speed is calculated as Eq. (15).

$$v_c = \sqrt{\left(v_t^2 + v_s^2 + 2A_{\max}L_c\right)/2} \tag{15}$$

If $v_c \le V_F$, the velocity curve is shown in the left figure in Fig. 5. If $v_c > V_F$, the velocity curve is shown in the right figure in Fig. 5.

2.2 $v_t^2 - v_s^2 > 2A_{\max}L_c$

This situation indicates that the velocity cannot be raised to v_t. If the transition radius is 0, namely, $L_s = 0$, the velocity can be increased directly. The maximum velocity can be calculated by Eq. (16).

$$v_{tn} = \sqrt{v_s^2 + 2A_{\max}L_c} \tag{16}$$

The velocity curve is shown in Fig. 6.

If $L_s \neq 0$, it is necessary to reduce the transition radius and increase the velocity as much as possible. As shown in Fig. 7, if the ratio of reducing the length of the transition section is a, there are the following three equations:

$$v_t^{'} = \sqrt{v_s^2 + 2A_{\max}L_c} \tag{17}$$

$$v_t^{''} = v_t\sqrt{1-a} \tag{18}$$

$$v_t^{''2} - v_t^{'2} = 2A_{\max}aL_s \tag{19}$$

By solving the above three equations, Eq. (20) can be obtained.

$$a = \left(v_t^2 - v_s^2 - 2A_{\max}L_c\right)/\left(2A_{\max}L_s + v_t^2\right) \tag{20}$$

Then Eq. (21) can be obtained to update the transition radius and transition velocity.

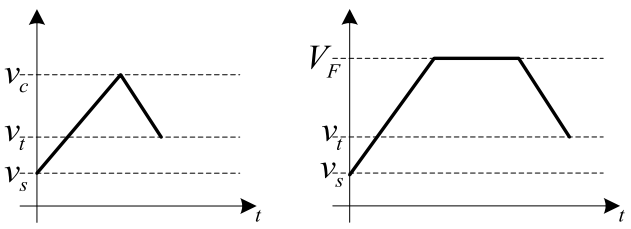

Fig. 5 Acceleration case 1

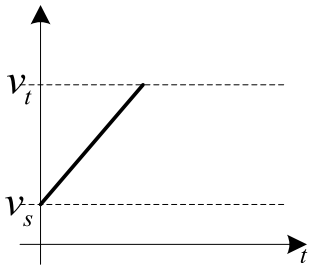

Fig. 6 Acceleration case 2

$$\begin{aligned} L_{cn} &= L_c + aL_s \\ L_{sn} &= (1-a)L_s \\ v_{tn} &= v_t\sqrt{1-a} \end{aligned} \tag{21}$$

The velocity curve obtained is shown in Fig. 6.

2.6 Transition trajectory planning

There is no transition zone while in the last segment or the direction cosine vectors of two adjacent segments are equal. Otherwise, if the transition zone exists, that is, $L_s \neq 0$,then trajectory planning for the transition zone is required. Since the generalized velocity of the transition zone is constant, the transition trajectory can be calculated easily.

As shown in Fig. 8, $\vec{q}_{ts}$ and $\vec{q}_{te}$ are the starting and ending velocity vector of the transition zone, v_t is the generalized velocity, L_s is the transition radius, A_p is the generalized acceleration, and $\vec{A}_t = \overrightarrow{\Delta DR} \cdot A_p$ is the acceleration vector. According to Eq. (5) and Eq. (7), the transition duration is $t_m = 2L_s/v_t$, and the velocity of all the axis at the given time can be obtained from Eq. (22).

$$\vec{V}(t) = v_t\vec{R}_i + \vec{A}_t \cdot t \tag{22}$$

The corresponding position of all the axis at the given time can be obtained from Eq. (23).

$$\vec{q}(t) = \vec{q}_{ts} + \int_0^t \vec{V}(t)dt = \vec{q}_{ts} + v_t\vec{R}_i \cdot t + 0.5\vec{A}_t \cdot t^2 \tag{23}$$

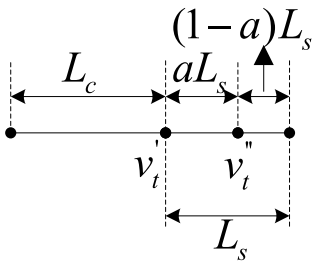

Fig. 7 The transition radius and transition velocity revise

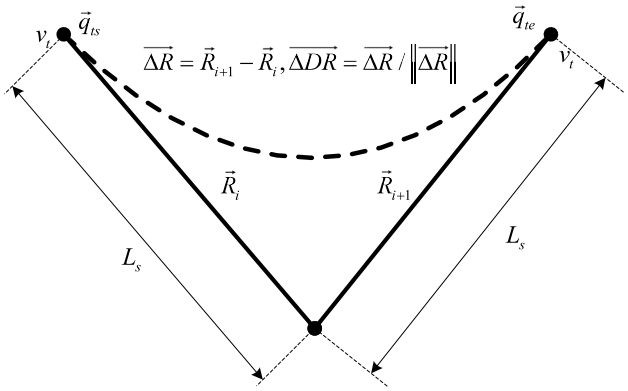

Fig. 8 Trajectory planning in the transition zone

2.7 Multi-axis synchronous look-ahead trajectory planning algorithm

2.7.1 [Algorithm 1] Multi-axis synchronous look-ahead trajectory planning (MSLTP)

The flowchart of the algorithm, shown in Fig. 9, can be described as the following:

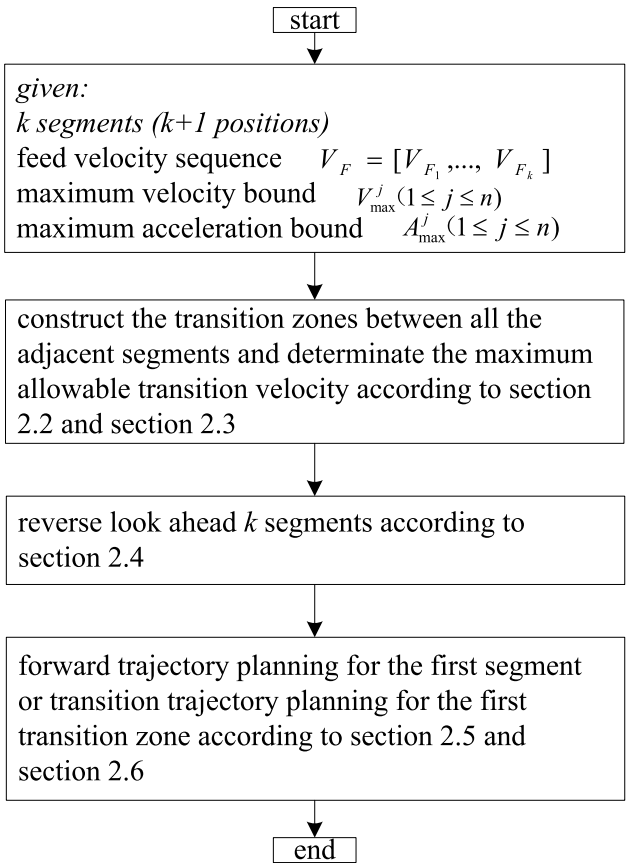

Fig. 9 Flowchart of multi-axis synchronous look-ahead trajectory planning algorithm

1. Given $k+1$ positions, every position is a vector with dimension n, constituting k segments, given feed velocity sequence $V_F = [V_{F_1}, ..., V_{F_k}]$, the maximum velocity bound $V^j_{\max} 1 \le j \le n)$, and the maximum acceleration bound $A^j_{\max} 1 \le j \le n)$.
2. Construct the transition zones between all the adjacent segments and determinate the maximum allowable transition velocity according to Sects. 2.2 and 2.3.
3. Set the velocity of the last position (the velocity usually sets as zero), conduct reverse velocity look-ahead planning from the segment k to segment 1, and revise transition zones and the maximum allowable transition velocity according to Sect. 2.4.
4. Forward trajectory planning for the first segment and transition trajectory planning for the first transition zone according to Sects. 2.5 and 2.6.

3 Dynamic real-time velocity tuning

3.1 Dynamic real-time acceleration and deceleration

Real-time velocity tuning is designed to improve dynamic performance and flexibility. It is the basis for dynamic tracking, such as visual tracking, weld tracking, or other guidance functions.

Real-time velocity change mainly includes real-time velocity increase and real-time velocity decrease. Real-time velocity increase can be realized only by updating the target feed velocity $V_{F_i}(1 \le i \le m-1)$ of all segments and execute the MSLTP algorithm again immediately.

In order to achieve real-time deceleration, firstly, update the feed velocity $V_{F_i}(1 \le i \le m-1)$; secondly, recalculate the first transition zone and the maximum transition velocity by the following methods; and finally, execute the MSLTP algorithm again.

As shown in Fig. 10, the absolute values of the generalized length, the initial generalized velocity, and the generalized maximum acceleration of the current segment are set as L_c, v_s, and A_m, respectively. The allowable transition radius, transition velocity, and generalized maximum acceleration of the transition segment are set as L_s, v_t, and A_p, respectively. The feed velocities of the current and the next segment are V_{F_1} and V_{F_2}, respectively, and the cosine ratios of the two adjacent segments are $\vec{R}_1$ and $\vec{R}_2$, respectively.

Set $v_{tm} = \min\left(v_t, V_{F_1}, V_{F_2}\right)$ and $\overrightarrow{\Delta R} = \vec{R}_2 - \vec{R}_1$, then equations $L_{sn} = \left\|\overrightarrow{\Delta R}\right\| v_{tm}^2 \Big/ \left(2A_p\right)$ and $L_{cn} = L_c + L_s - L_{sn}$ can be calculated, update $L_s = L_{sn}, L_c = L_{cn}$.

