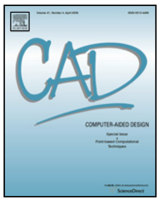
Computer-Aided Design 43 (2011) 612–628

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad



# The feedrate scheduling of NURBS interpolator for CNC machine tools

An-Chen Lee[a], Ming-Tzong Lin[b,*], Yi-Ren Pan[a], Wen-Yu Lin[a]

[a] Department of Mechanical Engineering, National Chiao Tung University, Hsinchu 30010, Taiwan, ROC

[b] Department of Mechanical Design Engineering, National Formosa University, Yunlin 63201, Taiwan, ROC

ARTICLE INFO

*Article history:*
Received 18 April 2010
Accepted 26 February 2011

*Keywords:*
NURBS
Feedrate scheduling
Curve splitting
Off-line scanning

ABSTRACT

This paper proposes an off-line feedrate scheduling method of CNC machines constrained by chord tolerance, acceleration and jerk limitations. The off-line process for curve scanning and feedrate scheduling is realized as a pre-processor, which releases the computational burden in real-time task. The proposed method first scans a non-uniform rational B-spline (NURBS) curve and finds out the crucial points with large curvature (named as critical point) or $G^0$ continuity (named as breakpoint). Then, the NURBS curve is divided into several NURBS sub-curves using curve splitting method which guarantees the convergence of predictor–corrector interpolation (PCI) algorithm. The suitable feedrate at critical point is adjusted according to the limits of chord error, centripetal acceleration and jerk, and at breakpoint is adjusted based on the formulation of velocity variation. The feedrate profile corresponding to each NURBS block is constructed according to the block length and the given limits of acceleration and jerk. In addition, feedrate compensation method for short NURBS blocks is performed to make the jerk-limited feedrate profile more continuous and precise. Because the feedrate profile is established in off-line, the calculation of NURBS interpolation is extremely efficient for CNC high-speed machining. Finally, simulations and experiments with two free-form NURBS curves are conducted to verify the feasibility and applicability of the proposed method.

## 1. Introduction

In order to achieve high-speed and high-accuracy machining, many scholars devote to investigate the fields such as parametric curve interpolation, feedrate profile scheduling and servo-loop control techniques. Since 1950s, parametric curves like Bezier, *B*-spline and NURBS have been developed. Because of the benefits of NURBS [1], NURBS even becomes the standard format of free-form curve and surface in 1991. Recently, STEP compliant NC programming, STEP-NC has been specified as a new NC data model [2,3]. NURBS is adopted by STEP-NC and becomes the standard interface for data exchange between CAD/CAM and CNC systems. How to design a reliable and efficient NURBS interpolator is critical for developing the next generation intelligent CNC machine tool.

Shpitalni et al. [4] first proposed parametric curve interpolation and realized the parametric interpolator in CNC machine, which accepts parametric curve codes directly from CAD/CAM. For generating more precise motion trajectory, Yang and Kong [5] applied Taylor's expansion method to develop the first-order and second-order interpolation algorithms with constant feedrate. Nevertheless, because of high-order truncation errors [6], Taylor series interpolator might not generate accurate feedrate command along high-curvature tool paths for high-speed machining. Tsai and Cheng [7] proposed a closed-loop predictor–corrector interpolator (PCI) algorithm to replace Taylor's expansion method and provided the convergent condition of corrector. The advantage of PCI method is that the feedrate fluctuation can be controlled through setting tolerance of feedrate error for either given constant or variable feedrate command. Erkorkmaz and Altintas [8] proposed a quintic spline interpolation method to minimize feedrate fluctuation. This is done by either approximating the relation between the arc length and spline parameter using a feed correction polynomial or by solving the exact parameter using an iterative interpolation method. Otherwise, Lei et al. [9] proposed a fast NURBS interpolation method which generated inverse length functions (ILF) for each parameter subinterval in off-line. The new setting path parameter was calculated directly by using the ILF without the need for any time-consuming computation of NURBS derivatives and iterations in real time. However, most of the proposed methods attempt to maintain constant feedrate without considering chord error and acceleration/deceleration (ACC/DEC).

Yeh and Hsu [10] first proposed an adaptive-feedrate interpolator to adjust curve speed according to chord tolerance. Zhiming et al. [11] presented a curvature-based interpolation algorithm based on curvature of curves. Yang and Narayanaswami [12] proposed an off-line algorithm to detect feedrate sensitive corners and

* Corresponding author. Tel.: +886 5 631 5342; fax: +886 5 636 3010.
*E-mail addresses:* aclee@mail.nctu.edu.tw (A.-C. Lee), mtlin@nfu.edu.tw, ericfishxp@gmail.com (M.-T. Lin).

planned ACC/DEC accordingly so that chord errors are bounded. Sun et al. [13] developed a guide spline-based feedrate scheduling method for machining along curvilinear paths with constraints of chord error and ACC/DEC. However, the jerk might be out of limit for machine if the curvature of a curve changes abruptly.

To obtain a smooth jerk-limited feedrate profile with chord error constraint, many interpolation algorithms were developed in [14–17]. Lai et al. [17] proposed a more complete process to identify the segment points whose acceleration changed across zero, as the basis of feedrate scheduling. The algorithm embedded in look-ahead module can plan jerk-limited feedrate profile under various continuity conditions for composite curves. Nevertheless, most of the algorithms did not include dynamics effects of machine tool; thus the tracking or contouring error might not keep within desired accuracy for real machining. Liu et al. [15] considered machining dynamics by utilizing notching filtering or time spacing based on FFT analysis to eliminate the components containing high frequencies or frequencies matching machine natural ones in the interpolated acceleration profile. However, they did not provide the dynamics model of machine tool and consider the effects of servo dynamics. Lin et al. [18] and Tsai et al. [19] considered the effect of servo dynamics and appended servo dynamics model to look-ahead function. The experiment results demonstrate that the tracking and contouring performance were improved significantly. Dong and Stori [20] proposed a time-optimal algorithm based on dynamics of machine tool and capabilities of individual motion axis. A minimum-time feedrate profile subject to the constraints of velocity, acceleration, bandwidth and contouring error can be generated for a complex trajectory. Otherwise, Tikhon et al. [21] and Choi et al. [22] proposed the criterion of feedrate scheduling in accordance with material removal rate and surface roughness, respectively. Fleisig and Spence [23] extended spline curve interpolation algorithm from three-axis to five-axis machining. Mohan et al. [24] presented a review of various parametric interpolation methods for NURBS and discussed the salient features, problems and solutions. Recent approaches on variable feedrate interpolation, parameter compensation were also reviewed and research trends were addressed.

Feedrate scheduling is worthy of going deep into research since it is one of the most important factors for achieving the efficiency and quality of machining. Based on the architecture of CNC controller, feedrate scheduling can be performed in on-line or in off-line mode. In particular, real-time interpolation algorithms with look-ahead function are proposed in [15,17–19]. However, while complicated NURBS interpolation algorithms are performed in real time, the large number of backtracking process may cause time-consuming computations or the buffer could be used up for storing pre-interpolated data. Failure of the interpolator might incur tool chatter or breakage, even damage machine tool.

To solve the problems of CNC machining occurred in reality and to achieve the goal of high-speed and high-accuracy machining, this paper proposes an off-line feedrate scheduling method for CNC machines. The factors that affect machining precision such as chord error, feedrate fluctuation, and machine kinematics constraints are considered simultaneously. The method first finds out crucial points as the basis of feedrate scheduling and splits a NURBS curve at breakpoints into several NURBS sub-curves. Therefore, the divergence of PCI interpolation method can be avoided. Accordingly, the unique feedrate profile constrained by chord tolerance, acceleration and jerk limitations for a NURBS curve is constructed. In the real-time process, it only needs to perform PCI algorithm for updating path parameter and the Cox–de Boor algorithm for generating interpolation point. Taking the advantage of the proposed method, the complicated and heavy calculation of backtracking process in real time can be avoided while maintaining the desired precision within machine kinematics constraints.

## 2. NURBS curves and interpolation algorithms

A NURBS curve $C(u)$ can be expressed as follows [1]:

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

where $P_i$ is the control point, $w_i$ is the corresponding weight of $P_i$, $(n+1)$ is the number of control points, and $p$ is the degree of a NURBS curve. $N_{i,p}(u)$ is the $p$th-degree $B$-spline basis function defined on the non-uniform knot vector $U = \{u_0, u_1, \ldots, u_{n+p+1}\}$. The $p$th-degree $B$-spline basis function is recursively defined as follows

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

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

For generating a motion trajectory of parametric curve $C(u)$, the first step is to determine the curve parameter $u$. Taylor series expansion method is adopted in most NURBS interpolation algorithms. By employing Taylor's expansions of $u(t)$ at $t = t_i$ and neglecting high-order terms, the second-order Taylor interpolation algorithm is given as [6]:

$$u_{i+1} = u_i + \frac{V(u_i) \cdot T_s}{|C'(u_i)|} + \frac{1}{|C'(u_i)|}\left(A(u_i) - \frac{C'(u_i) \cdot C''(u_i)}{|C'(u_i)|^3} V(u_i)^2\right)\frac{T_s^2}{2} \tag{4}$$

where $V(u_i)$, $A(u_i)$, $T_s$, $C'(u_i)$ and $C''(u_i)$ are the feedrate, acceleration, sampling time, first and second derivatives of a NURBS curve, respectively.

For further reducing feedrate fluctuation, the PCI method [7] is chosen to update the parameter $u$ in this paper. In the predictor stage, the interpolation command at the next sampling time is estimated using the equation:

$$u_{i+1} = 3u_i - 3u_{i-1} + u_{i-2}. \tag{5}$$

In the correct stage, the following equations are utilized to repeatedly update $u_{i+1}$ within sampling period until the specified feedrate accuracy is satisfied.

$$u_{i+1}^{(j)} = -\alpha(u_i - u_{i+1}^{(j-1)}) + u_i \quad \forall j \ge 1 \tag{6}$$

$$\alpha = \frac{V_i^*}{-\beta(V_i^* - V_i^{(j-1)}) + V_i^*} \tag{7}$$

$$V_i^{(j-1)} = \frac{\left\| C(u_{i+1}^{(j-1)}) - C(u_i) \right\|}{T_s} \tag{8}$$

where $u_i$ is the value of parameter $u$ at time $t_i$, $u_{i+1}^{(j)}$ is the value of parameter $u$ after $j$ iteration step at time $t_{i+1}$, $\beta$ is the correctional coefficient, $V_i^*$ is the desired feedrate command at $u_i$ and $V_i^{(j-1)}$ is the current feedrate command computed at $u_i$ after $(j-1)$ iteration, respectively. The termination condition for the corrector is chosen as

$$\left| \frac{V_i^* - V_i^{(j-1)}}{V_i^*} \right| \le \varepsilon_{\text{PCI}} \tag{9}$$

where $\varepsilon_{\text{PCI}}$ is the tolerance of feedrate command error.

