



A high-precision non-uniform rational B-spline interpolator based on S-shaped feedrate scheduling

Lijin Fang¹ · Guanghui Liu¹ · Qiang Li² · Hualiang Zhang³

Received: 8 September 2021 / Accepted: 21 May 2022 / Published online: 14 June 2022
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2022

Abstract

Parametric interpolation has been widely used in robot control because of its advantages over traditional linear or circular interpolation. The NURBS interpolation based on S-shaped feedrate scheduling can realize the smooth continuous motion of robots, but it also faces two challenges. One challenge is that when the NURBS curve is split into multiple blocks, if one or more blocks are too short, the final feedrate profile is often discontinuous. The other challenge is that the NURBS curve cannot be obtained entirely due to calculation errors. In this paper, a high-precision NURBS interpolator based on S-shaped feedrate scheduling is proposed. In this interpolator, a bidirectional velocity scanning scheme is developed to effectively handle the too short split blocks of the NURBS curve. Meanwhile, an optimization algorithm is proposed to ensure the precision of NURBS curves. The feasibility and advantages of the proposed method are verified through case studies.

Keywords NURBS · Interpolation · S-shaped feedrate profile · Industrial robot

1 Introduction

Parametric interpolation for the non-uniform rational B-spline (NURBS) curve has become an important part of the control of robot path planning [1, 2]. An effective NURBS interpolator can not only obtain accurate contour trajectories but also have smooth dynamic performance. The discontinuities in feed motion can evoke the natural modes of the mechanical structure and the servo control system, which will degrade the contouring accuracy [3]. To achieve a shock-free and smooth machining process, researchers have considered jerks when designing interpolators.

With the inequality constraints of physical limits including joint velocity, acceleration, and jerk, an iterative optimization strategy that takes the motion time as the optimization goal was proposed in [4, 5]. In [6], Hashemian et al.

proposed to utilize the reparameterization technique which expresses the relationship between the motion path parameter of the flank machining toolpath and the system time variable by an optimal transfer function to minimize the total jerk of the tool's motion. Compared with these optimization strategies, the S-shaped feedrate profile does not require iterative optimization while satisfying the physical limits, so it is widely used in the field of robotics. An S-shaped feedrate scheduling approach was proposed for the NURBS interpolator to ensure the continuity of acceleration and control the range of jerk in [7, 8]. Ni et al. [9] also developed a bidirectional adaptive feedrate scheduling method based on the S-shaped ACC/DEC algorithm for high-speed two-axis machining. Though the S-shaped method is widely used in smooth feedrate scheduling because of its simplicity and smoothness, the conventional S-shaped feedrate profile usually cannot deal with all the conditions, e.g., the target displacement is too short. In this case, the robot cannot reach the ending velocity from the starting velocity even by accelerating or decelerating with the max acceleration and jerk. To address this issue, this paper proposes a bidirectional velocity scanning scheme to overcome this drawback of the S-shaped feedrate profile.

Another big challenge is the calculation error that consists of two parts: the error caused by the inaccurate estimation of the arc length, and the truncation error caused by Taylor expansion

✉ Lijin Fang
ljfang@mail.neu.edu.cn

¹ Faculty of Robots Science and Engineering, Northeastern University, Shenyang 110819, China

² Neuroinformatics, Group & CITEC, Bielefeld University, Bielefeld 33619, Germany

³ State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110169, China

[10, 11]. With these errors, it is difficult to achieve highly precise and integral parametric planning. Many studies have been carried out in this field. A high-order polynomial spline was utilized to fit the relationship of arc length and the curve parameter in [12]. A quartic equation-based interpolation method was proposed in [13], which works well in reducing feedrate fluctuation. Lo proposed a feedback parametric interpolation method, which approximates the desired point iteratively [14]. A real-time predictor–corrector interpolator was proposed by Tsai and Chen which can guarantee that the feedrate command error is within a specified tolerance [15]. Besides, a feedback interpolator with arc-length compensation and feedback correction was developed by Zhao et al. in [16, 17], which achieves good efficiency and accuracy. In this paper, an interpolation method is proposed based on the S-shaped profile with arc length prediction and feedback correction.

The process of the proposed method is as follows. Firstly, the NURBS curve is partitioned into several continuous blocks, and the speed at the junctions is generated based on the limitation of the maximum tangential acceleration, the maximum velocity, the chord error tolerance, etc. Then, a bidirectional velocity scanning is employed to calculate the ultimate feedrate at the junctions between the blocks. Afterward, the Simpson rule and the adaptive-corrected method proposed in this study are utilized to estimate the arc length of each NURBS sub-curve. Finally, the NURBS parameter u is calculated through Taylor’s expansion.

The main contributions of this paper are summarized as follows:

- A bidirectional velocity scanning scheme is proposed to deal with the case of too short target displacement of S-shaped feedrate profile.
- An interpolation feedrate interpolator based on S-shaped profile with arc length prediction and feedback correction is proposed to ensure precise and integral NURBS curve planning.

The rest of this paper is organized as follows. In Sect. 2, the background including the definition of the NURBS curve, the derivatives of the NURBS curve, the curvature of the NURBS curve, and the method to calculate its arc length are briefly introduced. Then, a complete feedrate scheduling scheme for the NURBS curve is introduced in Sect. 3. In Sect. 4, experiments are conducted on a 6-DOF industrial robot system, and the conclusions are presented in Sect. 5.

2 Background

2.1 Definition of a NURBS curve

A NURBS curve $C(u)$ can be defined as follows:

$$C(u) = \frac{\sum_{i=0}^n N_{i,p}(u)w_iP_i}{\sum_{i=0}^n N_{i,p}(u)w_i} \tag{1}$$

where $\{P_i\}$ indicates the control points; $\{w_i\}$ indicates the corresponding weights of $\{P_i\}$; $(n + 1)$ is the number of control points; p is the degree of a NURBS curve; $\{N_{i,p}(u)\}$ indicates the B-spline basis functions of the p -th degree defined on the non-uniform knot vector $U = \{u_0, u_1, \dots, u_m\} = \underbrace{\{a, \dots, a\}}_{p+1}, u_{p+1}, \dots, u_{m-p-1}, \underbrace{b, \dots, b}_{p+1}$; $(m + 1)$ is the number of knots. Generally, it is assumed that $a = 0$, $b = 1$, and $w_i > 0$ for all i . The degree p , the number of control points $(n + 1)$, and the number of knots $(m + 1)$ satisfy $m = n + p + 1$.