If $v_s > v_{tm}$, calculate $v'_{tm} = \sqrt{v_s^2 - 2A_m L_c}$. If $v'_{tm} \le v_{tm}$, update the data according to Eq. (24).

$$v_{tn} = v_{tm}, V_F = v_{tn} \tag{24}$$

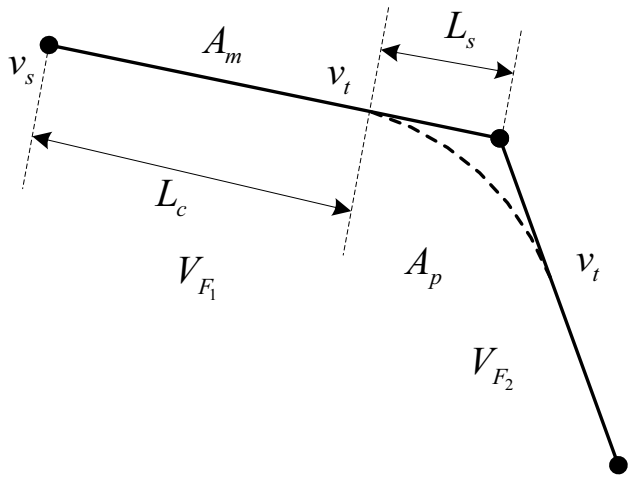

Fig. 10 Trajectory planning for the first and second segments

If $v_{tm}^{'} > v_{tm}$, then L_c and L_s need to be revised and transition velocity must be recalculated. As shown in Fig. 11, since the velocity cannot be reduced to v_{tm}, L_c needs to be reduced. Let the reduction ratio be a; there is the following equation:

$$v_t^{'2} - v_{tm}^2 = 2A_p a L_c \Big/ \left\|\overrightarrow{\Delta R}\right\| \tag{25}$$

$$v_s^2 - v_t^{'2} = 2A_m(1-a)L_c \tag{26}$$

According to Eq. (25) and Eq. (26), Eq. (27) can be obtained.

$$a = \left(v_s^2 - v_{tm}^2 - 2A_m L_c\right) \Big/ \left(2L_c\left(A_p \Big/ \left\|\overrightarrow{\Delta R}\right\| - A_m\right)\right) \tag{27}$$

Then the corresponding data can be updated according to the following formula:

$$L_{sn} = aL_c + L_s, L_{cn} = (1-a)L_c \tag{28}$$

$$v_{tn} = v_t^{'} = \sqrt{2A_p L_{sn} \Big/ \left\|\overrightarrow{\Delta R}\right\|}, V_F = v_{tn} \tag{29}$$

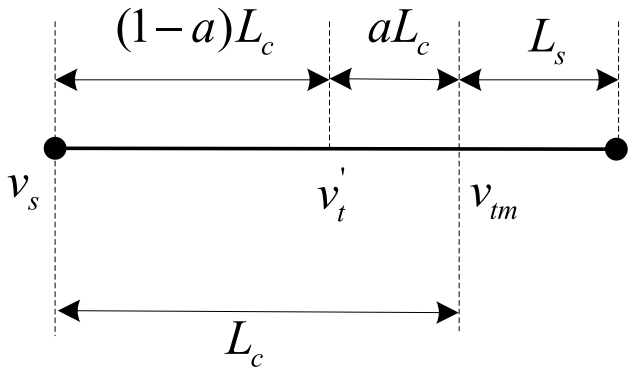

Fig. 11 Real-time deceleration for the first transition zone

3.2 Real-time trajectory planning for stop command

When receiving the stop command, the first step is to judge whether the actuator can stop in the current segment. Calculate $L = L_c + L_s$ and $L^{'} = v_s^2/(2A_m)$. If $L^{'} \le L$, it means that the velocity can be reduced to 0 in the current segment, then only this segment needs trajectory planning. Otherwise, the transition zone should be revised to reduce the velocity as much as possible. Set the velocity after deceleration (the transition velocity) as v_{tm}, then the following equation can be obtained.

$$L_s^{'} = v_{tm}^2 \frac{\left\|\overrightarrow{\Delta R}\right\|}{2A_p}, L_c^{'} = L - L_s^{'} = \frac{v_s^2 - v_{tm}^2}{2A_m}$$

Then Eq. (30) is obtained from above.

$$v_{tm}^2 = \frac{v_s^2 - 2A_m L}{1 - A_m \Big/ \left(A_p \Big/ \left\|\overrightarrow{\Delta R}\right\|\right)}, L_s^{'} = \frac{\left\|\overrightarrow{\Delta R}\right\|}{2A_p} v_{tm}^2 \tag{30}$$

Let $v_t = v_{tm}, L_s = L_s^{'}$, and continue to plan the next segment using the same trajectory planning method until the velocity in a segment can be reduced to 0 (Fig. 12).

4 Digital simulation and result analysis

4.1 S-type digital filtering

The velocity curve obtained by algorithm MSLTP is trapezoidal, which can be transformed into *S*-type velocity curve through an *S*-type digital filter. *S*-type digital filter is designed based on the principle of digital convolution, which is very suitable for the application of real-time planners.

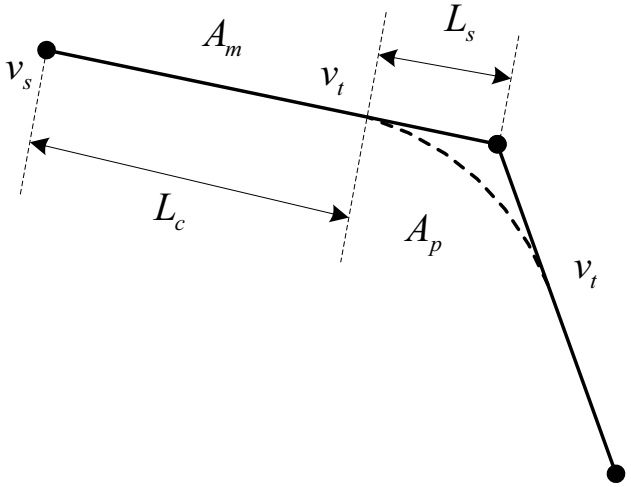

Fig. 12 Real-time trajectory planning for stop command response

S-type digital filter is realized by series connection of two linear-type filters. The discrete transfer function of linear-type filters is shown in the following formula:

$$H_L(z) = \frac{1}{m}\frac{1-z^{-m}}{1-z^{-1}} \tag{31}$$

The transfer function of *S*-type digital filter is shown in Eq. (32).

$$\begin{aligned} H_s(z) &= H_L(z) * H_L(z) \\ &= \frac{1}{m}\frac{1-z^{-m}}{1-z^{-1}} * \frac{1}{m}\frac{1-z^{-m}}{1-z^{-1}} \end{aligned} \tag{32}$$

The effect of linear-type filter and *S*-type filter on input step signals is shown in Fig. 13. If the input signal is set to $V_i(k), k = 1, 2, \ldots, V_i(k)\big|_{k\le 0} = 0$, the digital implementation of *S*-type filter can adopt the following way:

$$\begin{aligned} V_{ot}(k) &= \tfrac{1}{m}\big(V_i(k) - V_i(k-m)\big) + V_{ot}(k-1) \\ V_o(k) &= \tfrac{1}{m}\big(V_{ot}(k) - V_{ot}(k-m)\big) + V_o(k-1) \end{aligned} \tag{33}$$

4.2 Multi-axis real-time synchronous look-ahead trajectory planning algorithm

4.2.1 [Algorithm 2] multi-axis real-time synchronous look-ahead trajectory planning (MRTSLTP)

The flowchart of the algorithm, shown in Fig. 14, can be described as the following:

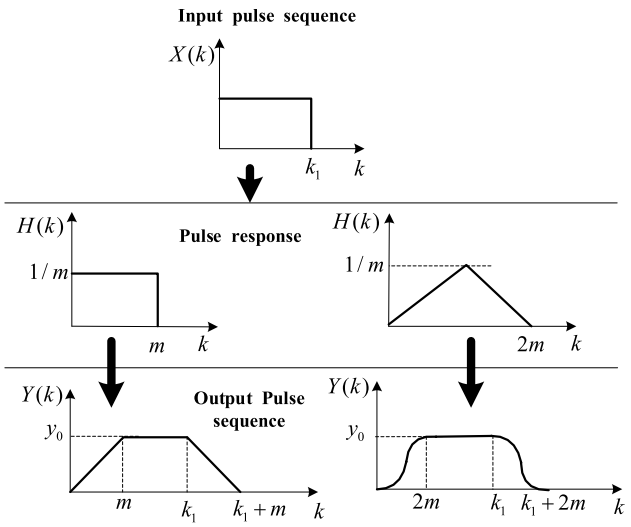

Fig. 13 Linear-type and *S*-type digital filters

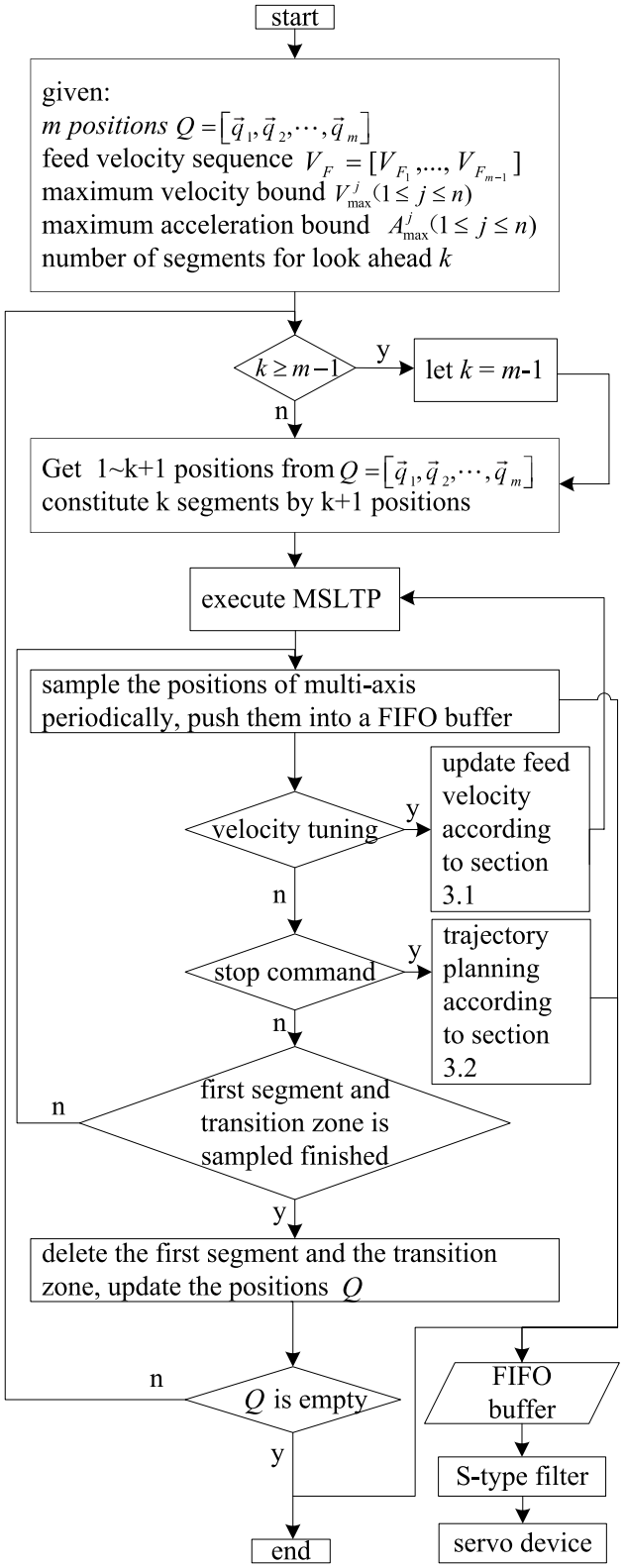

Fig. 14 Flowchart of multi-axis real-time synchronous look-ahead trajectory planning algorithm