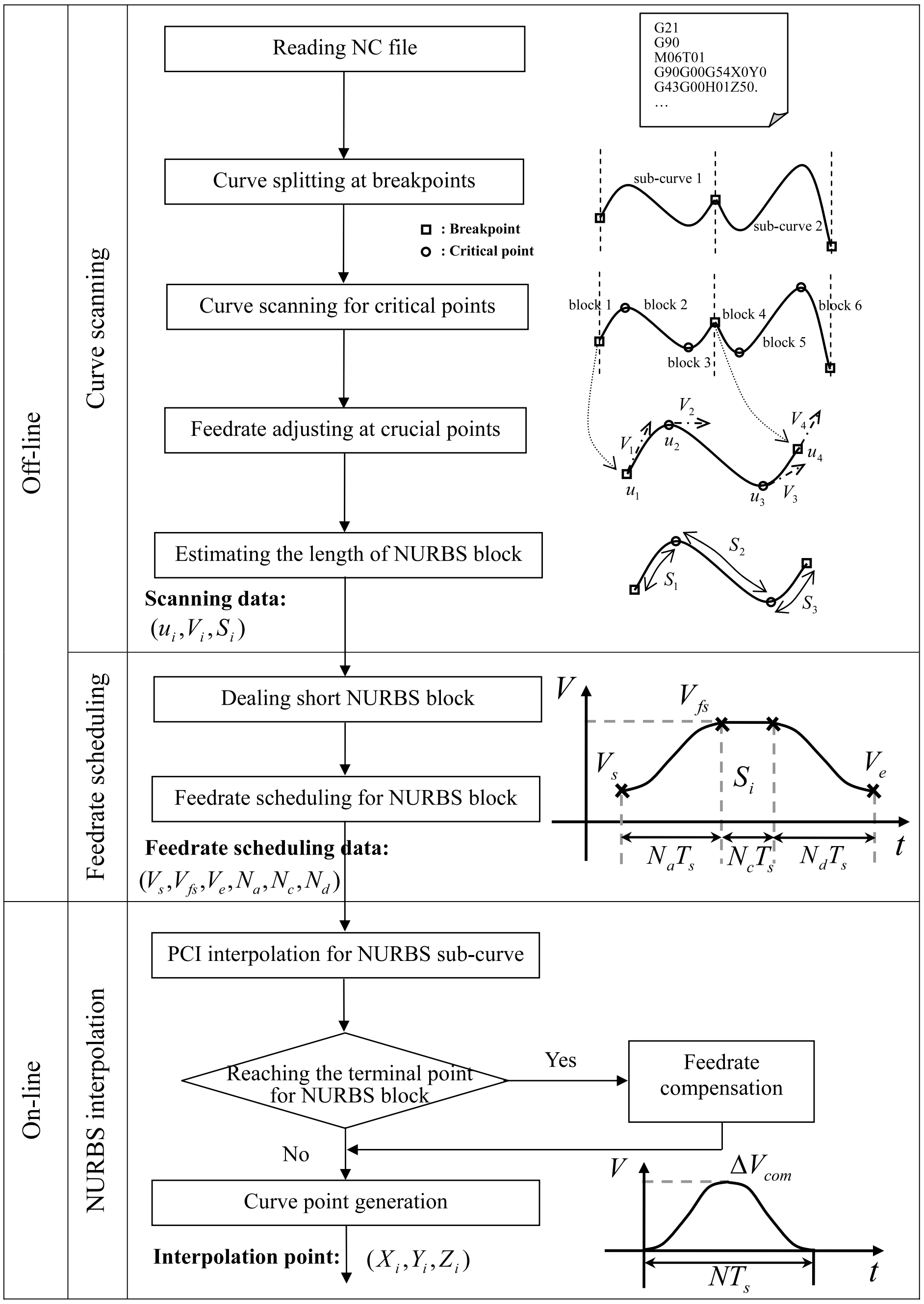

Off-line
Curve scanning
Reading NC file
G21
G90
M06T01
G90G00G54X0Y0
G43G00H01Z50.
…
Curve splitting at breakpoints
sub-curve 1
sub-curve 2
: Breakpoint
: Critical point
Curve scanning for critical points
block 1
block 2
block 3
block 4
block 5
block 6
Feedrate adjusting at crucial points
$V_1$ $V_2$ $V_3$ $V_4$ $u_1$ $u_2$ $u_3$ $u_4$
Estimating the length of NURBS block
$S_1$ $S_2$ $S_3$
Scanning data:
$(u_i, V_i, S_i)$
Feedrate scheduling
Dealing short NURBS block
Feedrate scheduling for NURBS block
Feedrate scheduling data:
$(V_s, V_{fs}, V_e, N_a, N_c, N_d)$
$V$ $V_{fs}$ $V_s$ $S_i$ $V_e$ $N_aT_s$ $N_cT_s$ $N_dT_s$ $t$
On-line
NURBS interpolation
PCI interpolation for NURBS sub-curve
Reaching the terminal point for NURBS block
Yes
No
Feedrate compensation
Curve point generation
Interpolation point:
$(X_i, Y_i, Z_i)$
$V$ $\Delta V_{com}$ $NT_s$ $t$

**Fig. 1.** The flowchart of feedrate scheduling method.

## 3. Off-line curve scanning

In order to plan the feedrate profile of any given NURBS curve, an off-line process for curve scanning and feedrate scheduling is developed as a pre-processor, which releases the computational burden in real-time task. The flowchart of the proposed feedrate scheduling method is shown in Fig. 1. In the stage of curve scanning, the breakpoints with $G^0$ continuity are detected by checking the multiplicity of knots in the knot vector of a NURBS curve, and the curve is split into several NURBS sub-curves. The adaptive-feedrate with curvature-based feedrate interpolation algorithm [18] is utilized to scan a NURBS curve for determining critical points with large curvatures. Furthermore, the suitable feedrate $V_i$ at each breakpoint is adjusted according to the limits of velocity variation on motion axes. The suitable feedrate $V_i$ at each critical point is evaluated with the constraints of chord error, centripetal acceleration and jerk limitations. Consequently, the NURBS curve is divided into small NURBS blocks after determining the breakpoints and critical points, which are called crucial points. The curve parameters $u_i$ of crucial points are recoded simultaneously. The length $S_i$ of each NURBS block between two adjacent crucial points is estimated by the adaptive Lobatto quadrature method [25]. Finally, the scanning data $(u_i, V_i, S_i)$ for each NURBS sub-curve and block are obtained and ready for the next stage of feedrate scheduling presented in Section 4.

### 3.1. Kinematic constraints

To generate smooth tool path for NURBS curves, jerk-limited feedrate profile should be planned in terms of acceleration and jerk limits. Here, $A_{\max}$ and $J_{\max}$ are denoted as acceleration and jerk limits on each axis for CNC machine, respectively. The limits

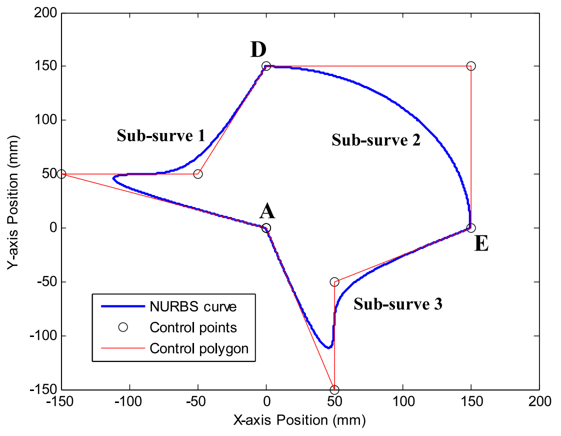


**Fig. 2.** The hat curve.

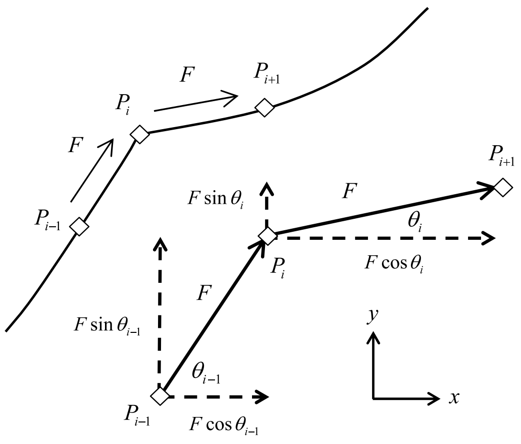


**Fig. 3.** Velocity variations of two axes across a breakpoint.

**Table 1**
Parameters of a hat curve.

| Parameters | Items |
|---|---|
| Control points:$P_{9\times 3}$ | (0, 0, 0); (−150, 50, 0); (−50, 50, 0); (0, 150, 0); (150, 150, 0); (150, 0, 0); (50, −50, 0); (50, −150, 0); (0, 0, 0) |
| Knot vector:$U_{1\times 12}$ | 0, 0, 0, 1/6, 1/3, 1/3, 2/3, 2/3, 5/6, 1, 1, 1 |
| Weights:$w_{1\times 9}$ | 1, 2, 3, 1, 0.7071, 1, 3, 2, 1 |
| Degree:$p$ | 2 |

of acceleration and jerk in the normal and tangent directions are set as

$$\begin{aligned} A_n &= A_t = A_{\max} \\ J_n &= J_t = J_{\max} \end{aligned} \tag{10}$$

where $A_n, A_t, J_n$ and $J_t$ are the centripetal acceleration, tangent acceleration, centripetal jerk and tangent jerk, respectively. It demonstrates that if the feedrate profile is planned with the constraints of maximum centripetal and tangent accelerations, the acceleration on each axis would not exceed the limitation and the same token for jerk. Therefore, tool chattering or system vibration due to high jerk can be avoided when adopting the settings.

### 3.2. NURBS curve splitting on breakpoints

The function of curve splitting plays an important role in the off-line curve scanning method. Since $G^1$ continuity is necessary for calculating curve length accurately by numerical integration method, if a $p$th-degree NURBS curve has $p$ repeated knots in the knot vector except first and last $(p+1)$ knots, the curve should be split into at least two sub-curves. If a knot has multiplicity $r = p$, a breakpoint with $G^0$ continuity may occur. After inserting $(p-r+1)$ knots at the knot with multiplicity $r$, an original NURBS curve can be split into two NURBS sub-curves using the recursive algorithm of knot refinement [1].

A hat curve is provided here to illustrate the concept of curve splitting. The curve is a degree two NURBS curve with 9 control points shown in Fig. 2. The parameters of the curve are listed in Table 1 unless stated otherwise. Since its knot vector $U = \{0, 0, 0, 1/6, 1/3, 1/3, 2/3, 5/6, 1, 1, 1\}$ has two knots with multiplicity 2 at the parameter $u$ i.e., 1/3 or 2/3, two breakpoints represented by $D$ and $E$ are detected. Therefore, the curve is split into three new NURBS sub-curves such as $\overline{AD}$, $\overline{DE}$ and $\overline{EA}$ shown in Fig. 2. In this paper, the values of two knots and two breakpoints are recorded in a curve scanning algorithm. The scanning data will be used to adjust the feedrate profile and to estimate the length of a NURBS curve in Sections 3.3 and 3.5, respectively.

### 3.3. Feedrate adjusting at breakpoints with $G^0$ continuity

While the tool moves across breakpoints with $G^0$ continuity, it may result in violent change of acceleration or jerk on each axis. To maintain the continuity of acceleration at the breakpoints, it is inefficient that the feedrate in one of the axes should approach to zero. The method in [26] for limiting velocity variation at breakpoints is adopted to solve this problem. As shown in Fig. 3, $P_{i-1}, P_i$ and $P_{i+1}$ are the current position, breakpoint and next position, $F$ is the feedrate command across the breakpoint, $\theta_{i-1}$ is the angle between the line $\overline{P_{i-1}P_i}$ and $x$-axis, $\theta_i$ is the angle between the line $\overline{P_iP_{i+1}}$ and $x$-axis. Therefore, the velocity variation on $x$-axis and $y$-axis are given as

$$\begin{aligned} \Delta V_x &= F|\cos\theta_i - \cos\theta_{i-1}| \\ \Delta V_y &= F|\sin\theta_i - \sin\theta_{i-1}|. \end{aligned} \tag{11}$$

If the velocity variation on any axis is larger than the limit of velocity variation $\Delta V_{\max}$, it is necessary to adjust the feedrate command $F$ to meet the constraints of velocity variation, acceleration and jerk using the following equations:

$$\begin{aligned} \Delta V_{\max} &= \min\left(A_{\max}T_s, J_{\max}T_s^2/2\right) \\ R_v &= \max(\Delta V_x, \Delta V_y)/\Delta V_{\max} \\ V_i &= \frac{F}{R_v} = \frac{\min\left(A_{\max}T_s, J_{\max}T_s^2/2\right)}{\max(|\cos\theta_i - \cos\theta_{i-1}|, |\sin\theta_i - \sin\theta_{i-1}|)} \end{aligned} \tag{12}$$

where $A_{\max}, J_{\max}$ and $R_v$ are the maximum acceleration and jerk on each axis, and scaling factor, respectively. Here, $\Delta V_{\max}$ is set as $\min\left(A_{\max}T_s, J_{\max}T_s^2/2\right)$. Eq. (12) illustrates that the acceleration or jerk on any axis caused by velocity variation should never exceed the limits of acceleration and jerk within one sampling time. The curve parameter $u_i$ and suitable feedrate $V_i$ at each breakpoint are recorded and will be used to plan the jerk-limited feedrate profile in Section 4.