The B-spline basis functions $\{N_{i,p}(u)\}$ of the p -th degree are recursively defined as follows:

$$N_{i,0}(u) = \begin{cases} 1 & u_i \leq u < u_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$N_{i,0}(u) = \frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u) \tag{3}$$

$i = 0, 1, \dots, n$

In this paper, the knot vector is derived following the Rosenfeld method [18]:

$$U = \left\{ \underbrace{0, \dots, 0}_{p+1}, \frac{l_1 + l_2}{L}, \frac{l_1 + l_2 + l_3}{L}, \dots, \frac{\sum_{i=1}^{n-2} l_i}{L}, \underbrace{1, \dots, 1}_{p+1} \right\} \tag{4}$$

where l_i is the straight-line distance between the $(i - 1)$ -th control point and i -th control point, and L is the sum of l_1, l_2, \dots, l_n . Moreover, when $\{w_iP_i\}$ and $\{w_iQ\}$ are represented as the control points, two B-spline curves $A(u), B(u)$ can be defined as:

$$A(u) = \sum_{i=0}^n N_{i,p}(u)w_iP_i \tag{5}$$

$$B(u) = \sum_{i=0}^n N_{i,p}(u)w_iQ \tag{6}$$

where Q is a constructed vector. It has the same dimension as the P_i , and all elements of Q are 1. Then, the NURBS curve Eq. (1) can be rewritten as:

$$C(u) = \frac{A(u)}{B_{st}(u)} \tag{7}$$

where $B_{st}(u)$ represents the first term of vector $B(u)$.

2.2 Curvature of a NURBS curve

For a NURBS curve of the p -th degree, it has derivatives of the d -th order where $d \leq p$. According to Eq. (7), the first- and second-order derivatives of a NURBS curve, i.e., $C'(u)$ and $C''(u)$, can be evaluated as follows:

$$C'(u) = \frac{A'(u) - C(u)B'_{st}(u)}{B_{st}(u)} \tag{8}$$

$$C''(u) = \frac{A''(u) - 2C'(u)B'_{st}(u) - C(u)B''_{st}(u)}{B_{st}(u)} \tag{9}$$

As shown in Eqs. (5) and (6), $A(u)$ and $B(u)$ can be treated as two different B-spline curves. Assuming that $H_i = w_i P_i$, the k -order derivatives of B-spline curve $A(u)$ can be written as:

$$A^{<k>}(u) = \sum_{i=0}^{n-k} N_{i,p-k}(u) H_i^{<k>} \tag{10}$$

$$H_i^{<k>} = \begin{cases} H_i, & k = 0 \\ \frac{p-k+1}{u_{i+p+1}-u_{i+k}} (H_{i+1}^{<k-1>} - H_i^{<k-1>}), & k > 0 \end{cases} \tag{11}$$

where k is the order of derivatives, and $U^{<k>} = \{ \underbrace{0, \dots, 0}_{p-k+1}, u_{p+1}, \dots, u_{m-p-1}, \underbrace{1, \dots, 1}_{p-k+1} \}$.

Similarly, $B^{<k>}(u)$ can be obtained. Therefore, $C'(u)$ and $C''(u)$ can be calculated based on $A^{<k>}(u)$ and $B^{<k>}(u)$. Then, the curvature of a NURBS curve is calculated, as shown in Eq. (12).

$$\rho = \frac{\|C'(u) \times C''(u)\|}{\|C'(u)\|^3} \tag{12}$$

2.3 Arc length of a NURBS curve

The arc length $s(u)$ of a NURBS curve $C(u)$ over parameter interval $[u_i, u_{i+1}]$ is defined as:

$$s(u) = \int_{u_i}^{u_{i+1}} \|C'(u)\| du \tag{13}$$

where $\|C'(u)\|$ is the norm of the first-order derivative $C'(u)$.

In this study, a numerical integration method, i.e., the Simpson rule, is adopted to approximately estimate the arc length of a NURBS curve. $S(u_i, u_{i+1})$ represents the approximate arc length of the NURBS curve $C(u)$ on the interval $[u_i, u_{i+1}]$, and it can be calculated by:

$$s(u_i, u_{i+1}) = \frac{(u_{i+1} - u_i)}{6} \left[s'(u_i) + 4s'(\frac{u_{i+1} + u_i}{2}) + s'(u_{i+1}) \right] \tag{14}$$

However, the direct calculation of the Simpson rule will cause approximation error in the numerical result. The issue can be addressed by bisecting the parameter interval $[u_i, u_{i+1}]$ into two equal sub-intervals, which is called the composite Simpson rule. Then, Eq. (14) is repeatedly applied over each subinterval:

$$s(u_i, \frac{u_i + u_{i+1}}{2}, u_{i+1}) = s(u_i, \frac{u_i + u_{i+1}}{2}) + s(\frac{u_i + u_{i+1}}{2}, u_{i+1}) \tag{15}$$

Assuming that δ is the tolerance for the parameter interval, we have:

$$\left| s(u_i, \frac{u_i + u_{i+1}}{2}, u_{i+1}) - s(u_i, u_{i+1}) \right| / 10 < \delta \tag{16}$$

If the condition in Eq. (16) cannot be satisfied, the two subintervals in Eq. (15) will be divided into two subintervals again. The loop repeats until Eq. (16) holds for each subinterval. Finally, the sum of the arc lengths of all subintervals is the real length of the arc. In this study, $f_s(a, b)$ is defined as the iteratively calculated arc length between a and b .

2.4 NURBS parameter and curve interpolation

The second-order Taylor’s expansion is widely used to establish the relationship between the NURBS parameter and the arc length. It is shown as follows:

$$u_{i+1} = u_i + \left. \frac{du}{dt} \right|_{t=t_i} \cdot T_s + \left. \frac{d^2u}{dt^2} \right|_{t=t_i} \cdot \frac{T_s^2}{2} \tag{17}$$

where u_i is the curve parameter when $t = t_i$, and T_s is the sampling period. After derivation, it can be rewritten as follows:

$$u_{i+1} = u_i + \frac{V(t_i)}{|C'(u_i)|} \cdot T_s + \frac{1}{|C'(u_i)|} \left(A(t_i) - \frac{C'(u_i) \cdot C''(u_i)}{|C'(u_i)|^3} V^2(t_i) \right) \cdot \frac{T_s^2}{2} \tag{18}$$

where $V(t)$ and $A(t)$ are the desired velocity and acceleration of the NURBS curve, respectively.