1. Given m positions, every position is a vector with dimension n, constituting $m-1$ segments, given feed velocity sequence $V_F = [V_{F_1}, ..., V_{F_{m-1}}]$, the maximum velocity bound $V^j_{\max}(1 \le j \le n)$, and the maximum acceleration bound $A^j_{\max} 1 \le j \le n)$, the number of segments to look-ahead k.
2. If the look-ahead segment number k is larger than the number of total segment $m-1$, let $k = m-1$. Get the $1 \sim k+1$ positions from position sequence$Q = [\vec{q}_1, \vec{q}_2, \cdots, \vec{q}_m]$, which constitute k segments. And then execute the MSLTP algorithm.
3. According to the MSLTP algorithm, trajectory of the first segment and transition zone is obtained, then the positions of multi-axis can be sampled periodically from the trajectory, push them into a FIFO buffer. These positions will pass through the *S*-type filter and feed into the servo device finally.
4. If velocity tuning command is received while sampling, then update feed velocity according to Sect. 3.1, and execute the MSLTP algorithm again. If the stop command is received while sampling, then begin the trajectory planning according to Sect. 3.2, and the algorithm will terminate sampling is finished.
5. When the first segment and the transition zone are sampled finished, the segment and transition zone will be deleted, the positions Q will be updated. If Q is not empty, then return to step (2); otherwise, the algorithm will terminate.

4.3 Simulation

Each joint of a 6-DOF serial industrial robot corresponds to a motion axis. The coordinate system configuration of the robot is shown in Fig. 15, and the corresponding robot DH parameters are shown in Table 1. The ai and di in Fig. 15 correspond to the a_i and d_i in Table 1, respectively, and the rightmost column of Table 1 is the range of joint angles. The velocity range of the six axes is set to [−4,4] rad/s, and the acceleration range is [−30,30] rad/s^2.

Table1 DH parameters of 6-DOF serial robot

Link i	a_{i-1}(mm)	α_{i-1}(°)	θ_i(°)	d_i(mm)	Angle range
$i=1$	0	0	$\theta_1 0$	0	−170 ~ 170
$i=2$	85	−90	θ_2 (−90)	0	−92 ~ 135
$i=3$	380	0	$\theta_3 0$	0	−129 ~ 116
$i=4$	85	−90	$\theta_4 0$	425	−160 ~ 160
$i=5$	0	90	$\theta_5 0$	0	−120 ~ 120
$i=6$	0	−90	$\theta_6 0$	0	−360 ~ 360
$i=7$	0	0	0	85	0

The orientation and position of a rigid body can be represented as a 4 × 4 homogeneous matrix $T = [R, p; 0, 0, 0, 1]$, where R is the 3 × 3 orthogonal orientation matrix and p is the 3 × 1 position vector. Construct an arc path with start point [−0.898, 0.066, 0.435, 98.09; −0.215, 0.798, −0.563, 55.47; −0.385, −0.599, −0.702, −537.73; 0, 0, 0, 1], middle point [−0.901, −0.001, 0.434, 53.26; 0.003, 1.000, 0.009, 380.43; −0.434, 0.009, −0.901, −419.81; 0, 0, 0, 1], and end point [−0.898, −0.066, 0.435, 228.00; 0.215, 0.797, 0.564, 87.47; −0.384, 0.600, −0.702, −396.72; 0, 0, 0, 1]. Firstly, without considering the velocity and acceleration limits of axes, the EOT (end of tool) of the robot moves along the arc path with a *S*-type velocity profile in the task space. Secondly, 202 points in the path are obtained by sampling with 2-ms period. The corresponding position sequence of the six axes is obtained with inverse kinematics, as shown in Fig. 16. The corresponding velocity and acceleration sequences can also be calculated easily, as shown in Figs. 17 and 18.

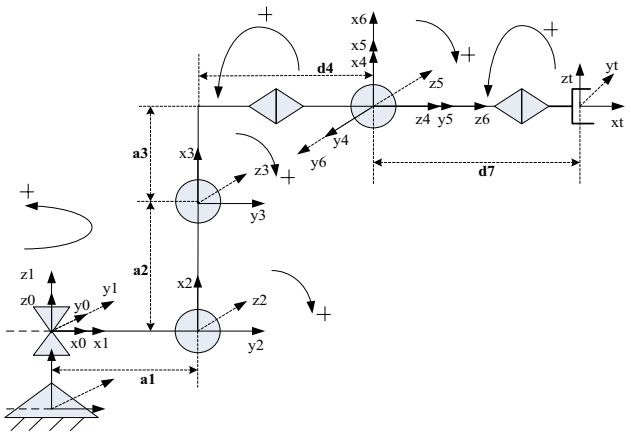

Fig. 15 Coordinate system configuration of 6-DOF serial robot

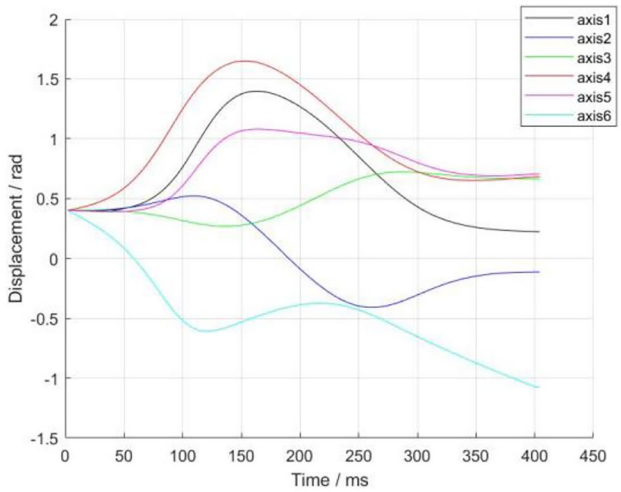

Fig. 16 Displacement curve of original target trajectory

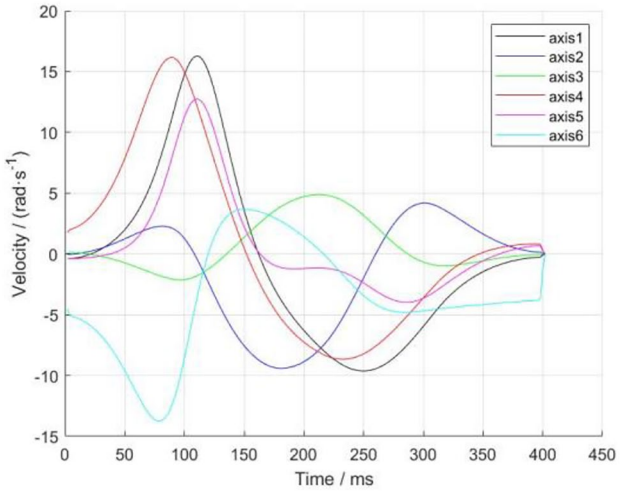

Fig. 17 Velocity curve of the original target trajectory

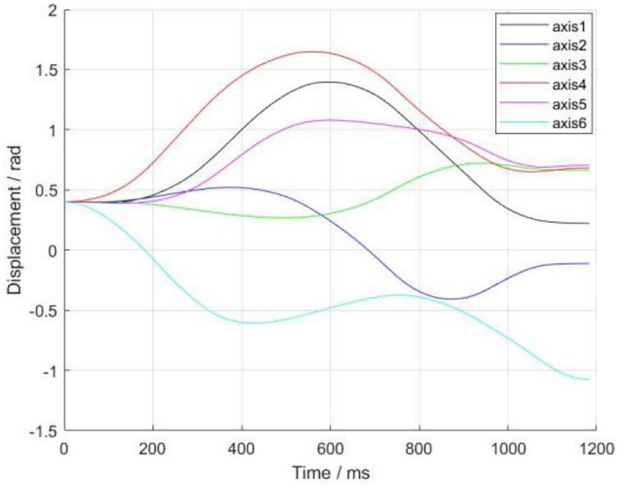

Fig. 19 Displacement curve planned by MRTSLTP algorithm

The 202 positions constitute 201 segments; the feed velocity of every segment can be set as length of the segment dividing 2 ms, look-ahead 100 segments, a fine interpolation period of 1 ms. The MRTSLTP algorithm is carried out; the position, velocity, and acceleration obtained after planning are shown in Figs. 19, 20, and 21. If the velocity is dynamically reduced to 1/10 of the original at 400 ms and then increased to 10 times of the current at 1500 ms, MRTSLTP algorithm can response immediately, and the velocity and acceleration curves obtained are shown in Figs. 22, 23, and 24.

It can be seen from Fig. 17 and Fig. 18 that the velocity and acceleration of the given target trajectory are far beyond the given range of velocity and acceleration. After trajectory planning with MRTSLTP algorithm under the constraints of velocity and acceleration, the target trajectory is significantly improved, which are shown in Fig. 20 and Fig. 21; the velocity and acceleration of the planned target trajectory are effectively constrained within the given range, and both curves are very smooth, which can make the robot move more stable in practical applications. Figure 23 shows the velocity curve of the target trajectory after real-time velocity tuning; the MRTSLTP algorithm can respond to velocity tuning command very quickly.