### 3.4. Feedrate adjusting at critical points with large curvatures

Although feedrates at breakpoints have been adjusted according to kinematic property in Section 3.3, sharp corners with large curvatures in a NURBS curve still could violate kinematic property for high-speed machining. To find out critical points which are generally called sharp corners and determine their corresponding feedrates, three constraints of chord error, centripetal acceleration and jerk are considered simultaneously in Eq. (13). The critical curvature $\kappa_{cr}$ for identifying the critical points is given as

$$\kappa_{cr} = \min\left(\frac{8\delta}{(V_{\max}\cdot T_s)^2 + 4\delta^2}, \frac{A_n}{V_{\max}^2}, \sqrt{\frac{J_n}{V_{\max}^3}}\right) \tag{13}$$

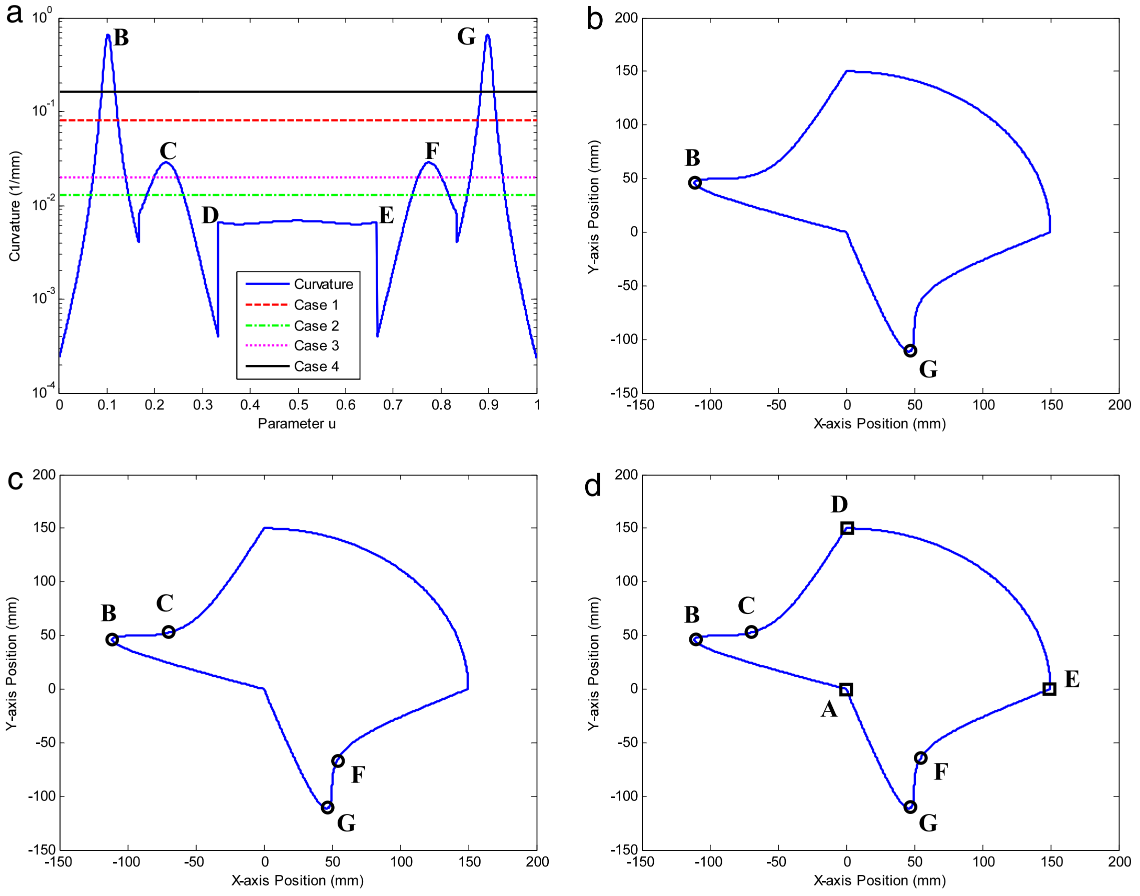


**Fig. 4.** The geometry and kinematic properties of a hat contour (a) curvature profile of hat curve; (b) critical points in Cases I and VI; (c) critical points in Cases II and III; (d) critical points and breakpoints in Case II.

where $\delta, T_s$ and $V_{\max}$ are chord tolerance, sampling time and maximum feedrate, respectively. In Eq. (13), the first condition is derived from the adaptive-feedrate interpolation algorithm [10]; the second and third conditions are derived from the equations of centripetal acceleration and jerk in [17]. Any interpolation point with its curvature $\kappa_i$ being larger than the critical curvature is defined as a *candidate point* since it exceeds one of three constraints. The point which has the local maximum curvature among the candidate points is defined as a *critical point* and its suitable feedrate $V_i$ is modified according to the following equation.

$$V_i = \min\left(\frac{2}{T_s}\sqrt{\frac{1}{\kappa_i^2} - \left(\frac{1}{\kappa_i} - \delta\right)^2}, \sqrt{\frac{A_n}{\kappa_i}}, \sqrt[3]{\frac{J_n}{\kappa_i^2}}\right) \tag{14}$$

where $\kappa_i$ is the curvature of the critical point. The hat curve is provided to further demonstrate the concept. The contour and curvature profiles of the hat curve are shown in Figs. 2 and 4(a), respectively. In general, one could expect that four points marked as B, C, F and G should be identified as critical points because the curvatures at these points are local maximum. In fact, the points C and F might satisfy three constraints under some geometry and kinematic conditions, only the points B and G need to be regarded as critical points. In order to demonstrate how many critical points are detected under different conditions, four cases are tested and the corresponding parameters are listed in Table 2. Case I includes the following default values such as $\delta = 1\ \mu\text{m}$, $V_{\max} = 100$ mm/s, $A_n = 800$ mm/s$^2$ and $J_n = 26\,400$ mm/s$^2$. In Case I, two critical points of B and G are detected as shown in Fig. 4(a) and (b). The critical curvature ($\kappa_{\text{cr}} = 0.08$) is determined by the second condition in Eq. (13), and the suitable feedrates at the critical points are constrained by centripetal acceleration. When the given feedrate increases to 250 mm/s in Case II, four critical points of B, C, F and G are obtained as shown in Fig. 4(c). The critical curvature ($\kappa_{\text{cr}} = 0.0128$) is still determined by the second condition. As compared with Case I, if the chord tolerance is reduced from 1 μm to 0.1 μm in Case III, four critical points of B, C, F and G are detected. The critical curvature ($\kappa_{\text{cr}} = 0.02$) is determined by the first condition. The suitable feedrates are limited by chord tolerance. As compared with Case I, if the centripetal acceleration increases to 2000 mm/s$^2$ in Case VI, only two critical points marked as B and G are detected. The critical curvature ($\kappa_{\text{cr}} = 0.0162$) is determined by the third condition. The suitable feedrates are constrained by centripetal jerk. Finally, the curve parameters $u_i$ and suitable feedrates $V_i$ at the critical points are stored and will be utilized in Sections 3.5 and 4.

### 3.5. The length estimation of NURBS blocks

After detecting all breakpoints and critical points within a NURBS curve, their parameters $u_i$ and suitable feedrate $V_i$ are obtained; the curve is divided into small NURBS blocks corresponding to the scanning data $(u_i, V_i)$. For example, the hat curve is divided into seven NURBS blocks $\overline{AB}, \overline{BC}, \overline{CD}, \overline{DE}, \overline{EF}, \overline{FG}$ and $\overline{GA}$ shown in Fig. 4(d). The length of each NURBS block is required for feedrate scheduling. However, it is difficult to calculate the length by integrating NURBS parametric equation. The adaptive Lobatto quadrature method [25] is adopted to estimate the length of each NURBS block accurately. Given a tolerance of length error $\varepsilon$, the length of each block $S_i$ is calculated over the parameter interval

**Table 2**
The parameters of four test cases for the hat curve (fixed $J_n = 26\,400$ mm/s$^3$).

| | Chord tolerance $\delta$ ($\mu$m) | Feedrate $V_{\max}$ (mm/min) | Centripetal acceleration $A_n$ (mm/s$^2$) | Critical curvature $\kappa_{cr}$ (1/mm) |
|---|---|---|---|---|
| Case I | 1.0 | 100 | 800 | min(0.20, 0.08, 0.162) |
| Case II | 1.0 | 250 | 800 | min(0.032, 0.0128, 0.041) |
| Case III | 0.1 | 100 | 800 | min(0.02, 0.08, 0.162) |
| Case VI | 1.0 | 100 | 2000 | min(0.199, 0.20, 0.162) |

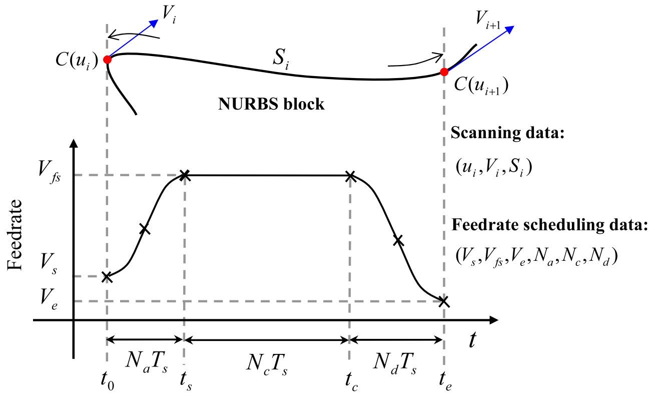


**Fig. 5.** Feedrate scheduling for a NURBS block.

$[u_i^s, u_i^e]$ using the following equation:

$$S_i \approx \int_{u_i^s}^{u_i^e} C'(u)\mathrm{d}u \pm \varepsilon \tag{15}$$

where $i$ is the index of each NURBS block and $u_i^s/u_i^e$ are the start/end curve parameters of $i$th block. After performing the procedure of curve scanning, the scanning data $(u_i, V_i, S_i)$ for each NURBS block will be provided to the feedrate scheduling algorithm which is presented in the next section.

## 4. Feedrate scheduling algorithm

When the process of curve scanning is finished, the scanning data $(u_i, V_i, S_i)$ are obtained and stored. In the stage of feedrate scheduling, the goal is to construct a jerk-limited feedrate profile for each NURBS block through the scanning data. Its concept is shown in Fig. 5, where $C(u_i)$ and $C(u_{i+1})$ are the start and end points in each NURBS clock, $V_i$ and $V_{i+1}$ are the corresponding feedrates at the start/end points, $S_i$ is the length of each NURBS block. $V_s$, $V_e$ and $V_{fs}$ denote the start, end, and constant feedrate after feedrate scheduling, respectively. $N_a$, $N_c$ and $N_d$ are the number of sampling time in the ACC, constant feedrate (CF) and DEC sections of feedrate profile. Finally, the feedrate profile and corresponding parameters $(V_s, V_{fs}, V_e, N_a, N_c, N_d)$ of a NURBS curve are obtained by connecting those feedrate profiles for each NURBS block.