3 S-shaped feedrate scheduling based NURBS interpolator

In this section, the division of the curve into several blocks by using the curvatures is introduced in Sec 3.1. Then, the S-shaped feedrate profile is introduced briefly in Sect. 3.2. In

Sect. 3.3, the precise calculation of the arc length of blocks and NURBS parameter is described.

3.1 Critical points and curve segmentation

In NURBS curves, the sharp corners with large curvature can undermine the dynamic performance of high-speed machining. Thus, it is critical to segment each block at the key points that are commonly referred to as sharp corners. The feedrate velocity over the candidate points can be calculated based on the two constraints of chord error and centripetal acceleration [19].

Constrained by the chord error, the feedrate should satisfy Eq. (19):

$$v_i \leq \frac{2}{T_s} \sqrt{\frac{1}{\rho_i^2} - \left(\frac{1}{\rho_i} - \delta_{\max}\right)^2} \tag{19}$$

where ρ_i is the curvature at the i -th interpolation point; δ_{\max} is the maximum chord error, and T_s is the sampling period.

Constrained by the centripetal acceleration, the feedrate should also satisfy Eq. (20):

$$v_i \leq \sqrt{\frac{A_{c\max}}{\rho_i}} \tag{20}$$

where $A_{c\max}$ is the maximum centripetal acceleration, so the restricted feedrate v_i^R can be expressed as follows:

$$v_i^R = \min\left\{\frac{2}{T_s} \sqrt{\frac{1}{\rho_i^2} - \left(\frac{1}{\rho_i} - \delta_{\max}\right)^2}, \sqrt{\frac{A_{c\max}}{\rho_i}}, V_{\max}\right\} \tag{21}$$

where V_{\max} is the allowable max feedrate of the actuator.

Ultimately, the critical points are found from the candidate points, where v_i^R is less than the feedrate command. Based on these critical points, the NURBS curve can be split into several continuous blocks.

3.2 S-shape velocity profile and bidirectional scanning

3.2.1 S-shaped feedrate profile

The S-shaped feedrate profile consisting of seven phases is shown in Fig. 1. The expressions of S-shaped velocity, acceleration, and jerk in seven phases are summarized in Table 1. The symbols $t_i (i = 1, 2, \dots, 7)$ represent the time parameters. where v_s and v_e are the starting and ending velocities,

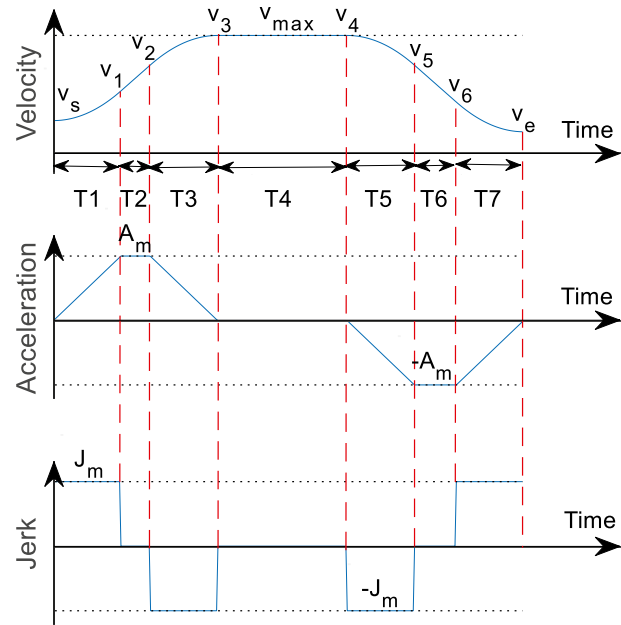


Fig. 1 The S-shaped feedrate profile with seven sections

respectively; A_m is the allowable maximum acceleration, and J_m is the maximum jerk of the actuator.

As the change of target displacement, initial velocity, end velocity, maximum acceleration, and jerk, there are 17 types of the S-shape feedrate profile. It is critical to formulate an S-shape feedrate profile basic function that can deal with all 17 cases. The velocity and acceleration profiles of the 17 cases are provided in the Appendix. A more detailed formula expression of the 17 types S-shape feedrate profile is given by Sencer et al. in [4].

3.2.2 Bidirectional velocity scanning

If the target displacement is too short but the velocity calculated by Eq. (21) is big, the system cannot reach the

Table 1 Expressions for the S-shaped feedrate profile

t	$J(t)$	$a(t)$	$v(t)$
t_1	J_m	$J_m t_1$	$v_s + \frac{1}{2} J_m t_1^2$
t_2	0	A_m	$v_1 + A_m t_2$
t_3	$-J_m$	$A_m - J_m t_3$	$v_2 + A_m t_3 - \frac{1}{2} J_m t_3^2$
t_4	0	0	v_3
t_5	$-J_m$	$-J_m t_5$	$v_4 - \frac{1}{2} J_m t_5^2$
t_6	0	$-A_m$	$v_5 - A_m t_6$
t_7	J_m	$-A_m + J_m t_7$	$v_6 - A_m t_7 + \frac{1}{2} J_m t_7^2$

ending velocity from the starting velocity even by accelerating or decelerating with the max acceleration and jerk. Besides an S-shape feedrate profile basic function that can deal with all 17 cases, a bidirectional velocity scanning scheme (forward velocity and backward velocity scanning) is also needed to correct the starting or ending velocity in these cases.

Forward velocity scanning In this case, $v_s < v_e$, where v_s and v_e are the starting and ending velocities, respectively. According to the 7-segment S-shape velocity profile, the minimum acceleration time t_a and displacement s_a required for the velocity from v_s to v_e in the pure acceleration stage can be expressed as:

$$\begin{cases} t_a = \sqrt{\frac{v_e - v_s}{J}} \\ s_a = t_a(v_e + v_s) \end{cases} \quad (22)$$

Assume that s is the target displacement. If $s < s_a$, then the velocity cannot reach v_e within this displacement, and the ending velocity needs to be revised. As shown in Eq. (23), v'_{max} is the maximum velocity that can be reached within the displacement s .

$$(v'_{max} + v_s) \cdot \sqrt{\frac{v'_{max} - v_s}{J}} - s = 0 \quad (23)$$

Backward velocity scanning In this case, $v_s > v_e$. Similarly, the minimum deceleration time t_d and displacement s_d required for the velocity from v_s to v_e in the pure deceleration stage can be expressed as:

$$\begin{cases} t_d = \sqrt{\frac{v_s - v_e}{J}} \\ s_d = t_d(v_s + v_e) \end{cases} \quad (24)$$

If $s < s_d$, the velocity cannot reach v_e within this displacement, and the starting velocity needs to be revised. As shown in Eq. (25), v'_{max} is the maximum velocity that can be decelerated to v_e within the displacement s .