5 Algorithm implementation and experiments

5.1 Control schemes and implementation of MRTSLTP algorithm

The industrial six-axis motion control platform is built to test the practicability of the algorithm. The hardware

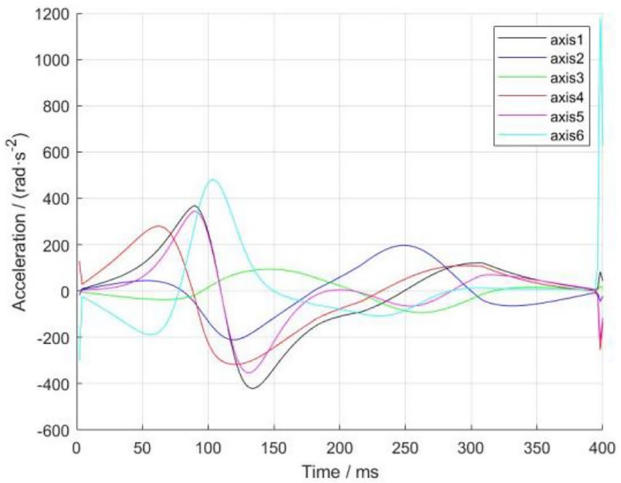

Fig. 18 Acceleration curve of original target trajectory

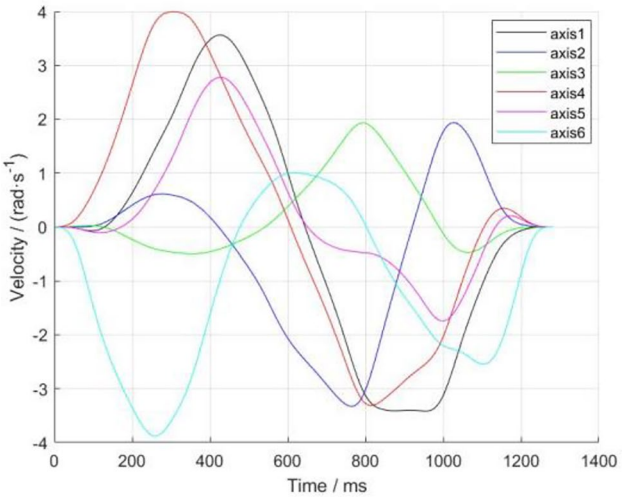

Fig. 20 Velocity curve planned by MRTSLTP algorithm

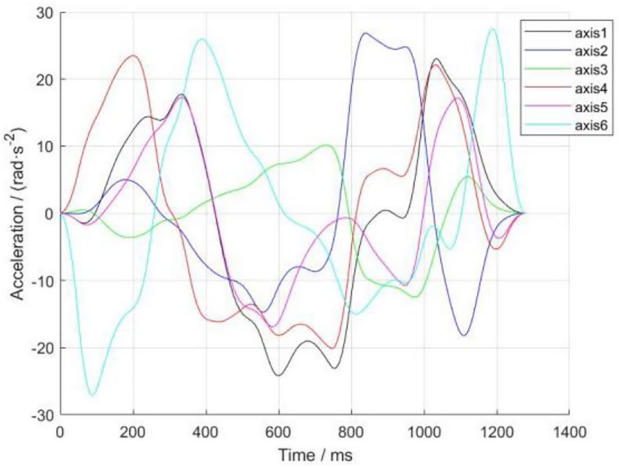

Fig. 21 Acceleration curve planned by MRTSLTP algorithm

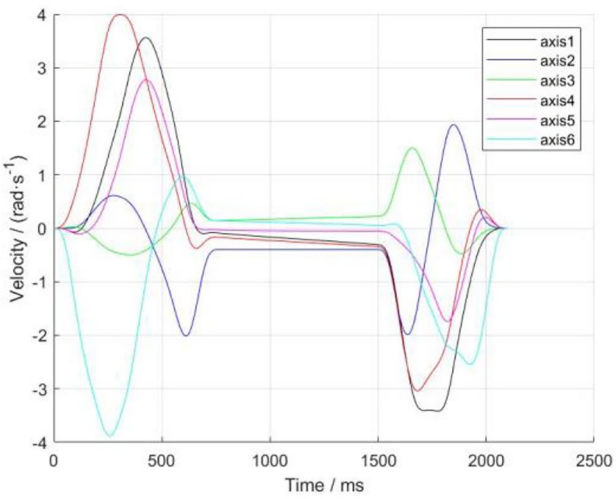

Fig. 23 Velocity curve planned by MRTSLTP with velocity tuning

platform of the control system consists of one master station equipment and multiple slave station equipment. The CTH3-C series motion controller of Hexin is selected as the master station equipment, the H1A servo driver of Hexin is selected as the slave station equipment, and the M series servo motor is selected as the driving device. The communication between the master station and the slave station is carried out through EtherCAT (Ethernet Control Automation Technology). Figure 25 is the overall framework of the control system. Figure 26 shows the hardware platform of the industrial six-axis motion control system.

Figure 27 is the framework diagram of the control system software platform; the software develop environment is CODESYS (V3.5 sp11 version) in the industrial personal computer. The control system software consists two modules, including the human–computer interface module and the task module. The human–computer interface module provides the parameter input, system operation state information, and system control buttons. The task module composes two sub-tasks: one is the trajectory planning task whose cycle is 20 ms and the other is the communication task whose cycle is 1 ms. The communication task feeds position data to servo device every 1 ms periodically, and has the highest priority.

Figure 28 is the control system data flow diagram. There are four buffers in this system. The buffer1 mainly consists all the position sequence needed to be planned and feed velocity sequence, buffer2 mainly consists the current k segments to be planned, buffer3 consists the sampling positions after the trajectory planning, and buffer4 consists the smooth positions after passing through the S-type filter.

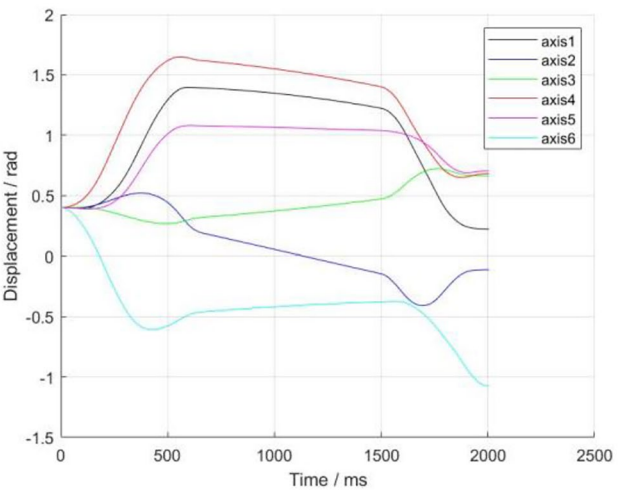

Fig. 22 Displacement curve planned by MRTSLTP with velocity tuning

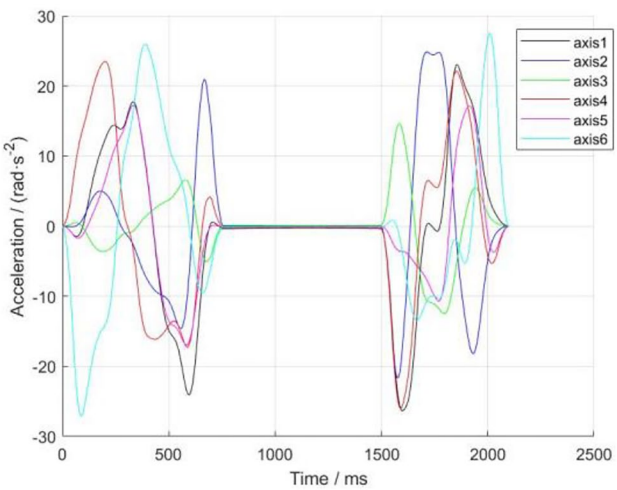

Fig. 24 Acceleration curve planned by MRTSLTP with velocity tuning

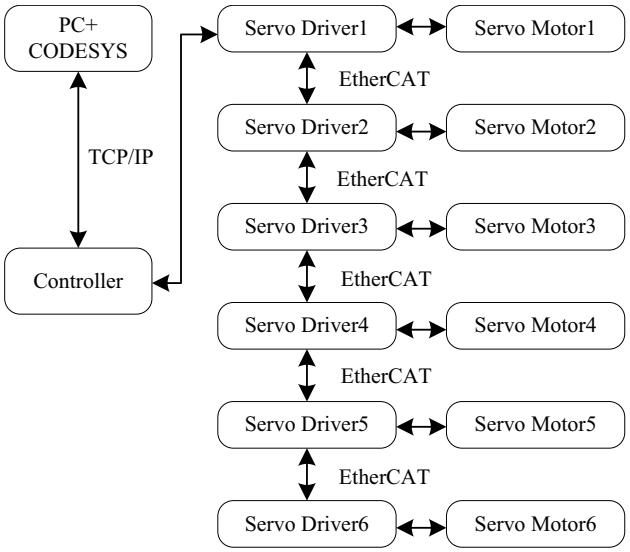

Fig. 25 Overall diagram of the control system

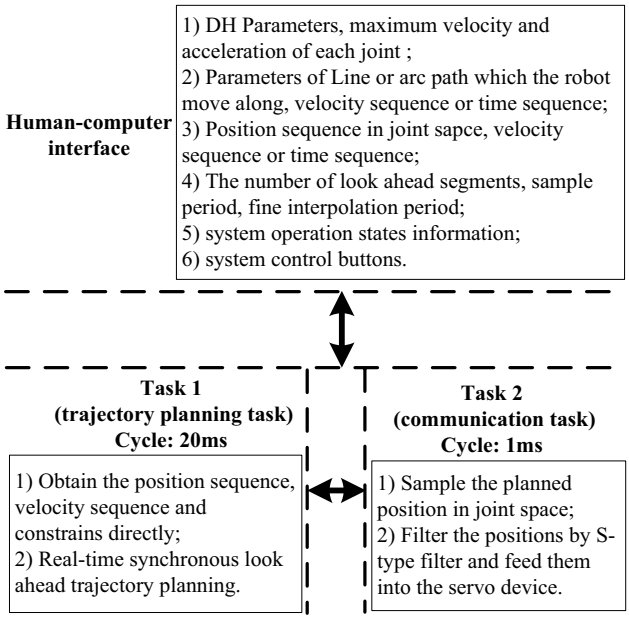

Fig. 27 Control system software platform block diagram

5.2 Experiment and analysis

According to the same DH model, let the orientation be $R = [-0.8979, 0.0653, 0.4354; -0.2155, 0.7972, -0.5640; -0.3839, -0.6002, -0.7017]$; construct an arc path with start point [98.09, 55.47, – 537.7], middle point [161.6, 3.531, – 407.3], and end point [245.8, 17.6, – 270.1]; and construct a line path with start point [93.59, 214, – 63.34] and end point [98.09, 55.47, – 537.7]. 1.25 m/s is used as the generalized velocity of line path, and 0.8 m/s is used as the generalized velocity of arc path, taking the line and arc in the task space as the target path, respectively. The target path is rough interpolated, and the positions in joint space are obtained by inverse kinematics of the interpolation points. Taking the position sequence as input, the servo motors are controlled by the control system software to track the positions with feed generalized velocity.