### *4.1. Sine-curve velocity profile*

Since sine-curve velocity profile shown in Fig. 6 is more continuous than trapezoid and blended spline velocity profile, it is chosen to generate the feedrate profile for each NURBS block in this paper and its velocity equation is given as [27]:

$$V_{tan}(t) = \begin{cases} \frac{V_{fs}-V_s}{2}\left[\sin\pi\left(\frac{t-t_0}{T_a}-\frac{1}{2}\right)+1\right]+V_s, & t_0 \le t < t_s \\ V_{fs} \quad t_s \le t < t_c \\ \frac{V_{fs}-V_e}{2}\left[\sin\pi\left(\frac{t-t_c}{T_d}-\frac{3}{2}\right)+1\right]+V_e, & t_c \le t < t_e \end{cases} \tag{16}$$

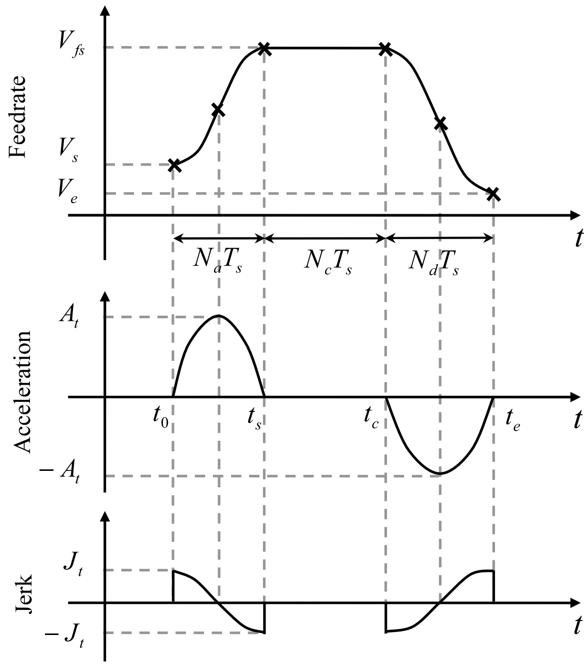


**Fig. 6.** Sine-curve velocity profile.

where $T_a = N_a \cdot T_s$ and $T_d = N_d \cdot T_s$. Differentiating Eq. (16) yields the acceleration equation,

$$A_{tan}(t) = \begin{cases} \frac{V_{fs}-V_s}{2}\frac{\pi}{T_a}\cos\pi\left(\frac{t-t_0}{T_a}-\frac{1}{2}\right), & t_0 \le t < t_s \\ 0 & t_s \le t < t_c \\ \frac{V_{fs}-V_e}{2}\frac{\pi}{T_a}\cos\pi\left(\frac{t-t_c}{T_a}-\frac{3}{2}\right), & t_c \le t < t_e. \end{cases} \tag{17}$$

Differentiating Eq. (17), one obtains the jerk equation,

$$J_{tan}(t) = \begin{cases} -\frac{V_{fs}-V_s}{2}\left(\frac{\pi}{T_a}\right)^2\sin\pi\left(\frac{t-t_0}{T_a}-\frac{1}{2}\right) & t_0 \le t < t_s \\ 0 & t_s \le t < t_c \\ -\frac{V_{fs}-V_e}{2}\left(\frac{\pi}{T_a}\right)^2\sin\pi\left(\frac{t-t_c}{T_a}-\frac{3}{2}\right), & t_c \le t < t_e. \end{cases} \tag{18}$$

The limits of tangent acceleration and jerk can be determined by setting the feedrate $V_{fs} = V_{\max}$:

$$\begin{aligned} |A_{tan}(t)| &= \left|\frac{V_{fs}-V_s}{2}\frac{\pi}{T_a}\cos\pi\left(\frac{t-t_0}{T_a}-\frac{1}{2}\right)\right| \\ &\le \frac{V_{fs}-V_s}{2}\frac{\pi}{T_a} \le A_t \\ J_{tan}(t) &= \left|\frac{V_{fs}-V_s}{2}\left(\frac{\pi}{T_a}\right)^2\sin\pi\left(\frac{t-t_0}{T_a}-\frac{1}{2}\right)\right| \\ &\le \frac{V_{fs}-V_s}{2}\left(\frac{\pi}{T_a}\right)^2 \le J_t. \end{aligned} \tag{19}$$

### *4.2. Feedrate profile generation for each NURBS block*

In this section, the procedure of feedrate scheduling for each NURBS block is shown in Fig. 7 and the details are illustrated schematically as follows.

*Step* I.

In the first step, it needs to calculate the number of sampling time $N_a$ and $N_d$ in the ACC/DEC sections of sine-curve profile. For given start feedrate $V_s$, allowable feedrate $V_{fs} = V_{\max}$ and limits of tangent acceleration and jerk, the number $N_a$ can be derived from Eq. (19),

$$N_a = \max\left(\frac{V_{fs}-V_s}{2}\frac{\pi}{A_t T_s}, \sqrt{\frac{V_{fs}-V_s}{2}\frac{1}{J_t}\left(\frac{\pi}{T_s}\right)^2}\right). \tag{20}$$

The process for calculating $N_d$ is similar to that of calculating $N_a$. The areas under the ACC/DEC sections of sine-curve profile are derived as

$$\begin{aligned} S_a &= \int_{t_0}^{t_s}\left\{\frac{V_{\max}-V_s}{2}\left[\sin\pi\left(\frac{t-t_0}{T_a}-\frac{1}{2}\right)+1\right]+V_s\right\}\mathrm{d}t \\ &= \frac{(V_s+V_{\max})\cdot N_a T_s}{2} \\ S_d &= \int_{t_c}^{t_e}\left\{\frac{V_{\max}-V_e}{2}\left[\sin\pi\left(\frac{t-t_c}{T_d}-\frac{3}{2}\right)+1\right]+V_e\right\}\mathrm{d}t \\ &= \frac{(V_e+V_{\max})\cdot N_d T_s}{2}. \end{aligned} \tag{21}$$

Therefore, the number of sampling time in the CF section of sine-curve profile is obtained:

$$N_c = \frac{S_i - S_a - S_d}{V_{\max} T_s}. \tag{22}$$

If $N_c > 0$, it means that the length of NURBS block $S_i$ is long enough to plan the CF section with maximum feedrate $V_{\max}$, and the feedrate profile is planned as Fig. 7(a). Otherwise, the process goes to *Step* II.

*Step* II.

For the feedrate profile without CF section, it is necessary to reduce the feedrate from maximum feedrate $V_{\max}$ to allowable feedrate $V_{fs}$. It is represented as

$$\begin{aligned} &\left\{\sum_{j=0}^{N_a-1}\frac{(V_{fs}-V_s)}{2}\left[\sin\pi\left(\frac{j}{N_a}-\frac{1}{2}\right)+1\right]\right\}T_s + V_s N_a T_s \\ &\quad+\left\{\sum_{j=0}^{N_d-1}\frac{(V_{fs}-V_e)}{2}\left[\sin\pi\left(\frac{j}{N_d}-\frac{3}{2}\right)+1\right]\right\}T_s \\ &\quad+V_e N_d T_s + V_{fs} N_c T_s = S_i. \end{aligned} \tag{23}$$

Simplifying Eq. (23), one obtains the allowable feedrate $V_{fs}$ as

$$V_{fs} = \frac{\frac{S_i}{T_s}+\frac{1}{2}(aV_s+bV_e)-V_sN_a-V_eN_d}{\frac{(a+b)}{2}+N_c} \tag{24}$$

where $a = \sum_{j=0}^{N_a-1}[\sin\pi(\frac{j}{N_a}-\frac{1}{2})+1]$ and $b = \sum_{j=0}^{N_d-1}[\sin\pi(\frac{j}{N_d}-\frac{3}{2})+1]$. If the $V_{fs}$ is greater than $\max(V_s, V_e)$, the process goes to feedrate scheduling without CF section as shown in Fig. 7(b). In this case, $N_a$ and $N_d$ are calculated using Eq. (20), and $N_c$ is equal to zero. There are three cases according to the relation between $V_s$ and $V_e$ (i.e., $V_s > V_e$ or $V_s < V_e$ or $V_s = V_e$). Otherwise, the acceleration or jerk may exceed the limitation, so that feedrate scheduling must be performed further and the process goes to *Step* III for the case of $V_{fs} \le \max(V_s, V_e)$.

*Step* III.

If the allowable feedrate $V_{fs}$ is greater than $\min(V_s, V_e)$, the feedrate profile with only ACC or DEC section is considered as shown in Fig. 7(c). In this case, the $V_{fs}$ is set as $\max(V_s, V_e)$, and the parameter $N_a$ or $N_d$ is evaluated from Eq. (20). Eq. (24) is used to calculate the $V_{fs}$. Note that if $V_s > V_e$, the parameters $N_a$, $N_c$ and $a$ in Eq. (24) are set to zero; otherwise, the parameters $N_d$, $N_c$ and $b$ are set to zero.

If the $V_{fs}$ is smaller than $\min(V_s, V_e)$ or equal to $\min(V_s, V_e)$, the acceleration or jerk still could exceed the limitation. Therefore, it needs to perform feedrate scheduling further and goes to *Step* IV.

*Step* IV.

For the case of $V_{fs} \le \min(V_s, V_e)$, the feedrate profile with only CF section is adopted as shown in Fig. 7(d). The feedrates $V_s$ and $V_e$ are all set as $\min(V_s, V_e)$. Therefore, the length of NURBS block $S_i$ is the only condition for feedrate scheduling. The $V_{fs}$ is determined through Eq. (24) where the parameters $N_a$, $N_d$, $a$ and $b$ are set to zero, and the process of feedrate scheduling for each NURBS block is finished.

In summary, the concept of feedrate scheduling for each NURBS block is summarized as follows:

1. By the constraints of tangent acceleration and jerk, the numbers of sampling time in the ACC/DEC section of sine-curve velocity profile can be evaluated from Eq. (20).
2. According to the length of NURBS block, the allowable feedrate $V_{fs}$ after feedrate scheduling is evaluated from Eq. (24).
3. Finally, the feedrate scheduling data $(V_s, V_{fs}, V_e, N_a, N_c, N_d)$ are determined by the feedrate scheduling algorithm.

### *4.3. Short NURBS block determination and prior handling*

In the process of feedrate scheduling proposed in the last section, the boundary condition $V_s$ or $V_e$ may be changed in some situations. This problem may cause jump and discontinuity at the junction of two feedrate profiles corresponding to two adjacent NURBS blocks. Furthermore, the acceleration and jerk could exceed the limitation.

For example, if the length of NURBS block is too short, its feedrate profile could be changed from the form of Fig. 7(b) to the form of Fig. 7(c) or (d) in the process of feedrate scheduling, so $V_s$ or $V_e$ is changed, too. Therefore, one hope to determine how "short" NURBS block the boundary condition may be changed. As shown in Fig. 8, the criterion length of NURBS block satisfied $V_{fs} = \max(V_s, V_e)$ can be derived as

$$S_{std} = \begin{cases} \frac{1}{2}(V_s+V_e)N_aT_s + V_eN_dT_s, & V_s < V_e \\ \frac{1}{2}(V_s+V_e)N_dT_s + V_sN_aT_s, & V_s > V_e \\ (N_a+N_d)V_sT_s & V_s = V_e. \end{cases} \tag{25}$$

Through Eq. (25), if $S_i \ge S_{std}$, the boundary condition is not changed after feedrate scheduling process and the corresponding NURBS block is called a "long" NURBS block. Otherwise, the NURBS block is called a "short" NURBS block. For avoiding the jump and discontinuity at the junction of two feedrate profiles, it is necessary to check if the next NURBS block is "short" or not before scheduling the feedrate profile for $i$th NURBS block. If so, searching all of the following short NURBS blocks is performed in priority, e.g., $(i+1)$th, $(i+2)$th, ..., $(i+k)$th block are short. After feedrate scheduling for those short NURBS blocks and $i$th NURBS block is performed in the opposite direction, e.g. $(i+k)$th, $(i+k-1)$th, ..., $(i+1)$th, $i$th, the work of feedrate scheduling is moved to $(i+k)$th block and goes on. The concept of feedrate rescheduling for short NURBS blocks is illustrated in Fig. 9.