$$(v'_{max} + v_e) \cdot \sqrt{\frac{v'_{max} - v_e}{J}} - s = 0 \quad (25)$$

Figure 2 illustrates the examples of forward and backward velocity scanning, where the red curves and blue curves

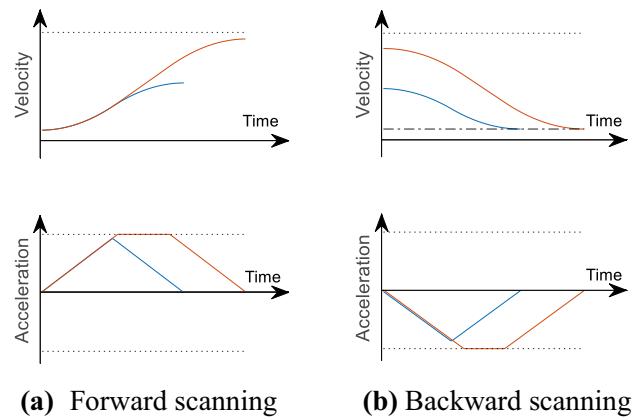


Fig. 2 Bidirectional velocity scanning. **a** Forward scanning. **b** Backward scanning

indicate the original planning trajectory and the updated after velocity scanning, respectively.

3.3 The high-precision NURBS interpolator

The Simpson bisection iterative algorithm can obtain the accurate arc length of each block after multiple iterations. Unfortunately, due to the truncation error of the Taylor expansion, the calculated parameter u usually cannot be accurately planned to 1. Therefore, the actuator cannot completely track the NURBS curve. To handle this problem, this paper proposes an iteration algorithm to calculate the block displacement with NURBS parameter u as the optimization target. The algorithm is as follows:

Assume that the number of breakpoints is n . Then, the number of blocks is $(n + 1)$, and the arc length of blocks is s_0, s_1, \dots, s_n . Denote the total arc length of the NURBS curve as S . S_H and S_L satisfy the inequality $S_H > S > S_L$, and $\Delta S = S_H - S_L$.

Firstly, the Simpson bisection iterative algorithm is used to calculate the arc length of each block. The velocities and accelerations of blocks can be calculated using the S-shape feedrate profile. Then, with these velocities and accelerations, the NURBS parameter u can be calculated by Eq. (18). Usually, the NURBS parameter u cannot meet the precision requirement due to the truncation error. Thus, S_H and S_L are modified by a dichotomy searching, and the arc length of each block is modified according to Eq. (26). The loop will

not break until the NURBS parameter u satisfies the precision requirement.

$$s_i = s_i + \Delta S \cdot f_s(u_{i-1}, u_i) / S_L \quad (26)$$

The whole algorithm is described in Algorithm 1: where Δl is a small offset, and eps is the accuracy tolerance. They are set to 5mm and 0.000005 in this paper, respectively.

Algorithm 1:

Data: $S, S_H, S_L, s_0, s_1, \dots, s_n, u;$

Initialize: $S = 0.0; \quad u = 0.0;$

$s_0 = f_s(0.0, u_0);$

$s_i = f_s(u_{i-1}, u_i);$

$s_n = f_s(u_{n-1}, 1.0);$

$S_H = f_s(0.0, 1.0) + \Delta l, \quad S_L = f_s(0.0, 1.0) - \Delta l;$

while $|u - 1.0| > eps$

$S = (S_H + S_L) / 2;$

$\Delta S = S - S_L;$

$s_0 = s_0 + \Delta S \cdot f_s(0.0, u_0) / f_s(0.0, 1.0);$

$s_i = s_i + \Delta S \cdot f_s(u_{i-1}, u_i) / f_s(0.0, 1.0);$

$s_n = s_n + \Delta S \cdot f_s(u_{n-1}, 1.0) / f_s(0.0, 1.0);$

for $i = 0$ to n **then**

Use the $s_i, v_s^i,$ and v_e^i to complete S-shape feedrate profile;

Using the velocities and accelerations of each block calculated by S-shape feedrate profile, the vector of NURBS parameter u can be calculated by Eq. 18;

end

if $u \geq 1.0$ **then**

$S_H = (S_H + S_L) / 2.0$

else if $u < 1.0$ **then**

$S_L = (S_H + S_L) / 2.0$

end

end

The degree: $p = 3.$

The control points (mm): $P = [(533.2, 52.0), (534.3, 52.0), (534.8, 49.4), (535.5, 44.8), (548.3, 51.2), (556.5, 58.4), (569.3, 66.9), (584.7, 63.6), (579.2, 47.1), (573.3, 39.7), (571.1, 30.3), (562.2, 33.6), (570.6, 28.3), (568.2, 20.2), (562.0, 15.3), (566.4, 4.6), (559.7, 9.1), (558.6, 14.3), (554.8, 8.3), (548.9, 12.4), (542.9, 16.7), (538.7, 21.9), (534.4, 36.2), (535.7, 24.8), (538.5, 19.6), (533.2, 14.8), (528.0, 19.6), (530.8, 24.8), (532.1, 36.2), (527.7, 21.9), (523.6, 16.7), (517.6, 12.4), (511.7, 8.3), (507.9, 14.3), (506.8, 9.12), (500.1, 4.69), (504.5, 15.3), (498.3, 20.2), (495.8, 28.3), (504.3, 33.6), (495.4, 30.3), (493.0, 39.6), (487.4, 47.2), (481.8, 63.6), (497.2, 66.9), (509.9, 58.4), (518.2, 51.2), (531.0, 44.8), (531.7, 49.4), (532.2, 52.0), (533.2, 52.0)].$

The knot vector: $U = [0, 0, 0, 0, 0.008161, 0.018293, 0.049457, 0.073221, 0.106681, 0.140978, 0.178852, 0.199462, 0.220485, 0.241155, 0.262784, 0.281181, 0.298390, 0.323584, 0.341160, 0.352734, 0.368200, 0.383845, 0.399920, 0.414476, 0.446994, 0.471980, 0.484841, 0.500412, 0.515823, 0.528684, 0.553670, 0.586251, 0.600671, 0.616746, 0.632392, 0.647858, 0.659389, 0.676880, 0.701893, 0.719102, 0.737562, 0.759375, 0.780046, 0.800961, 0.821519, 0.859257, 0.893554, 0.926833, 0.950760, 0.981924, 0.992057, 1.0, 1.0, 1.0, 1.0].$

The weight vector: $W = [1.0000, 1.0000, 1.0000, 1.2000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 2.0000, 1.0000, 1.0000, 5.0000, 3.0000, 1.0000, 1.1000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.1000, 1.0000, 3.0000, 5.0000, 1.0000, 1.0000, 2.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.0000, 1.2000, 1.0000, 1.0000, 1.0000].$

To describe the test curve as accurately as possible, the curve including 100,001 sampling points with a sampling spacing $\Delta u = 0.00001$ is shown in Fig. 3a. The curvature curve corresponding to the sampling points is shown in Fig. 3b. As shown in Fig. 3b, the curvature threshold ε is set as to 0.25, and 19 curvature extreme points can be selected. According to the constraints of chord error and centripetal acceleration, the critical points are found from the sampling points, and they are also shown in Fig. 3a as black dots. The maximum velocity at each critical point can be determined by Eq. (21).