Fig. 26 Hardware platform of six-axis motion control system

Figures 29, 30, 31, 32, 33, and 34 show the original displacement, velocity, and acceleration curves of the joints by rough interpolation and inverse kinematics along the line and arc path, respectively. Figures 35, 36, 37, and 38 show the results obtained by experiment. The preset values shown in Figs. 35, 36, 37, and 38 are the theoretical value obtained by the MRTSLTP algorithm, and the actual values are the feedback of the servo motor.

5.3 Analysis of path error

Figures 35, 36, 37, and 38 show that the preset and the actual trajectories of all the joints are nearly coincident. The differences between the preset and the actual values are calculated for comparison and analysis. The error curves of joint displacement corresponding to the two trajectories are shown in Figs. 39 and 40. The error of joint displacement is effectively controlled, and the maximum error is within 0.001 rad.

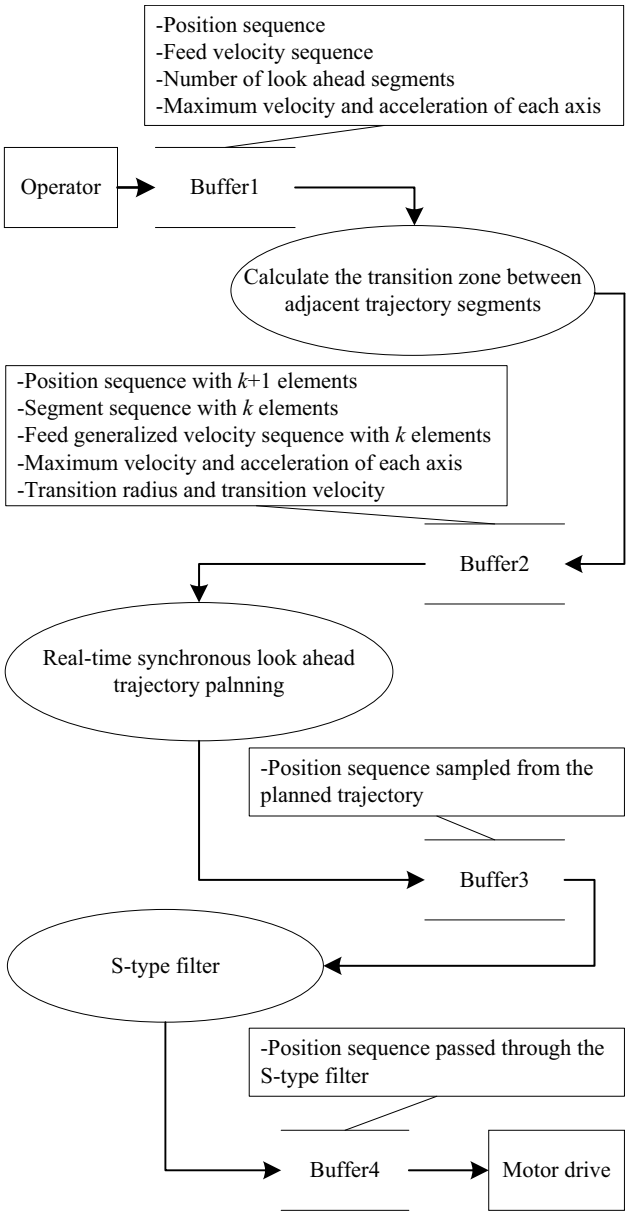

Fig. 28 Control system data flow diagram

To further verify the performance of tracking line and arc paths in task space, the joint trajectories are converted into space trajectories by the robot forward kinematics, and the results are shown in Figs. 41, 42, 43, and 44.

Figures 41 and 43 show that the initial paths, preset paths, and actual paths of both line and arc trajectories have good overlap. It can be seen from Figs. 42 and 44 that the maximum errors of the line and the arc path are around 1 mm and 0.6 mm, respectively.

The errors shown in Figs. 42 and 44 are relatively large. The length of the line path is 500 mm, and the actual moving time is around 0.8 s; it is inferred that the large error is caused by the high velocity. If the rough interpolation period

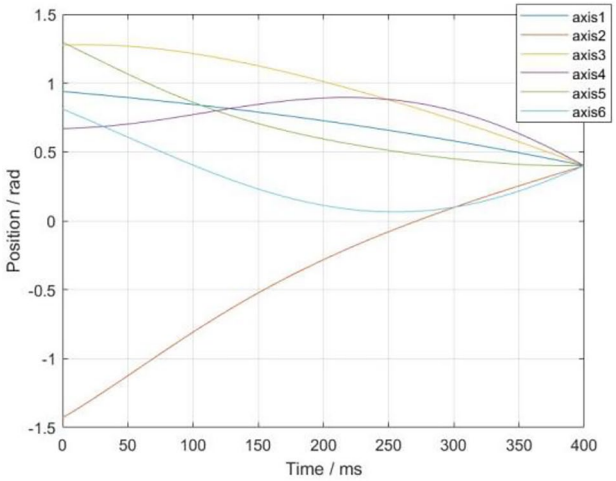

Fig. 29 Joint displacement curve of the original line path

is set to 10 ms, that is slowing down the velocity five times. The experiments are conducted again according to the above conditions, and the results are shown in Figs. 45, 46, 47, 48, 49, 50, 51, 52, 53, and 54.

As can be seen from Figs. 45, 46, 47, 48, 49, and 50, the trajectories of each joint under the above conditions almost completely overlap, and the joint angle errors are constrained to be within [−0.0004, 0.0003] radians. As can be seen from Figs. 51, 52, 53, and 54, the initial, preset, and actual trajectories also have very good overlap, and the maximum position errors of the line and arc paths are constrained to be below 0.1 mm and 0.06 mm, respectively. It can be concluded that the proposed MRTSLTP algorithm can meet the performance requirements of robot trajectory planning, and has good executability in practical applications.