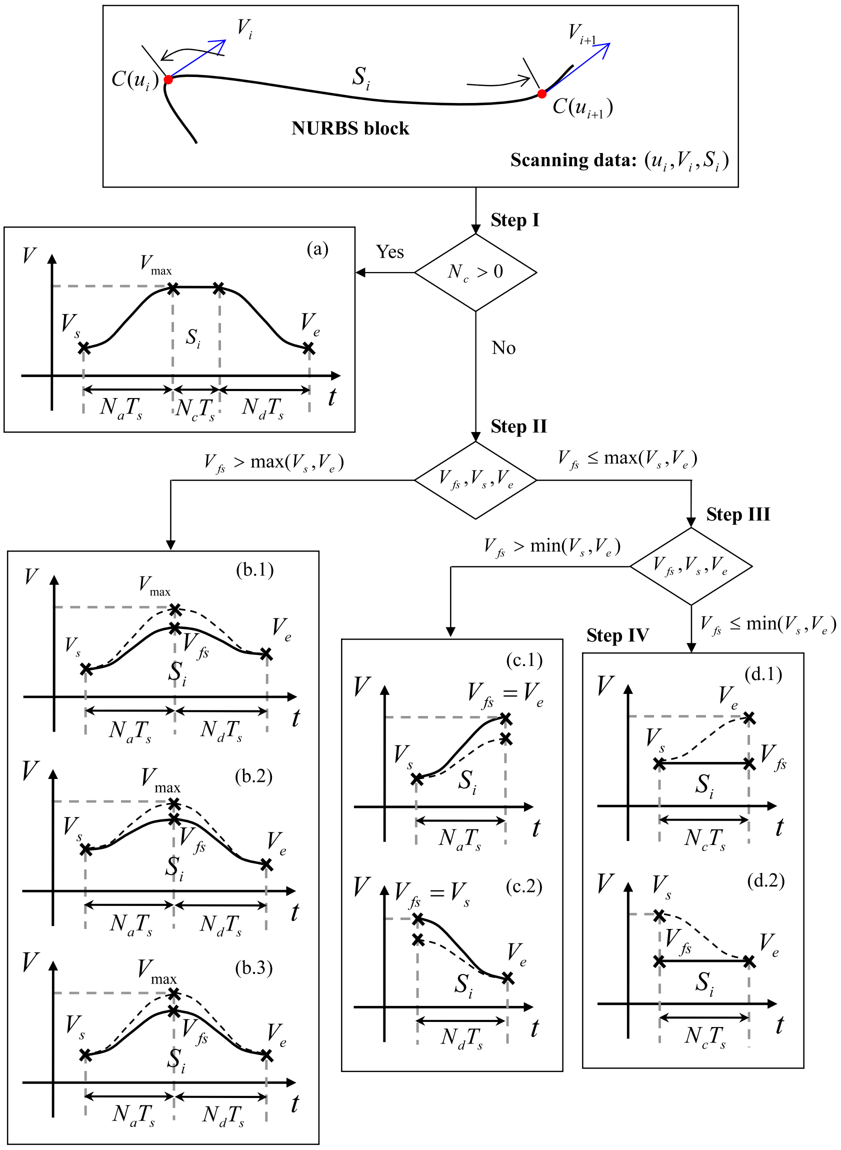

NURBS block
Scanning data: $(u_i, V_i, S_i)$
Step I
$N_c > 0$
Yes
No
Step II
$V_{fs} > \max(V_s, V_e)$
$V_{fs} \le \max(V_s, V_e)$
Step III
$V_{fs} > \min(V_s, V_e)$
$V_{fs} \le \min(V_s, V_e)$
Step IV
$V_{fs} = V_e$
$V_{fs} = V_s$

**Fig. 7.** The flowchart of feedrate profile generation for each NURBS block.

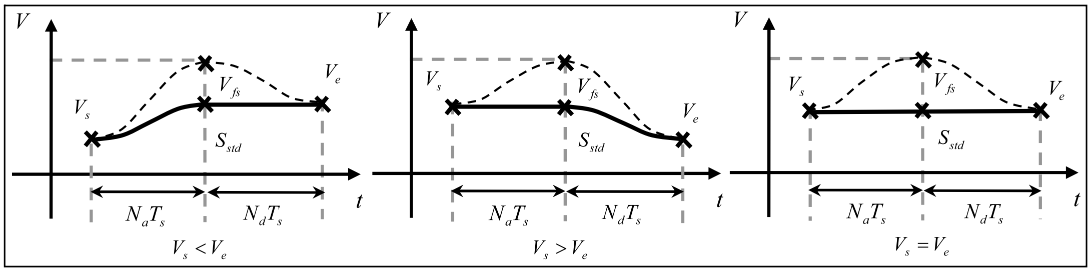


**Fig. 8.** The criterion of short NURBS block.

### 4.4. Terminal error compensation of NURBS block

The work of off-line feedrate scheduling for the whole NURBS curve is over up to here. Since the length $S_i$ of each NURBS block is estimated by the adaptive Lobatto quadrature method, estimation error exists between actual moving length and estimated length, and it may cause a problem of terminal error after interpolation. The actual moving length $S_i^{\text{itr}}$ after interpolation and the estimation error $\Delta S_i$ are calculated using the following equations:

$$S_i^{\text{itr}} = \sum_{j=1}^{N_{\text{itr}}} \left| C(u_{j+1}) - C(u_j) \right| \tag{26}$$

$$\Delta S_i = S_i^{\text{itr}} - S_i \tag{27}$$

where $i$ is the index of NURBS block and $N_{\text{itr}}$ is the number of interpolation points within the $i$th block. To solve the terminal error problem, the error distance is compensated by designing a compensated feedrate curve for distributing it in the last $N_{\text{com}}$

**Table 3**
Parameters of kinematic constraints and feedrate scheduling method.

| Parameters | Symbols | Units |
|---|---|---|
| Sampling time | $T_s$ | 0.002 s |
| Chord error | $\delta$ | 1 µm |
| Maximum feedrate | $V_{\max}$ | 250 mm/s |
| Maximum acceleration | $A_{\max}$ | 800 mm/s$^2$ |
| Maximum jerk | $J_{\max}$ | 26 400 mm/s$^3$ |
| Length estimation error | $\varepsilon$ | 0.1 µm |
| Feedrate error of PCI | $\varepsilon_{\text{PCI}}$ | 0.1% |
| The correctional coefficient of PCI | $\beta$ | 0.9 |
| The number of feedrate compensation | $N_{\text{com}}$ | 50 |

sampling time. The sine-curve interior $\left[-\frac{\pi}{2}, \frac{3\pi}{2}\right]$ is utilized as the compensated feedrate curve shown at the bottom of Fig. 1. Its equation is represented as

$$\Delta V(j) = \frac{\Delta V_{\text{com}}}{2}\left[\sin\pi\left(\frac{2j}{N} - \frac{1}{2}\right) + 1\right]. \tag{28}$$

Let the area under the compensated feedrate curve be equal to $\Delta S_i$, and represented as

$$\Delta S_i = \sum_{j=0}^{N_{\text{com}}-1} \Delta V(j) T_s \tag{29}$$

where $N_{\text{com}}$ is the number of sampling time of the compensated feedrate curve. The parameter $\Delta V_{\text{com}}$ can be obtained from Eqs. (28) and (29). If $\Delta V_{\text{com}}$ is positive, the actual moving length of NURBS block is larger than the estimated length, i.e. $\Delta S_i > 0$; otherwise, the actual moving length is smaller than the estimated length, i.e. $\Delta S_i < 0$.

To calculate the error distance, the interpolation should be performed for each NURBS block. The sum of the original feedrate profile and compensated feedrate profile becomes the new feedrate command profile which is used in the final $N_{\text{com}}$ times interpolation. It is noted that only $N_{\text{com}}$ sampling points need to backtrack, and the compensated feedrate curve generates new curve points in real-time interpolation process. Finally, the problem of terminal error is eliminated via the process of feedrate compensation.

## 5. Simulation and experimental verification

In the following, analytical simulations and experiments are performed to demonstrate the feasibility and applicability of the proposed feedrate scheduling method. The performance evaluation among first-order Taylor, second-order Taylor and PCI interpolation algorithms are also performed to verify the efficiency of the proposed method.

### 5.1. Efficiency and accuracy analysis

The environment of simulation consists of Intel Core i7-860 2.8 GHz personal computer with Windows XP operating system, and the proposed feedrate scheduling method is developed by MATLAB. The PCI interpolation algorithm in [7] is applied to generate curve parameter. Even though PCI has a disadvantage of stability problem that corrector may not converge at breakpoints with $G^0$ continuity, it still works in this paper since the NURBS curve which contains the breakpoints has been split into several NURBS sub-curves. In this section, two free-form curves are chosen to simulate and to compare the efficiency and accuracy among the first-order Taylor, second-order Taylor and PCI interpolation algorithms. The parameters of kinematic constraints and feedrate scheduling method are presented in Table 3 unless stated otherwise.

To illustrate the computing efficiency, the number and total clock cycles of arithmetic operations required to realize various interpolation techniques are summarized in Tables 4 and 5, respectively. The degree of NURBS curve $p$, the number of motion axes $N_a$ and the iteration number of PCI $N_j$ are considered as performance influencing factors. To improve feedrate accuracy, PCI algorithm requires to perform Eqs. (6)–(9) recursively. Since the Cox–de Boor algorithm needs $\frac{p(p+1)}{2}$ times of function calls to generate an interpolation point $C(u)$, the total number of ADD, SUB, MUL and DIV operations for PCI increases proportional to the factor $(N_j - 1)N_a\frac{p(p+1)}{2}$ in [29]. In contrast to PCI, since the Taylor series methods only need to calculate the first and second derivatives of a NURBS curve in Eq. (4) using the Cox–de Boor algorithm, the total number for the first- and second-order Taylor methods increases proportional to the factors $N_a\frac{p(p-1)}{2}$ and $N_a(p-1)^2$, respectively. Therefore, the computation of PCI algorithm consumes the most clock cycles as compared with the others shown in Table 5. In the hat case with $p = 2, N_a = 3, N_j = 2$, the total clock cycles consumed by PCI is about 4.08 and 6.63 times of that consumed by the first- and second-order Taylor methods, respectively.

Although computing efficiency of PCI algorithm is not better than the Taylor series methods, it is still worth to apply PCI method to reduce feedrate fluctuation for high-speed machining. Therefore, the hat and butterfly curves are tested to prove the accuracy and feasibility of the proposed feedrate scheduling algorithm with PCI method. The adaptive-feedrate interpolation algorithm (ADA), adaptive-feedrate with curvature-based feedrate interpolation algorithm (ADACB) and proposed feedrate scheduling method (FS) are adopted to plan the feedrate profile of a NURBS curve; the first- and second-order Taylor methods, and PCI methods are applied to generate an interpolation point. The comparisons of feedrate errors among various NURBS interpolation algorithms are summarized in Table 6. It is clear that the PCI method achieves the best feedrate accuracy as compared with the Taylor methods for the two curves. Furthermore, the FS with PCI method can outperform all of the other algorithms both in maximum and root mean square of feedrate accuracy. The corresponding feedrate error is within the tolerance 0.1%, and root mean square of feedrate error is only 0.04%. It is noted that the ADA algorithm with first-order Taylor method is not suitable for high-speed machining since the maximum feedrate fluctuations are over a hundred percent of given feedrate command. In the following sections, simulations are conducted to demonstrate the feasibility and applicability of the proposed feedrate scheduling method.

**Table 4**
Arithmetic operations of various NURBS interpolation methods.

| Arithmetic operation | First-order Taylor | Second-order Taylor | PCI |
|---|---|---|---|
| ADD | $N_a + N_a p(p-1)/2$ | $N_a(p-1)^2 + N_a + 2$ | $(N_j - 1)\,[N_a p(p+1)/2 + N_a + 1] + N_a(p^2 + p + 1)$ |
| SUB | $3N_a p(p-1)/2$ | $3N_a(p-1)^2 + 1$ | $(N_j - 1)\,[N_a p(p+1)/2 + N_a + 3] + N_a(3p^2 + 3p + 1) + 2$ |
| MUL | $N_a(p^2 - p + 1) + 1$ | $2N_a(p-1)^2 + N_a + 13$ | $(N_j - 1)\left[N_a(p^2 + p + 1) + 2\right] + N_a(2p^2 + 2p + 1) + 2$ |
| DIV | $N_a p(p-1)/2 + 1$ | $N_a(p-1)^2 + 4$ | $(N_j - 1)\,[N_a p(p+1)/2 + 3] + N_a(p^2 + p) + 2$ |
| SQRT | 1 | 1 | $N_j$ |

Note. ADD: Addition, SUB: Subtraction, MUL: Multiplication, DIV: Division, SQRT: Square Root, $p$: the degree of NURBS curve, $N_a$: the number of motion axes, $N_j$: the number of iterations for PCI.