The interpolator parameters used for the butterfly-shaped test curve are listed in Table 2.

The Simpson iterative algorithm and the method proposed in this paper are used to calculate the arc length of each block. The calculation results obtained by the Simpson iterative algorithm are as follows: $s_0 = 0.585141,$

4 Simulation and experimental results

4.1 Simulation results and analysis

In this study, the butterfly-shaped curve is taken as our NURBS test curve. The degree, control points, knot vectors, and weight vectors of the NURBS curve are given as follows:

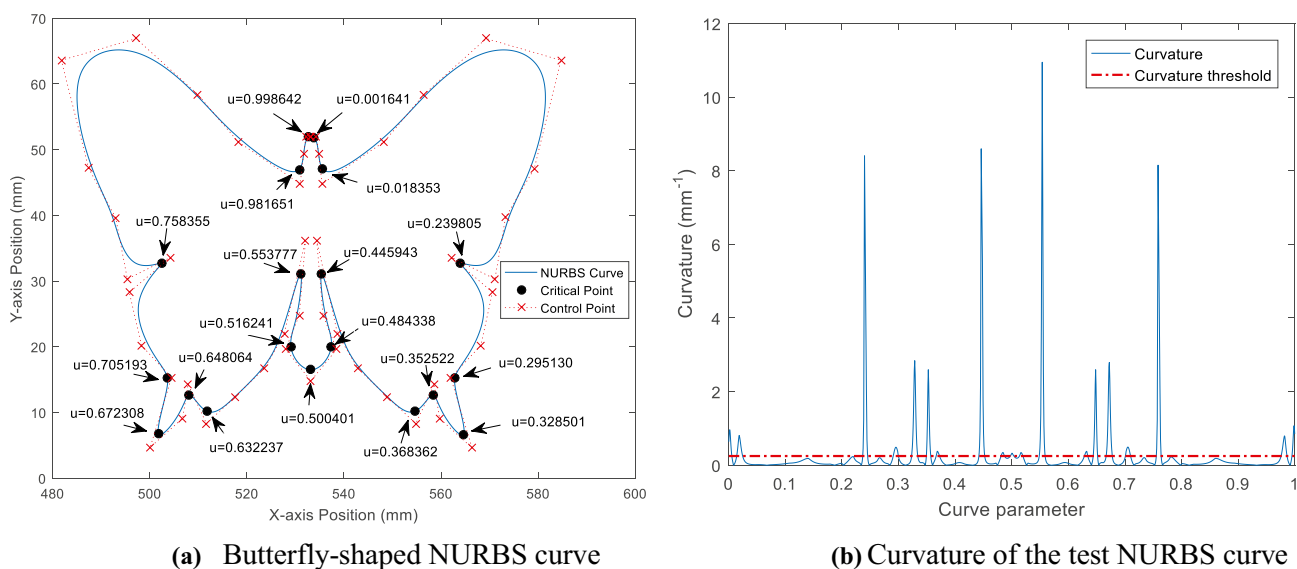


Fig. 3 The butterfly-shaped curve, critical points, and curvature. **a** Butterfly-shaped NURBS curve. **b** Curvature of the test NURBS curve

$s_1 = 5.348356, s_2 = 89.180639, s_3 = 21.120720, s_4 = 8.898868, s_5 = 8.896570, s_6 = 4.699582, s_7 = 29.888507, s_8 = 11.498431, s_9 = 5.799827, s_{10} = 5.699825, s_{11} = 11.487612, s_{12} = 29.917852, s_{13} = 4.698810, s_{14} = 8.897285, s_{15} = 8.798869, s_{16} = 21.084886, s_{17} = 89.210446, s_{18} = 5.431492, s_{19} = 0.467329$. The total arc length of the NURBS curve is $S = 371.611046$. The calculation results obtained by the method proposed in this paper are as follows: $s_0 = 0.585740, s_1 = 5.353978, s_2 = 89.243448, s_3 = 21.142144, s_4 = 8.908179, s_5 = 8.907302, s_6 = 4.704525, s_7 = 29.912613, s_8 = 11.511162, s_9 = 5.805811, s_{10} = 5.705704, s_{11} = 11.500059, s_{12} = 29.942049, s_{13} = 4.703753, s_{14} = 8.908096, s_{15} = 8.808009, s_{16} = 21.106301, s_{17} = 89.273012, s_{18} = 5.437201, s_{19} = 0.467807$. The total arc length is $S = 371.991906$.

Table 2 Test parameters of the interpolations

Parameters	Symbols	Units
Interpolator period	T_s	1ms
Chord error tolerance	δ_{max}	0.0005mm
Maximum tangential acceleration	A_{tmax}	2000mm ² /s
Maximum centripetal acceleration	A_{cmax}	3000mm ² /s
Maximum jerk	J_{max}	20000mm ³ /s
Maximum velocity	V_{max}	100mm/s
Arc-length tolerance	ϵ_l	0.000001

It can be found that for compensating the truncation error, the arc length of each block calculated by the proposed method in this paper increases little compared with that calculated by the Simpson iterative algorithm. The arc length of the first and last block is all too short for the S-shaped feedrate profile, so this study corrects the velocity of these junctions through the bidirectional velocity scanning scheme. Figure 4 shows the velocity, acceleration, and jerk curves of the S-shaped feedrate profile calculated by our method. The ending velocities obtained by the Simpson iteration method and our method are 0.0043165 mm/s and 0.00042 mm/s, respectively. Obviously, our proposed method achieves better accuracy. The NURBS parameter u of the two methods can be calculated by Eq. (18). As shown in Fig. 5a, the parameter u obtained by the two methods is basically the same, i.e., 0.998752 and 0.999993.