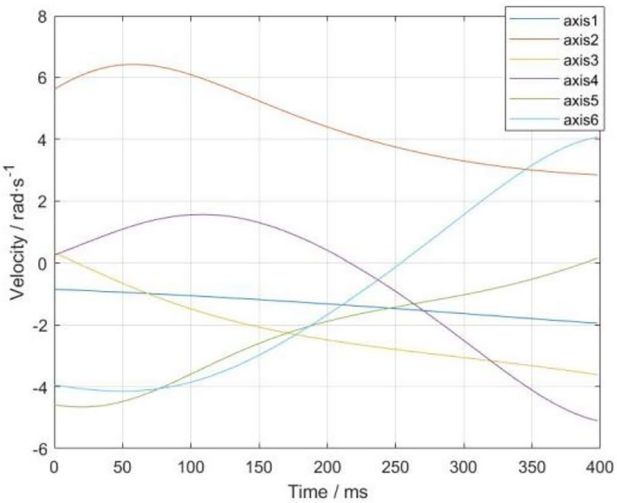

Fig. 30 Joint velocity curve of the original line path

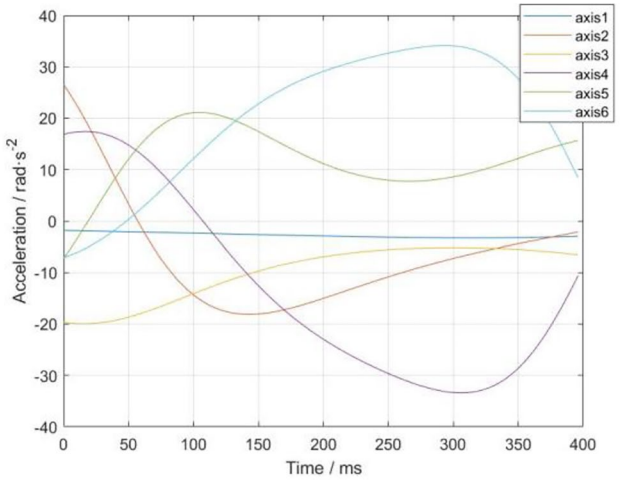

Fig. 31 Joint acceleration curve of the original line path

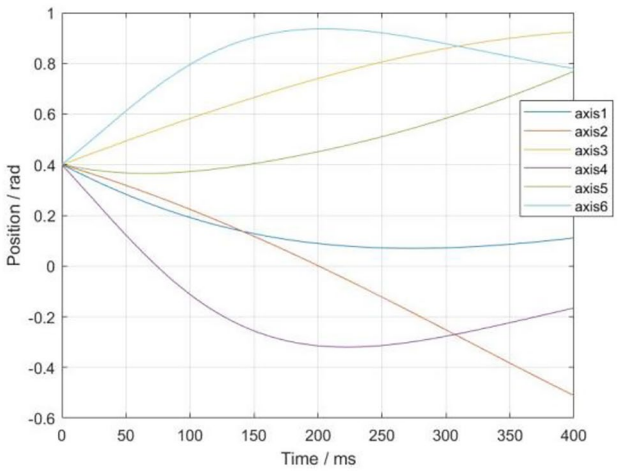

Fig. 32 Joint displacement curve of the original arc path

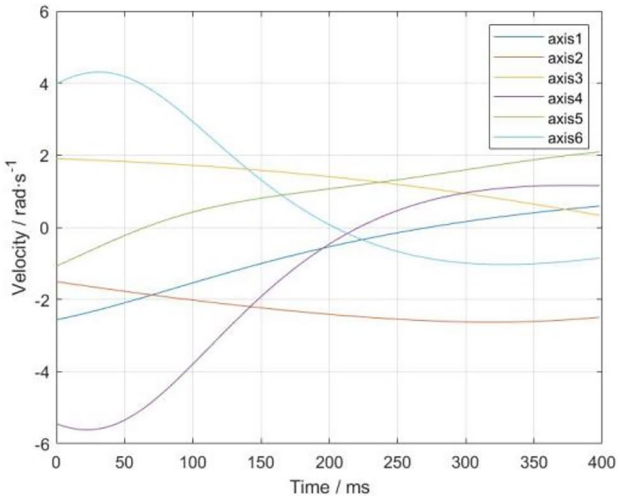

Fig. 33 Joint velocity curve of the original arc path

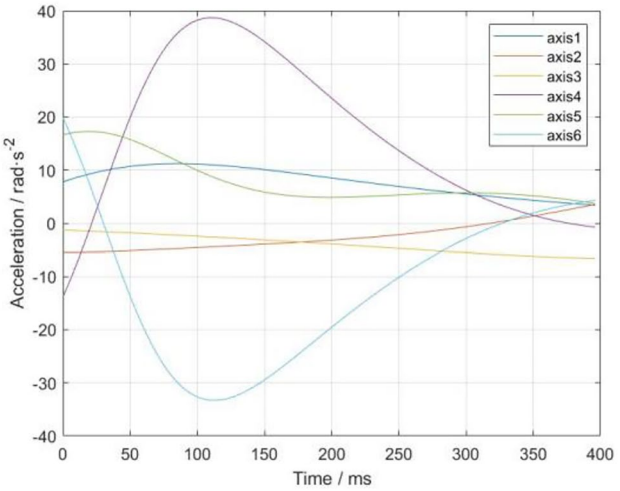

Fig. 34 Joint acceleration curve of the original arc path

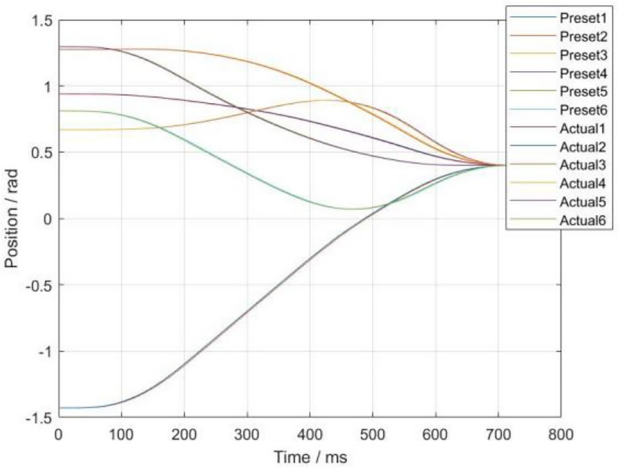

Fig. 35 Joint displacement curve of the line path by experiment

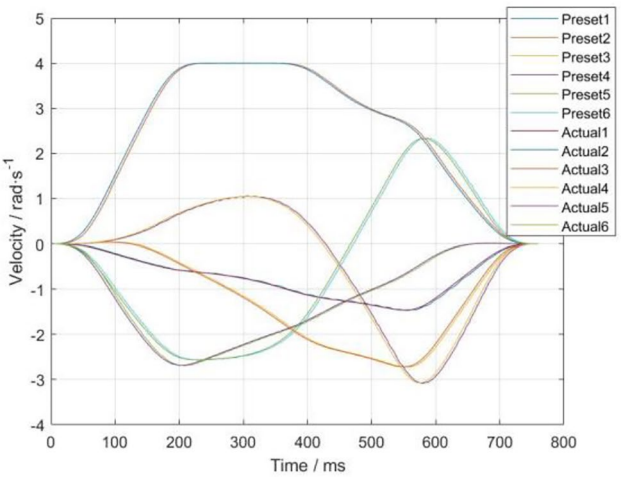

Fig. 36 Joint velocity curve of the straight path by experiment

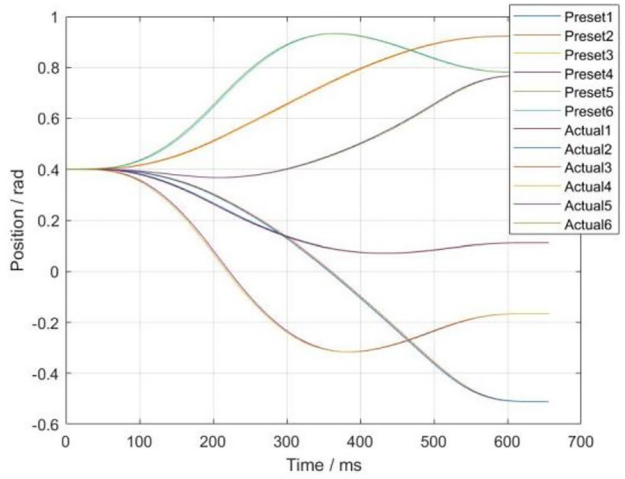

Fig. 37 Joint displacement curve of the arc path by experiment

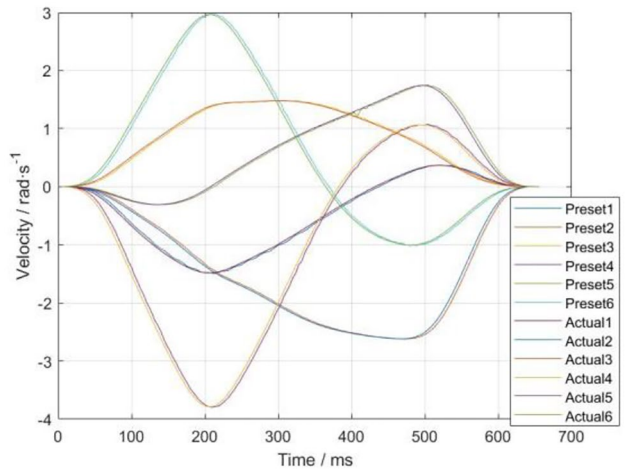

Fig. 38 Joint velocity curve of the arc path by experiment

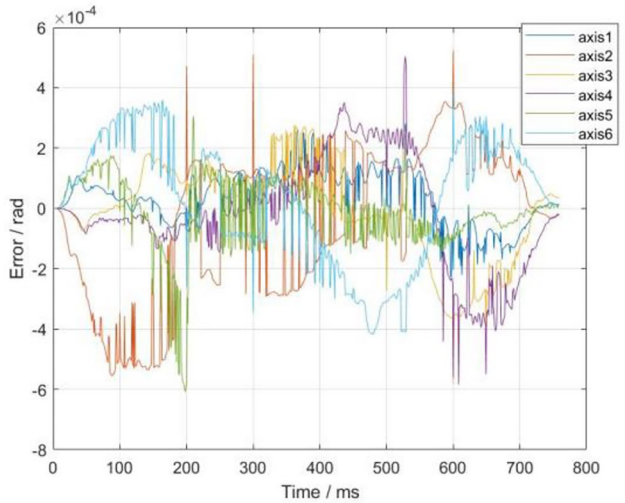

Fig. 39 Error curve of joint displacement of line path

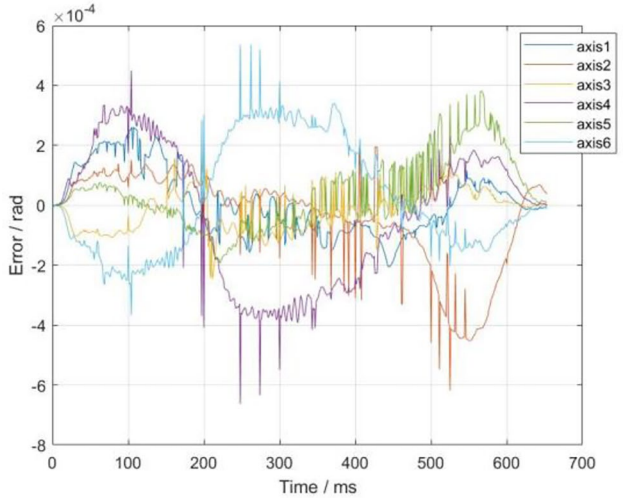

Fig. 40 Error curve of joint displacement of arc path

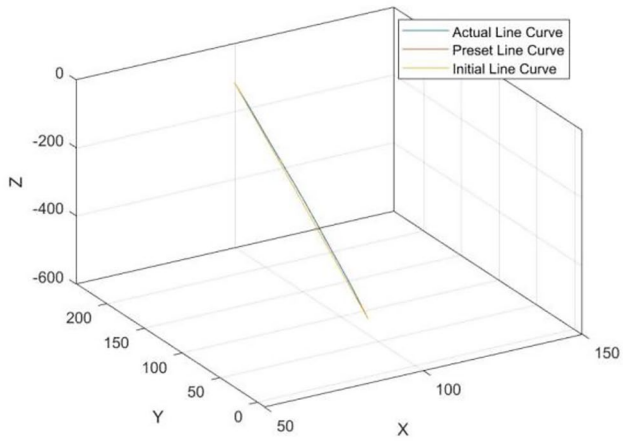

Fig. 41 Line path obtained by forward kinematics of joint displacement

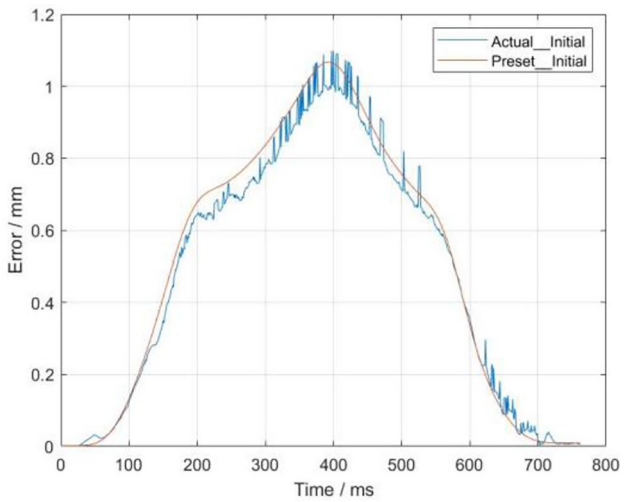

Fig. 42 Error curve of line path obtained by forward kinematics of joint displacement