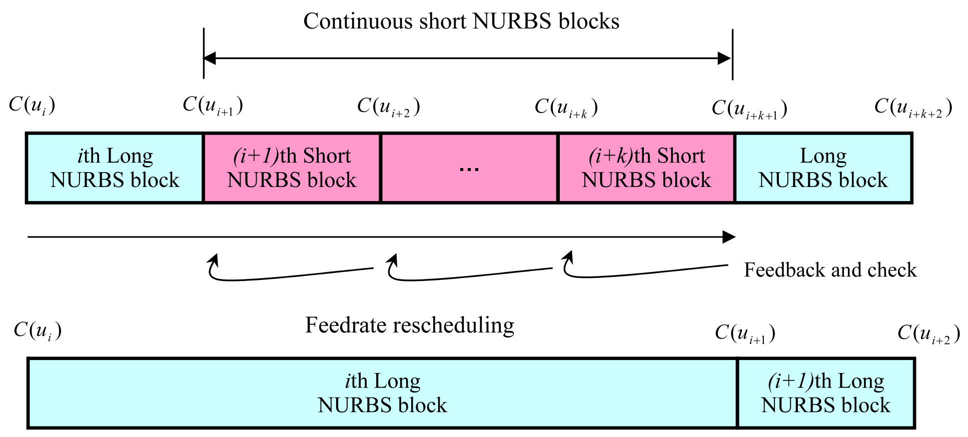


**Fig. 9.** The rescheduling process for short NURBS blocks.

**Table 5**
Total clock cycles of various NURBS interpolation methods (fixed $N_a = 3$, $N_j = 2$).

| NURBS curve | First-order Taylor | Second-order Taylor | PCI |
|---|---|---|---|
| Hat ($p = 2$) | 225 | 366 | 1492 |
| Butterfly ($p = 3$) | 501 | 780 | 2680 |

Note. The clock cycles for performing double-precision arithmetic operations on Intel Core i7 chip are given as ADD(3), SUB(3), MUL(5), DIV(24), SQRT(34) in [28].

## 5.2. Simulations of hat curve

The first NURBS curve is a hat contour shown in Fig. 2 and the parameters of simulations are listed in Table 3. The hat curve has nine control points and its degree is two. Fig. 10(b) illustrates that the proposed algorithm detects three breakpoints, i.e., A, D, and E, and divides the curve into three NURBS sub-curves ($\overline{AD}$, $\overline{DE}$ and $\overline{EA}$) using the curve splitting method. Furthermore, the proposed method obtains four zones where the curvatures are larger than the critical curvature $\kappa_{rc} = 0.0128$ as shown in Fig. 10(c). It also finds out four critical points, i.e., B, C, F, and G, and further divides the curve into seven NURBS blocks such as $\overline{AB}$, $\overline{BC}$, $\overline{CD}$, $\overline{DE}$, $\overline{EF}$, $\overline{FG}$ and $\overline{GA}$ based on the breakpoints and critical points. Finally, the sine-curve feedrate profile for the hat curve is planned as shown in Fig. 10(d). As shown in Fig. 11(a)–(c), the profiles of chord error, acceleration and jerk generated by the proposed method are all constrained on the values of 1 μm, 800 mm/s$^2$ and 26 400 mm/s$^3$, respectively. As shown in Fig. 11(d)–(f), the velocity, acceleration and jerk profiles of $x$-axis and $y$-axis are also constrained on the given limits of 250 mm/s, 800 mm/s$^2$ and 26 400 mm/s$^3$, respectively. From the simulation results, it demonstrates that the precision and the kinematic constraints of CNC machine are all satisfied.

It is obvious that the curve consists of two breakpoints D and E with $G^0$ continuity within the curve. Since the directions of the motion path change instantaneously while crossing these two breakpoints, the feedrate variation would cause the individual motion axis to exceed the constraints of acceleration and jerk if the feedrates are not adjusted. This example shows that not only the proposed algorithm can correct the feedrates at two breakpoints of D and E to suitable values using Eq. (12), but also the acceleration and jerk are bounded. Besides, the feedrates at four critical points of B, C, F, and G are adjusted according to Eq. (14). Note that the conditions for the breakpoints with $G^0$ continuity are stricter than those for critical points with $G^1$ continuity. That is why the feedrates at the breakpoints of D and E are far less than those at the critical points of B, C, F, and G shown in Fig. 10(d).

## 5.3. Simulations of butterfly curve

To further illustrate the merits of the proposed feedrate scheduling method, one considers the second NURBS curve which is more complex than the hat curve. The butterfly curve has 51 control points and its degree is three in [29]. Fig. 12(a)–(d) show the butterfly contour, its sub-curves and blocks, its curvature and feedrate profile, respectively. For the given parameters listed in Table 3, the critical curvature $\kappa_{cr}$ is obtained as 0.0128, the proposed method identifies 31 critical points marked as $C_1$ to $C_{31}$ as shown in Fig. 12(b)–(c), and the curve is divided into 32 NURBS blocks including $\overline{C_0C_1}$, $\overline{C_1C_2}$, $\overline{C_2C_3}$, $\overline{C_3C_4}$, ..., $\overline{C_{30}C_{31}}$ and $\overline{C_{31}C_0}$. The algorithm further estimates the length of each block and plans the sine-curve feedrate profile for each block. It is noted that the curve detects ten short blocks such as $\overline{C_4C_5}$, $\overline{C_8C_9}$, $\overline{C_9C_{10}}$, $\overline{C_{13}C_{14}}$, $\overline{C_{14}C_{15}}$, $\overline{C_{15}C_{16}}$, $\overline{C_{16}C_{17}}$, $\overline{C_{22}C_{23}}$, $\overline{C_{23}C_{24}}$ and $\overline{C_{28}C_{29}}$ utilizing the criterion of short NURBS block in Eq. (25). Therefore, the short blocks are combined with the nearby long blocks through the feedrate rescheduling process as shown in Fig. 9. For example, since the block $\overline{C_4C_5}$ is too short and its end velocity $V_e$ is too low, its deceleration profile cannot be planned

**Table 6**
Feedrate errors of various NURBS interpolation algorithms (fixed $V_{\max} = 250$ mm/s).

| NURBS curve | Interpolation algorithm | ADA | ADACB | FS (proposed) |
|---|---|---|---|---|
| | | MAX/RMS (%) | MAX/RMS (%) | MAX/RMS (%) |
| Hat | First-order Taylor | 137.00/10.81 | 32.72/10.58 | 15.44/4.20 |
| | Second-order Taylor | 134.94/10.80 | 31.60/10.57 | 12.79/4.10 |
| | PCI | 0.10/0.04 | 0.10/0.04 | 0.10/0.04 |
| Butterfly | First-order Taylor | 12.82/2.25 | 12.69/2.17 | 11.18/2.06 |
| | Second-order Taylor | 3.14/0.42 | 3.04/0.39 | 2.67/0.28 |
| | PCI | 0.10/0.04 | 0.10/0.04 | 0.10/0.04 |

Note. ADA: Adaptive-feedrate interpolation algorithm, ADACB: Adaptive-feedrate with curvature-based feedrate interpolation algorithm, FS: The proposed feedrate scheduling method.

**Table 7**
Parameters of the servo control systems and motion controllers.

| Quantity | Symbol | Value | | | Units |
|---|---|---|---|---|---|
| | | $x$-axis | $y$-axis | $z$-axis | |
| Servo system dynamics | $a$ | 35.833 | 58.532 | 23.117 | s |
| | $b$ | $3.640 \times 10^{-3}$ | $4.094 \times 10^{-3}$ | $0.425 \times 10^{-3}$ | mm/Volt |
| Position controller | $K_{pp}$ | 148.200 | 153.128 | 145.242 | 1/s |
| Velocity controller | $K_{vp}$ | $11.393 \times 10^{-2}$ | $10.285 \times 10^{-2}$ | $96.754 \times 10^{-2}$ | Volts/mm |
| | $T_i$ | $8.172 \times 10^{-3}$ | $7.323 \times 10^{-3}$ | $8.720 \times 10^{-3}$ | 1/s |
| Velocity feedforward controller | $K_{vff}$ | 0.95 | 0.95 | 0.95 | |
| | $\omega_n$ | 500 | 500 | 500 | Hz |

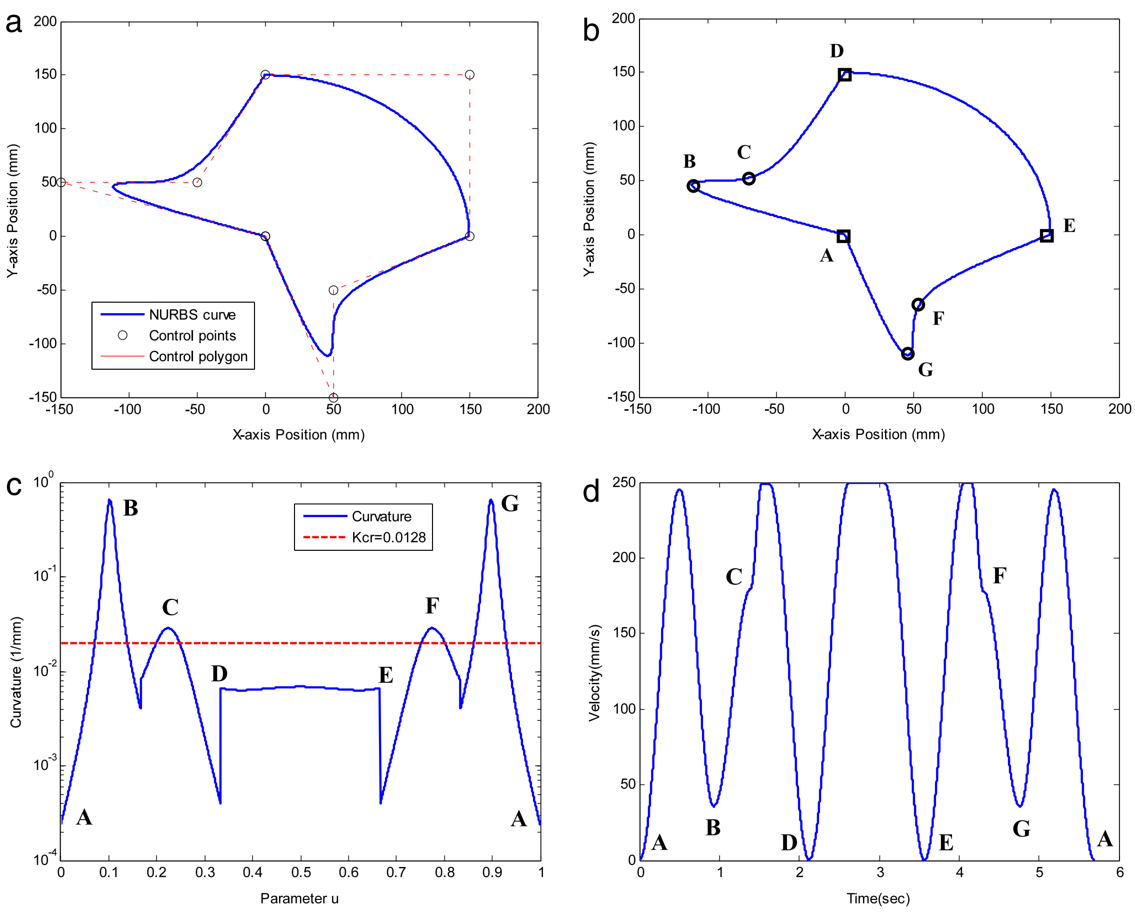


**Fig. 10.** The hat curve: (a) contour, (b) sub-curves and blocks, (c) curvature profile and (d) feedrate profile.

under the given jerk constraint. The proposed algorithm combines the long block $\overline{C_3C_4}$ and the short block $\overline{C_4C_5}$ into one long block $\overline{C_3C_5}$, and plans the deceleration profile for the block as shown in Fig. 7(c.2). Finally, the algorithm reschedules the feedrate profile for each combined block shown in Fig. 12(d), and the number of NURBS blocks decreases from 32 to 26. Comparing Fig. 12(c) with Fig. 12(d), it illustrates that the feedrates at critical points $C_5$, $C_8$, $C_{10}$, $C_{13}$, $C_{19}$, $C_{22}$, $C_{24}$ and $C_{27}$ with larger curvatures are lower than those of other critical points with smaller curvatures.