As shown in Fig. 6, the method based on the Simpson iteration algorithm leaves a small blank space at the end of the NURBS curve that cannot be executed. By contrast, the method proposed in this study obtains the NURBS curve precisely. Figure 5b shows the iterative process of calculating the total arc length of the NURBS curve using the method proposed in this paper. It is found that the ideal situation is reached immediately after 10 iterations, and the whole process takes less than 2.5 s. It indicates that the proposed algorithm improves the NURBS interpolation precision for offline trajectory planning.

Fig. 4 NURBS parameter u and the arc length

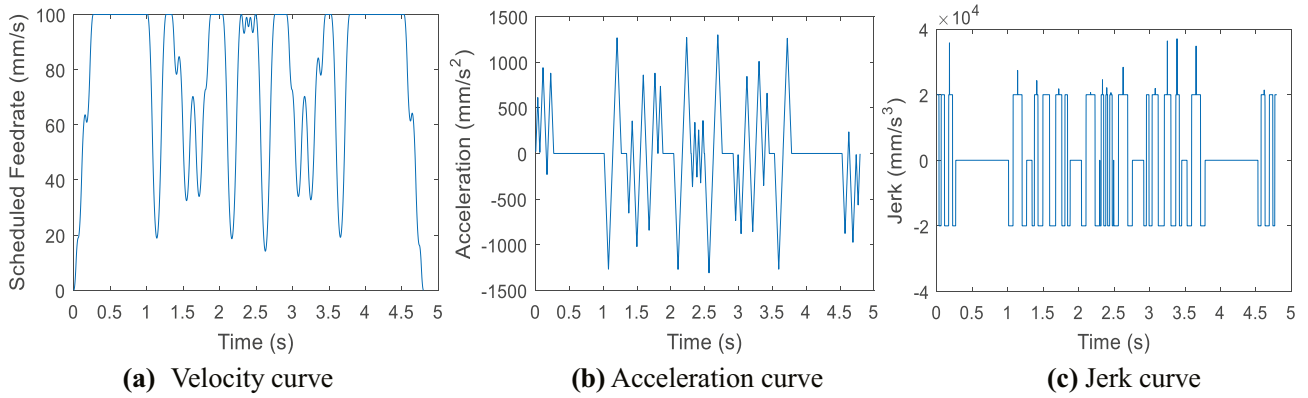
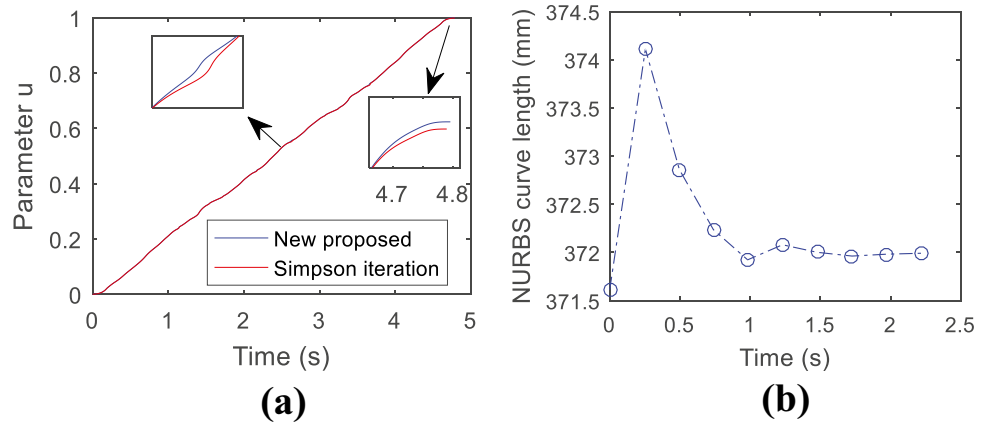


Fig. 5 The scheduled curves of the method proposed in this paper. **a** Velocity curve. **b** Acceleration curve. **c** Jerk curve

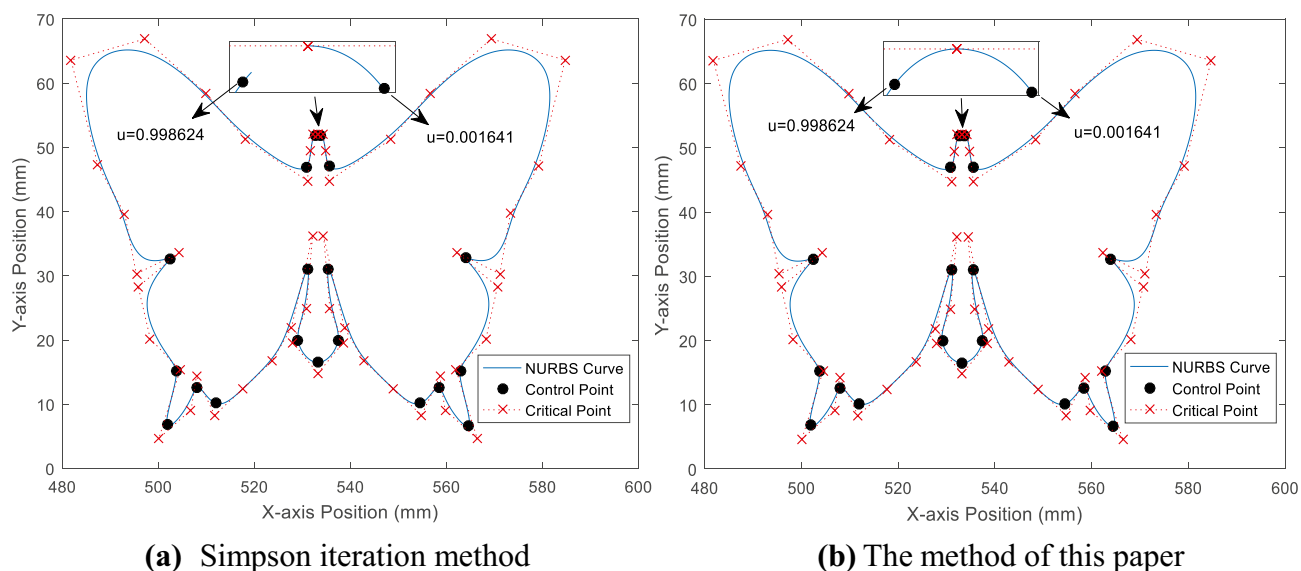


Fig. 6 Comparison of NURBS curves. a Simpson iteration method. b The method of this paper