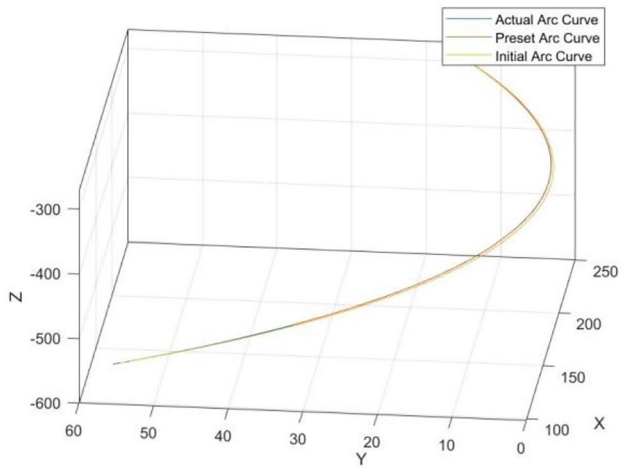

Fig. 43 Arc path obtained by forward kinematics of joint displacement

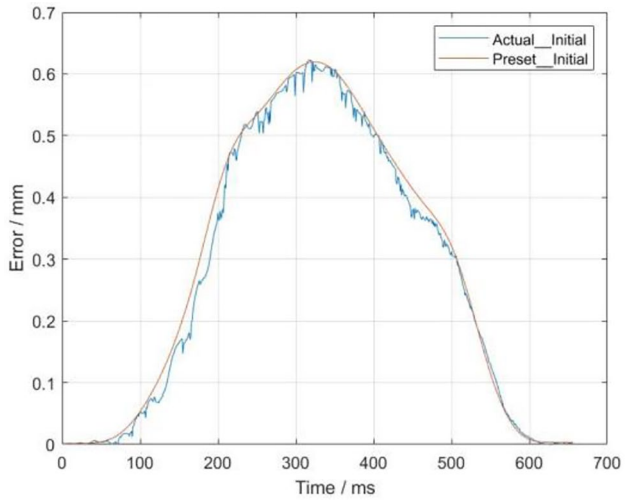

Fig. 44 Error curve of arc path obtained by forward kinematics of joint displacement

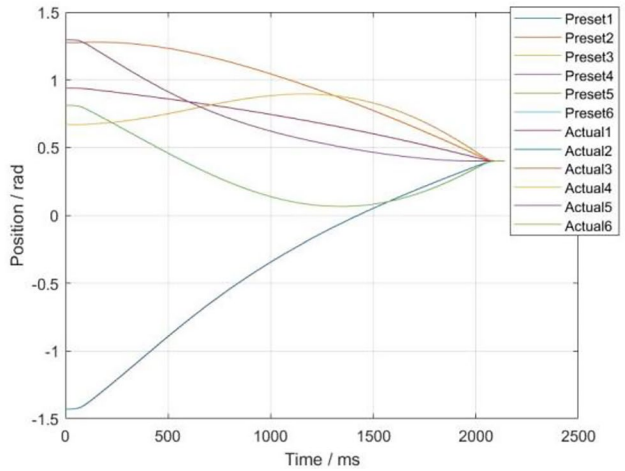

Fig. 45 Joint displacement curve of the line path obtained by the experiment with low velocity

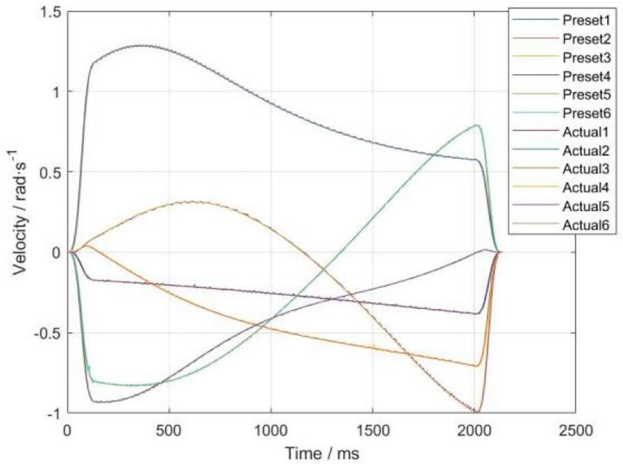

Fig. 46 Joint velocity curve of the line path obtained by the experiment with low velocity

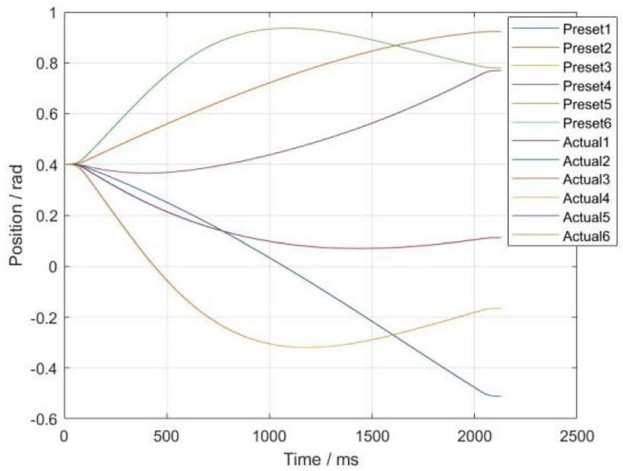

Fig. 47 Joint displacement curve of the arc path obtained by the experiment with low velocity

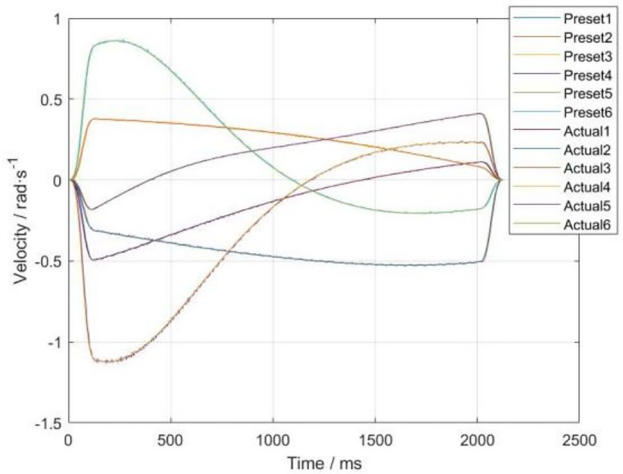

Fig. 48 Joint velocity curve of the arc path obtained by the experiment with low velocity

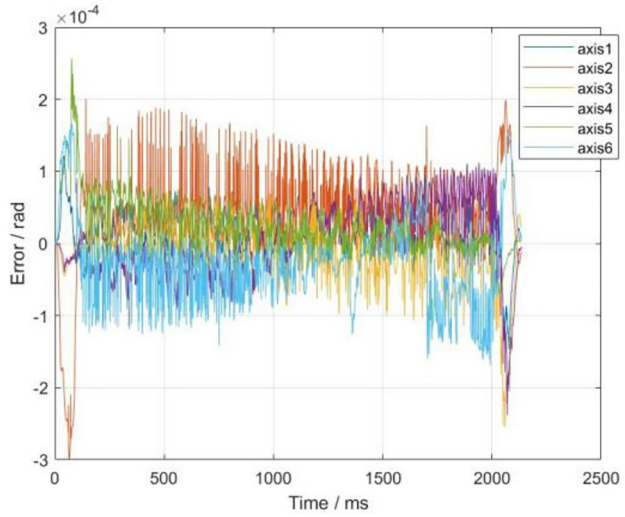

Fig. 49 Error curve of joint displacement of line path with low velocity

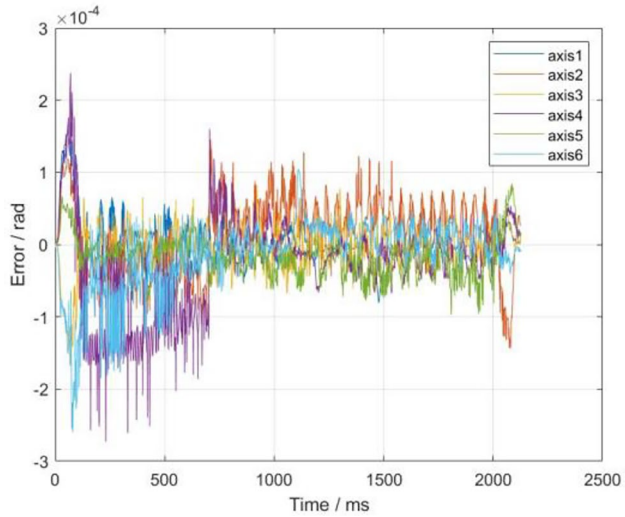

Fig. 50 Error curve of joint displacement of arc path with low velocity

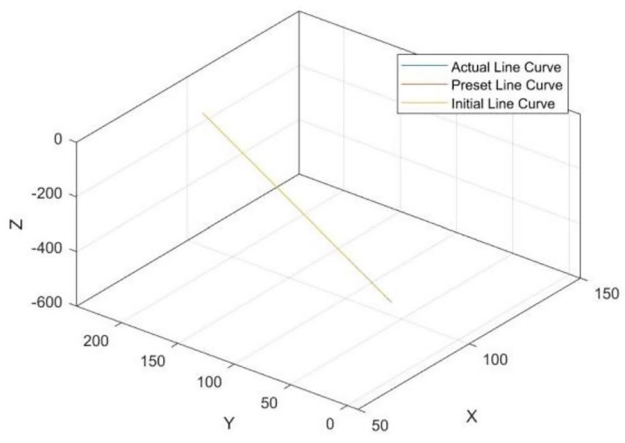

Fig. 51 Line path obtained by forward kinematics of joint displacement with low velocity