Again, one can check that the feedrate profile of the butterfly curve satisfies the constraints of chord error, acceleration and jerk limitations shown in Fig. 13(a)–(c). As shown in Fig. 13 (d)–(f), the profiles of velocity, acceleration and jerk for $x$-axis and $y$-axis are also constrained on the given limits of 250 mm/s, 800 mm/s$^2$ and 26 400 mm/s$^3$, respectively. From the simulation results, it demonstrates that the proposed feedrate scheduling method achieves the desired precision and satisfies the kinematic constraints of CNC machine for high-speed machining.

### *5.4. Simulation comparisons among different NURBS interpolation algorithms*

In order to point out the differences among the proposed method and previous researches including dynamic-based interpolator with real-time look-ahead algorithm (DBLA) in [18] and adaptive-feedrate interpolation algorithm (ADA) in [10], some simulations are performed as follows. The block diagram and specification of AC servo control systems are referred to [18]. The parameters of servo control systems and motion controller are listed in Table 7 unless stated otherwise.

For the hat case, since the DBLA and ADA algorithms cannot detect the breakpoints D and E as shown in Fig. 2, they will not suitably decrease the feedrate and consequently deteriorate

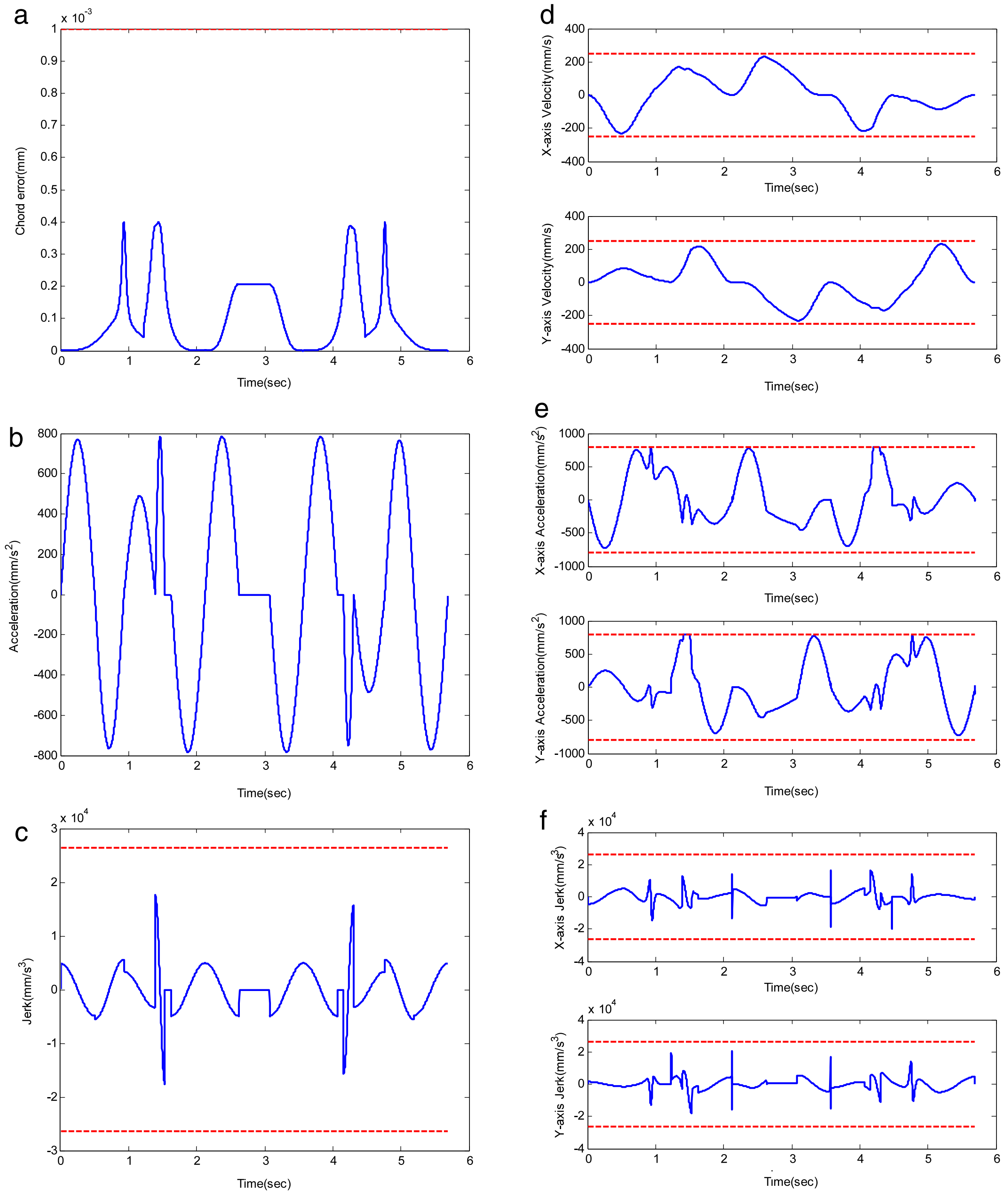


**Fig. 11.** Feedrate scheduling for the hat curve: (a) chord error profile, (b) acceleration profile, (c) jerk profile, (d) *x*- and *y*-axes velocity profiles, (e) *x*- and *y*-axes acceleration profiles and (f) *x*- and *y*-axes jerk profiles.

sharply in large chord, tracking and contouring errors around the breakpoints D and E. However, since the proposed algorithm can handle the problem of $G^0$ continuity in the off-line process, the performance can meet the specifications. The hat curve with 1/4 scale and the feedrate command 1454 mm/s is adopted to demonstrate the merits of the proposed method. Simulation results are shown in Fig. 14. Statistical data are summarized in Table 8 where the proposed method can reduce the maximum contour error by 65.56% and 68.95% as compared to that of DBLA and ADA algorithms, respectively. Furthermore, for both the hat and butterfly cases, if the maximum feedrate command is given as 15 000 mm/min, the sampling time 2 ms, the constraints of chord error, acceleration and jerk 1 µm, 800 mm/s$^2$ and 26 400 mm/s$^3$, respectively, the maximum allowable feedrate $F_{\max}$ for DBLA algorithm can be determined as

$$F_{\max} = \frac{1}{2} \times (2 \times T_a) \times A_{\max} = \frac{A_{\max}}{J_{\max}} \times A_{\max} \approx 24.24\ \text{mm/s}$$
$$= 1454.54\ \text{mm/ min}$$

where $T_a$ is the maximum acceleration time of DBLA algorithm. The DBLA algorithm constrained the maximum feedrate by 1454.54 mm/min, far less than the admissible feedrate. Nevertheless, the proposed feedrate scheduling method can achieve the feedrate command 15 000 mm/min by planning the sine velocity

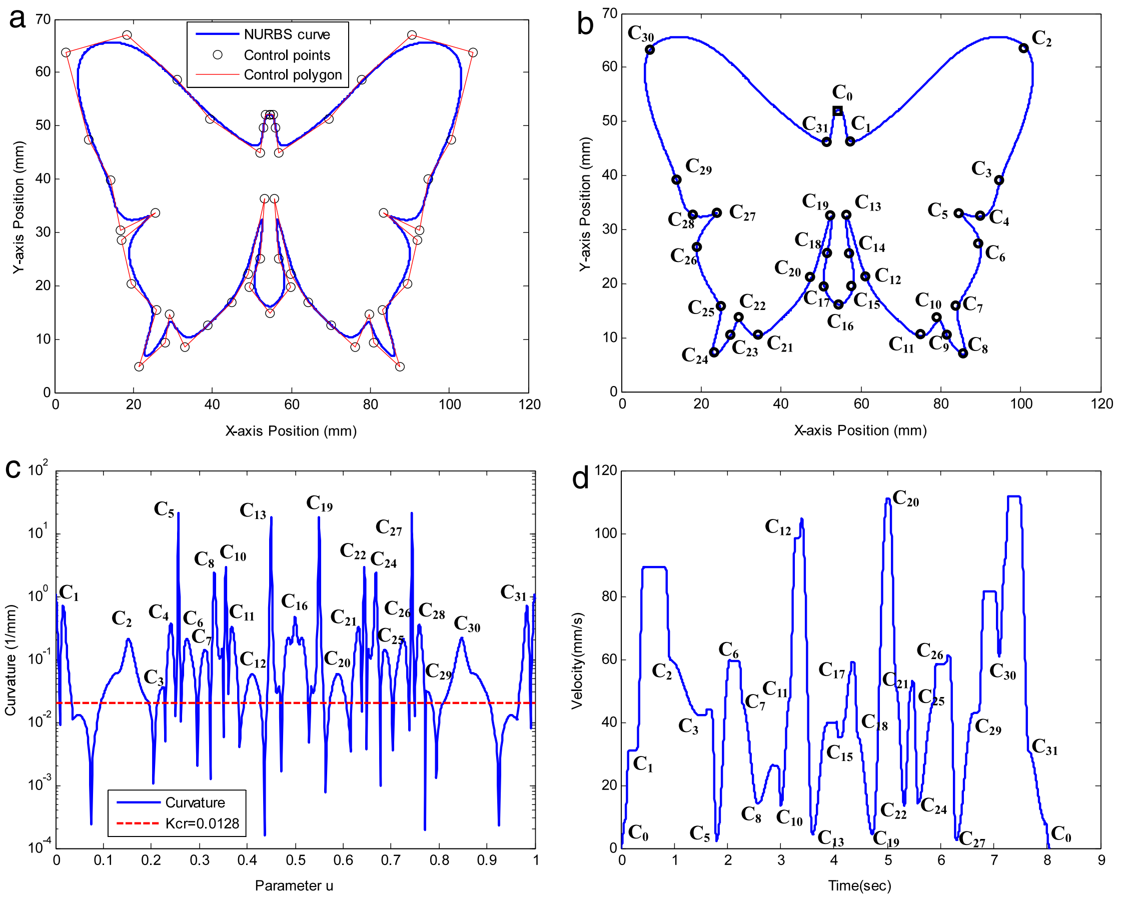


**Fig. 12.** The butterfly curve: (a) contour, (b) sub-curves and blocks, (c) curvature profile and (d) feedrate profile.

**Table 8**
Performance comparisons of simulations among different NURBS interpolation algorithms for hat contour.

| Feedrate command (mm/min) | Interpolation algorithm | Chord error (μm) | | Tracking error (μm) | | Contour error (μm) | | Machining time (s) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | MAX | RMS | MAX ($x/y$-axis) | RMS ($x/y$-axis) | MAX | RMS | PRE | MAC | Total |
| 1454 | ADA | 0.203 | 0.010 | 150.512/61.178 | 10.526/9.097 | 74.330 | 5.395 | 0 | 8.908 | 8.908 |
| | DBLA | 0.060 | 0.007 | 128.752/60.309 | 10.025/8.922 | 67.000 | 4.856 | 0 | 9.110 | 9.110 |
| | FS | 0.057 | 0.007 | 28.737/39.317 | 9.378/8.732 | 23.077 | 4.272 | 33.138 | 9.448 | 42.586 |

Note. MAX: maximum, RMS: root mean square, PRE: pre-process, MAC: machining.

profile through Eqs. (16)–(19); the process of feedrate scheduling is shown in Fig. 7.

## 5.5. Experimental results

In this section, experiments are performed on a three-axis engraving machine with Panasonic MINAS-A4 servo drivers and MSMD042S1S servo motors. The drivers of servo motors are set to torque mode. The off-line feedrate scheduling algorithm is implemented on MATLAB, and the three-axis engraving machine shown in Fig. 15(a) is controlled by a PC-based motion controller with Intel Core i7-860 2.8 GHz microprocessor. The hat and butterfly curves are tested under feedrate command of 15 000 mm/min. Since the maximum velocities of servo drivers are limited to 120 mm/s, the maximum feedrate for real machining is set to be 7200 mm/min. The feedrate scheduling method implemented as a pre-process consumes about 8.881 s and 11.361 s for the hat and butterfly cases, respectively. The real machining test of the butterfly curve is shown in Fig. 15(b).