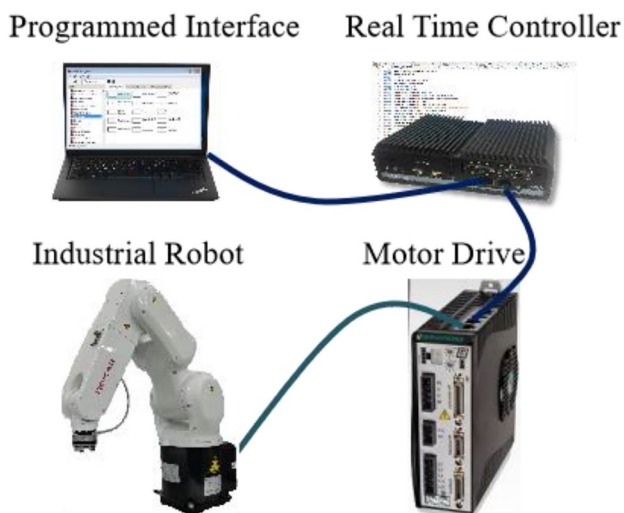


Fig. 7 The Experiment platform

4.2 Experimental results

Corresponding to the simulation tests, the butterfly-shaped curve is drawn by a real industrial robot. As shown in Fig. 7, the experiment was implemented on a 6-DOF ROKAE (XB7) light-weight industrial robot, and a carbon pen was installed on the wrist of the robot. Meanwhile, the existing commercial motor drive of the ROKAE was utilized to control the joint position. The task-level and intermediate-layer computation was run on a 64-Bit Linux-based real-time controller equipped with an Intel Core-i7 CPU with 8 GB memory. The controller communicates with the motor drive via the EtherCAT protocol at a sampling rate of 1 kHz.

The butterfly-shaped NURBS curve drawn by the 6-DOF industrial robot is shown in Fig. 8. As shown in the video of the attachment, the robot runs smoothly and stably throughout the execution process.

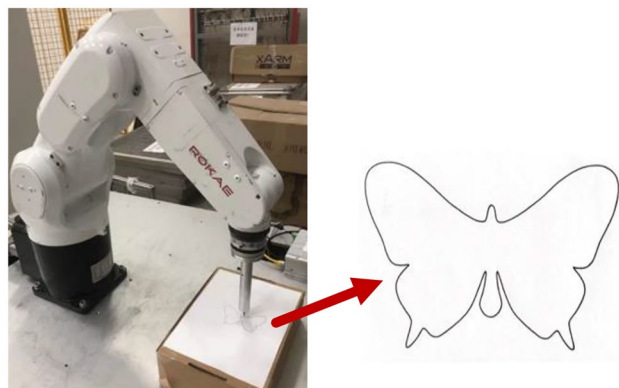
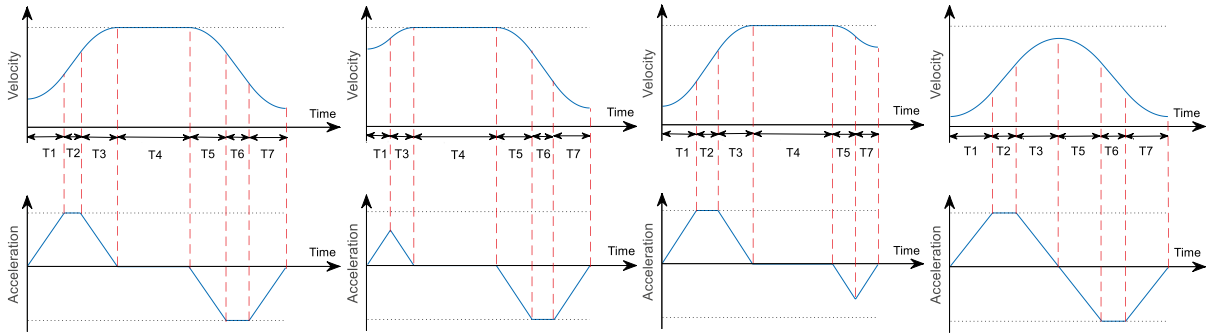


Fig. 8 Experimental result of butterfly-shaped curve

5 Conclusion

This paper proposes a high-precision S-shaped feedrate scheduling scheme for the NURBS interpolator. It consists of curve segmentation, bidirectional velocity scanning, and NURBS curve interpolation. Meanwhile, an interpolation method is proposed based on Taylor’s expansion with arc length prediction and feedback correction. This method effectively improves the accuracy of NURBS curve planning.

Appendix. Seventeen cases of S-shaped velocity profile

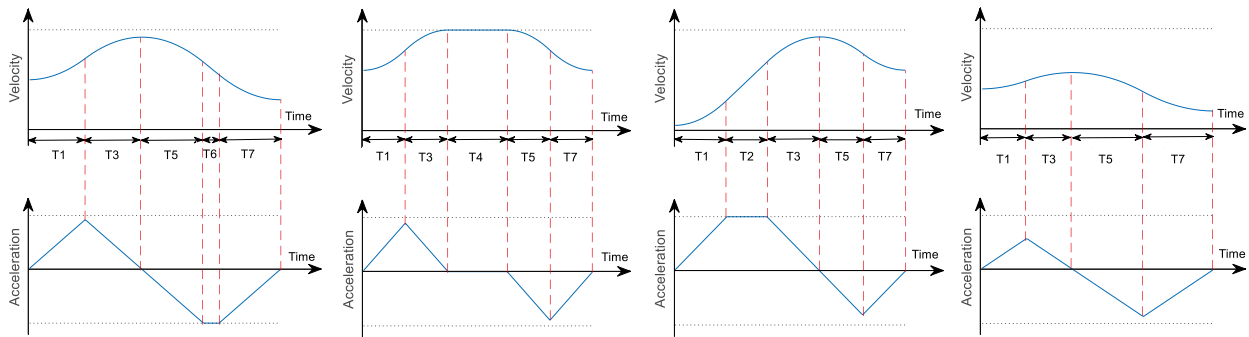


(a)

(b)

(c)

(d)

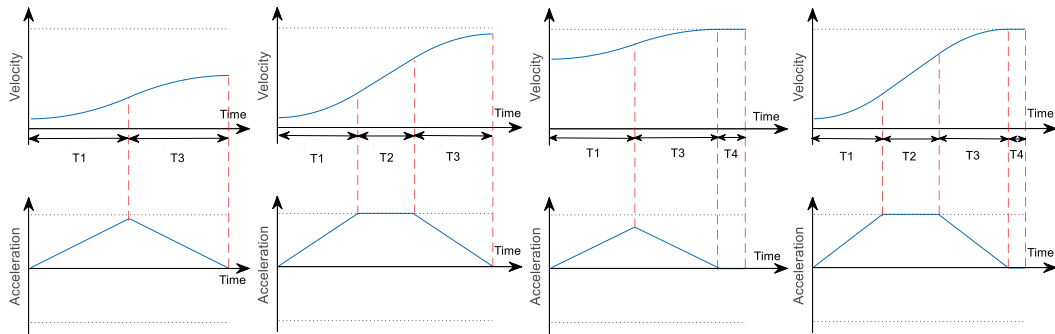


(e)