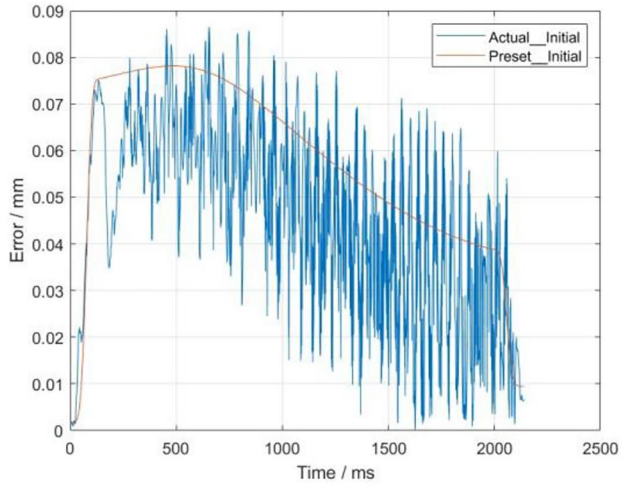

Fig. 52 Error curve of line path obtained by forward kinematics of joint displacement with low velocity

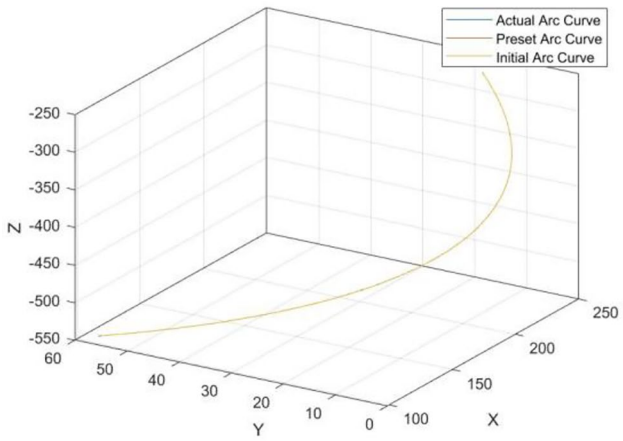

Fig. 53 Arc path obtained by forward kinematics of joint displacement with low velocity

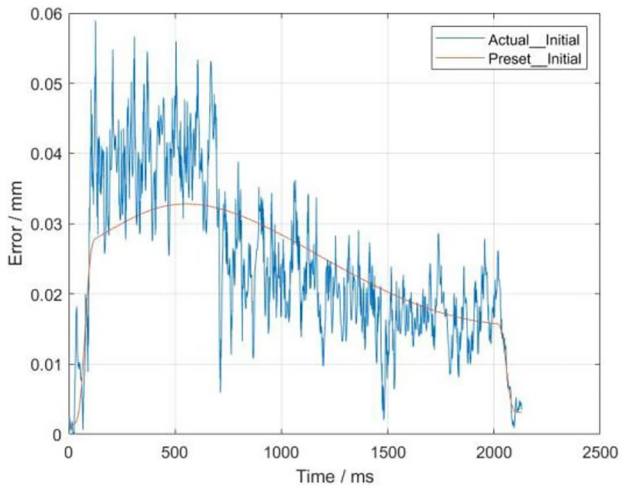

Fig. 54 Error curve of arc path obtained by forward kinematics of joint displacement with low velocity

6 Conclusion

Focus on kinematic constraints and smooth transition of a large number of continuous segments in trajectory planning, a multi-axis real-time synchronous look-ahead trajectory planning algorithm is proposed. The algorithm performs multi-axis look-ahead trajectory planning in real time according to given position sequence, feed velocity sequence under the constraints on maximum velocity, and acceleration of the axes. The algorithm also supports online real-time velocity tuning and real-time stop command response. Experiments show that the trajectory planned by the proposed algorithm can satisfy the constraints very well whatever the feed velocity and acceleration of the given initial trajectory are. In addition, the experimental results also show that the algorithm can respond to the velocity increasing and decreasing in real time. The subsequent work is to optimize the algorithm considering the position error of the robot in the actual trajectory tracking, and to connect the servo motor to the load for further experimental verification.

Author contribution Yanyang Liang performed the design of the algorithm and the writing of the manuscript. Chaozhi Yao and Wei Wu performed the code writing and experimental analysis. Li Wang and Qiongyao Wang organized the paper and revised the manuscript.

Funding This study is supported by the National Natural Science Foundation of China (No. 51905384) and Research Initiation Fund of Wuyi University (No. 409170190241).

Availability of data and materials All data generated or analyzed during this study are included in this published article.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Conflict of interest The authors declare no competing interests.

References

1. Ji W, Wang LH (2019) Industrial robotic machining: a review. Int J Adv Manuf Technol 103:1239–1255. https://doi.org/10.1007/s00170-019-03403-z
2. Gasparetto A, Zanotto V (2008) A technique for time-jerk optimal planning of robot trajectories. Robot Comput Integr Manuf 24:415–426. https://doi.org/10.1016/j.rcim.2007.04.001
3. Guan XQ, Wang JD (2011) Trajectory planning theory and method of industrial robot. Int Conf Comput Res Dev 340–343. https://doi.org/10.1109/ICCRD.2011.5764146
4. Guan YS, Yokoi K, Stasse O, Kheddar A (2005) On robotic trajectory planning using polynomial interpolations. IEEE Int Conf Robot Biomim 111–116. https://doi.org/10.1109/ROBIO.2005.246411
5. Wang H, Wang H, Huang JH, Zhao B, Quan L (2019) Smooth point-to-point trajectory planning for industrial robots with kinematical constraints based on high-order polynomial curve. Mech Mach Theor 139:284–293. https://doi.org/10.1016/j.mechmachtheory.2019.05.002
6. Fang Y, Hu J, Liu WH, Shao QQ, Qi J, Peng YH (2019) Smooth and time-optimal S-curve trajectory planning for automated robots and machines. Mech Mach Theor 137:127–153. https://doi.org/10.1016/j.mechmachtheory.2019.03.019
7. Fang Y, Qi J, Hu J, Wang WM, Peng YH (2020) An approach for jerk-continuous trajectory generation of robotic manipulators with kinematical constraints. Mech Mach Theory 153:103957. https://doi.org/10.1016/j.mechmachtheory.2020.103957
8. Liu HS, Lai XB, Wu WX (2013) Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints. Robot Comput Integr Manuf 29:309–317. https://doi.org/10.1016/j.rcim.2012.08.002
9. Gao MY, Ding P, Yang YX (2015) Time-optimal trajectory planning of industrial robots based on particle swarm optimization. Int Conf Instrum Meas Comput Commun Control 1934–1939. https://doi.org/10.1109/IMCCC.2015.410
10. Fares JA, Iyad FA, Rasha MA, Mohamed A (2017) Statistical evaluation of an evolutionary algorithm for minimum time trajectory planning problem for industrial robots. Int J Adv Manuf Technol 89:389–406. https://doi.org/10.1007/s00170-016-9050-1
11. Huang JS, Hu PF, Wu KY, Zeng M (2018) Optimal time-jerk trajectory planning for industrial robots. Mech Mach Theory 121:530–544. https://doi.org/10.1016/j.mechmachtheory.2017.11.006
12. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197
13. Paolo B, Dario R (2019) Energy-efficient design of multipoint trajectories for Cartesian robots. Int J Adv Manuf Technol 102:1853–1870. https://doi.org/10.1007/s00170-018-03234-4
14. Liu C, Cao GH, Qu YY, Cheng YM (2020) An improved PSO algorithm for time-optimal trajectory planning of Delta robot in intelligent packaging. Int J Adv Manuf Technol 107:1091–1099. https://doi.org/10.1007/s00170-019-04421-7
15. Wu MK, Mei JP, Zhao YQ, Niu WT (2020) Vibration reduction of delta robot based on trajectory planning. Mech Mach Theory 153:104004. https://doi.org/10.1016/j.mechmachtheory.2020.104004
16. Zhao K, Li SR, Kang ZJ (2019) Smooth minimum time trajectory planning with minimal feed fluctuation. Int J Adv Manuf Technol 105:1099–1111. https://doi.org/10.1007/s00170-019-04308-7
17. Zhang Y, Zhao MY, Ye PQ, Jiang JL, Zhang H (2018) Optimal curvature-smooth transition and efficient feedrate optimization method with axis kinematic limitations for linear toolpath. Int J Adv Manuf Technol 99:169–179. https://doi.org/10.1007/s00170-018-2496-6
18. Zhang Y, Ye PQ, Wu JQ, Zhang H (2018) An optimal curvature-smooth transition algorithm with axis jerk limitations along linear segments. Int J Adv Manuf Technol 95:875–888. https://doi.org/10.1007/s00170-017-1274-1
19. Wang H, Wu JH, Liu C, Xiong ZH (2018) A real-time interpolation strategy for transition tool path with C2 and G2 continuity. Int J Adv Manuf Technol 98:905–918. https://doi.org/10.1007/s00170-018-2242-0
20. Zhang Q, Gao XS, Li HB, Zhao MY (2017) Minimum time corner transition algorithm with confined feedrate and axial acceleration

for nc machining along linear tool path. Int J Adv Manuf Technol 89:941–956. https://doi.org/10.1007/s00170-016-9144-9
21. Zhang LQ, Du JF (2018) Acceleration smoothing algorithm based on jounce limited for corner motion in high-speed machining. Int J Adv Manuf Technol 95:1487–1504. https://doi.org/10.1007/s00170-017-1272-3
22. Liu XH, Peng JQ, Si L, Wang ZB (2017) A novel approach for NURBS interpolation through the integration of acc-jerk-continuous-based control method and look-ahead algorithm. Int J Adv Manuf Technol 88:961–969. https://doi.org/10.1007/s00170-016-8785-z
23. Li BR, Zhang H, Ye PQ (2018) Error constraint optimization for corner smoothing algorithms in high-speed CNC machine tools. Int J Adv Manuf Technol 99:635–646. https://doi.org/10.1007/s00170-018-2489-5
24. Chen WC, Chen CS, Lee FC, Chen LY (2019) High speed blending motion trajectory planning using a predefined absolute accuracy. Int J Adv Manuf Technol 104:2179–2193. https://doi.org/10.1007/s00170-019-03973-y
25. Tajima S, Sencer B (2020) Real-time trajectory generation for 5-axis machine tools with singularity avoidance. CIRP Ann 69:349–352. https://doi.org/10.1016/j.cirp.2020.04.050
26. Huang HM (2018) An adjustable look-ahead acceleration/deceleration hybrid interpolation technique with variable maximum feedrate. Int J Adv Manuf Technol 95:1521–1538. https://doi.org/10.1007/s00170-017-1277-y