(f)

(g)

(h)

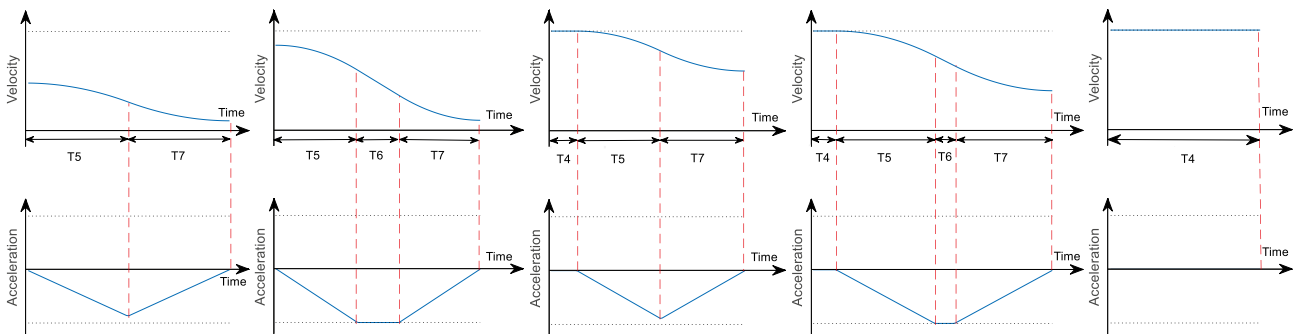


(i)

(j)

(k)

(l)



(m)

(n)

(o)

(p)

(q)

Author contribution Lijin Fang: conceptualization, methodology, writing—original draft. Guanghai Liu: software, data curation, validation, writing—review and editing. Qiang Li: supervision, writing—review and editing. Hualiang Zhang: supervision, writing—review and editing.

Funding This research was supported by the Central Leading Local Science and Technology Development Foundation of Liaoning Province under Grant No.2021 JH6/10500132, and in part by the “DEXMAN” project (Project number: 410916101) funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation).

Availability of data and material The datasets used or analyzed during the current study are available from the corresponding author on reasonable request.

Code availability The code during the current study is available from the corresponding author on reasonable request.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication All authors agree to publish the paper.

Competing interests The authors declare no competing interests.

References

- Chen C, Chen S (2019) Synchronization of tool tip trajectory and attitude based on the surface characteristics of workpiece for 6-DOF robot manipulator. *Robot Comput Integr Manuf* 59:13–27
- Ji S, Lei L, Zhao J, Lu X, Gao H (2021) An adaptive real-time NURBS curve interpolation for 4-axis polishing machine tool. *Robot Comput Integr Manuf* 67:102015
- Qiao Z, Liu Z, Hu M, Liu L, Hu W, Ding Y (2019) A space cutter compensation method for multi-axis machining using triple NURBS trajectory. *Int J Adv Manuf Technol* 103:3969–3978
- Sencer B, Altintas Y, Croft E (2008) Feed optimization for five-axis CNC machine tools with drive constraints. *Int J Mach Tools Manuf* 48:733–745
- Beudaert X, Pechard P, Tournier C (2011) 5-Axis tool path smoothing based on drive constraints. *Int J Mach Tools Manuf* 51:958–965
- Hashemian A, Bo P, Barton M (2020) Reparameterization of ruled surfaces: toward generating smooth jerk-minimized toolpaths for multi-axis flank CNC milling. *Comput Aided Des* 127:102868–
- Lin M, Tsai M, Yau H (2007) Development of a dynamics-based NURBS interpolator with real-time look-ahead algorithm. *Int J Mach Tools Manuf* 47:46–62
- Du X, Huang J, Zhu L (2015) A complete S-shape feed rate scheduling approach for NURBS interpolator. *J Comput Des Eng* 2(4):206–217
- Ni H, Zhang C, Ji Shuai HuT, Chen Q, Liu Y (2017) A bi-directional adaptive feedrate scheduling method of NURBS interpolation based on S-shaped ACC/DEC algorithm. *IEEE Access* 6:63794–63812
- Ji S, Hu T, Huang Z, Zhang C (2020) A NURBS curve interpolator with small feed rate fluctuation based on arc length prediction and correction. *Int J Adv Manuf Technol* 111:2095–2104
- Liu M, Huang Y, Yin L, Guo J, Shao X, Zhang G (2014) Development and implementation of a NURBS interpolator with smooth feedrate scheduling for CNC machine tools. *Int J Mach Tools Manuf* 87:1–15
- Heng M, Erkorkmaz K (2010) Design of a NURBS interpolator with minimal feed fluctuation and continuous feed modulation capability. *Int J Mach Tools Manuf* 50(3):281–293
- Liu H, Liu Q, Zhou S, Li C, Yuan S (2015) A NURBS interpolation method with minimal feedrate fluctuation for CNC machine tools. *Int J Adv Manuf Technol* 78:1241–1250
- Lo CC (1997) Feedback Interpolators for CNC Machine Tools. *J Manuf Sci Eng* 119:587–592
- Tsai M-C, Cheng C-W (2003) A real-time predictor-corrector interpolator for CNC machining. *J Manuf Sci Eng* 125(3):449–460
- Zhao H, Zhu LM, Han D (2013) A parametric interpolator with minimal feed fluctuation for CNC machine tools using arc-length compensation and feedback correction. *Int J Mach Tool Manuf* 75:1–8
- Wang T, Zhang Y, Dong J, Ke R, Ding Y (2019) NURBS interpolator with adaptive smooth feedrate scheduling and minimal feedrate fluctuation. *Int J Precis Eng Manuf* 21(2):273–290
- Gordon W, Riesenfeld R (1974) B-spline curves and surfaces[J]. *Comput Aided Geom Des* 23(91):95–126
- Du D, Liu Y, Guo X, Yamazaki K, Fujishima M (2010) An accurate adaptive NURBS curve interpolator with real-time flexible acceleration/deceleration control. *Robot Comput Integr Manuf* 26:273–281

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.