Some experiments are performed to do the comparative analysis among the ADA, the FS with and without real machining for the butterfly curve. Since the DBLA cannot meet the specification of maximum feedrate, only the ADA is used for comparison. Experimental results are shown in Fig. 16. Statistical data are summarized in Table 9. The ADA which only considers chord errors cause high jerks, large tracking and contour errors at the critical points of $C_1, C_5, C_8, C_{10}, C_{13}, C_{19}, C_{22}, C_{24}, C_{27}, C_{31}$, as shown in Fig. 16(b)–(d). The chord errors cannot be constrained by 1 μm using the ADA for high-speed machining as shown in Fig. 16(a). Nevertheless, the FS detects the critical points in advance, and plans a smoother feedrate profile under the given constraints of chord error, acceleration and jerk as shown in Fig. 16(f). Fig. 16(e)–(h) demonstrate that the proposed FS can improve the chord, tracking and contour performances simultaneously as compared with the ADA. As shown in Table 9, the total machining times for performing the FS with and without real machining are 19.411 s and 11.361 s, respectively. The tracking and contour errors are almost the same as expected between simulation and experimental results. The FS can reduce the maximum contour error by 76.52% as compared with the ADA, with a penalty of 464.27% more machining time. The experimental results demonstrate the feasibility of the proposed off-line feedrate scheduling method and the applicability for real-time implementation.

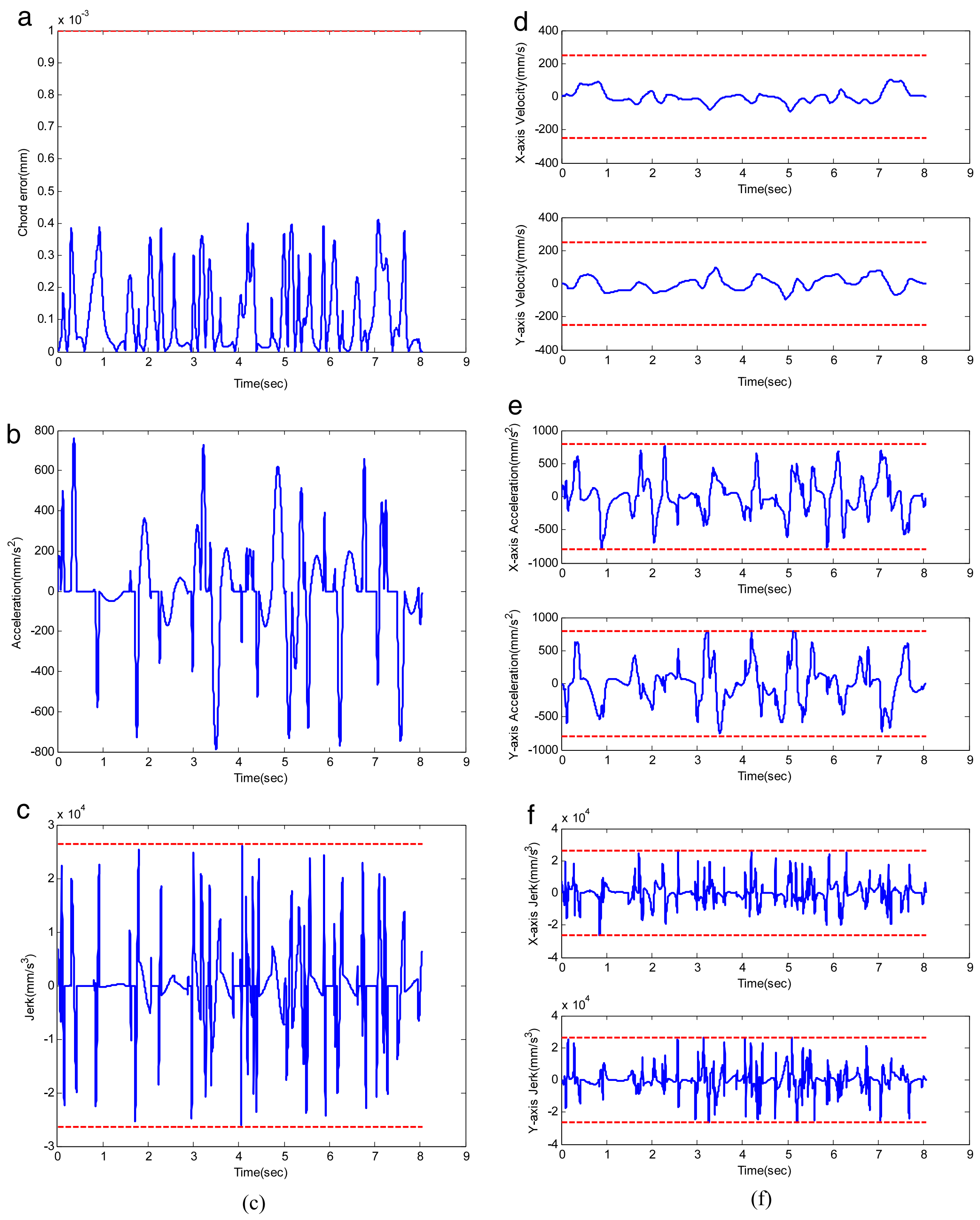
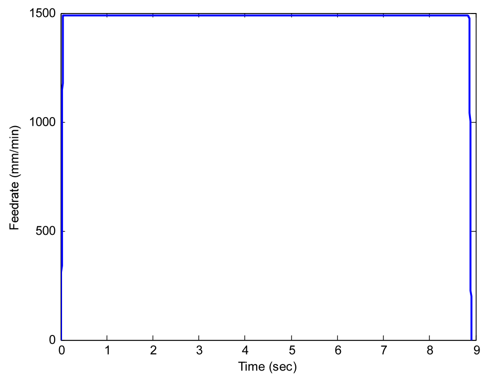


**Fig. 13.** Feedrate scheduling for the butterfly curve: (a) chord error profile, (b) acceleration profile, (c) jerk profile, (d) *x*- and *y*-axes velocity profiles, (e) *x*- and *y*-axes acceleration profiles and (f) *x*- and *y*-axes jerk profiles.

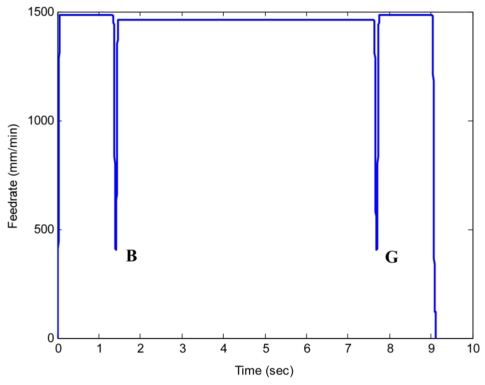
**Table 9**
Performance comparisons among different NURBS interpolation algorithms for butterfly contour.

| Feedrate command (mm/min) | Interpolation algorithm | Chord error (μm) | | Tracking error (μm) | | Contour error (μm) | | Machining time (s) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | MAX | RMS | MAX (*x*/*y*-axis) | RMS (*x*/*y*-axis) | MAX | RMS | PRE | MAC | Total |
| 7200 | ADA | 2.521 | 0.297 | 266.029/411.70 | 32.923/57.780 | 285.871 | 46.012 | 0 | 3.440 | 3.440 |
| | FS (SIM) | 0.411 | 0.155 | 116.736/186.611 | 18.945/21.252 | 58.000 | 18.638 | 11.361 | 0 | 11.361 |
| | FS (EXP) | 0.411 | 0.155 | 96.630/183.020 | 19.523/23.315 | 67.119 | 19.628 | 11.361 | 8.050 | 19.411 |

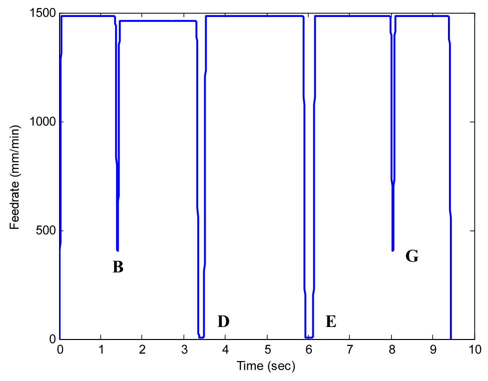
Note. SIM: simulation, EXP: experiment.



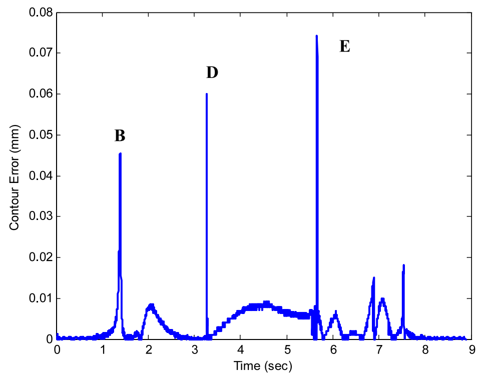
(a) ADA.



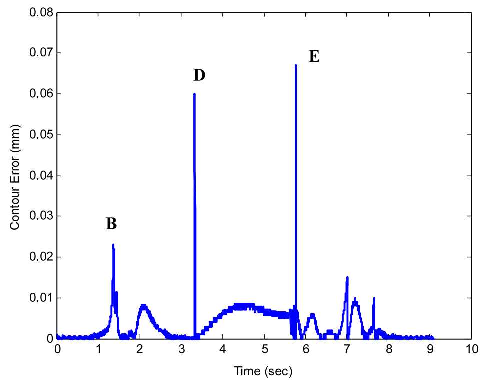
(b) DBLA.



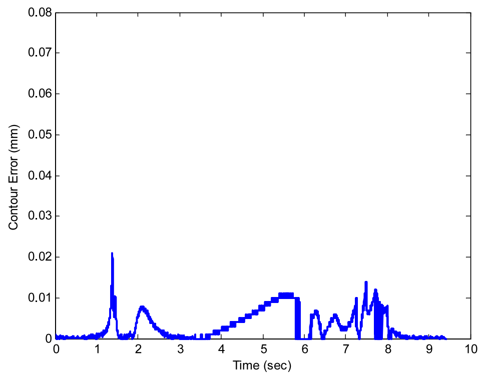
(c) FS.



(d) ADA.



(e) DBLA.



(f) FS.

**Fig. 14.** Simulation results of ADA, DBLA and FS interpolation algorithms for the hat curve.

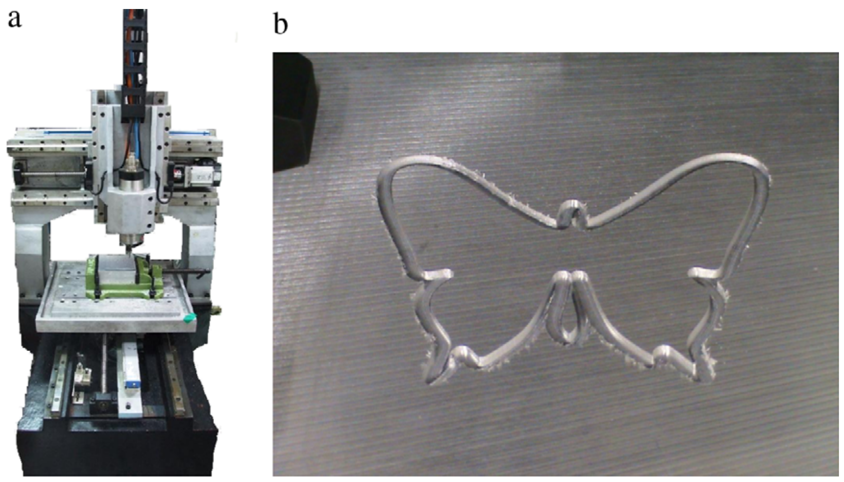


**Fig. 15.** The real machining test (a) three-axis engraving machine, (b) the butterfly contour.

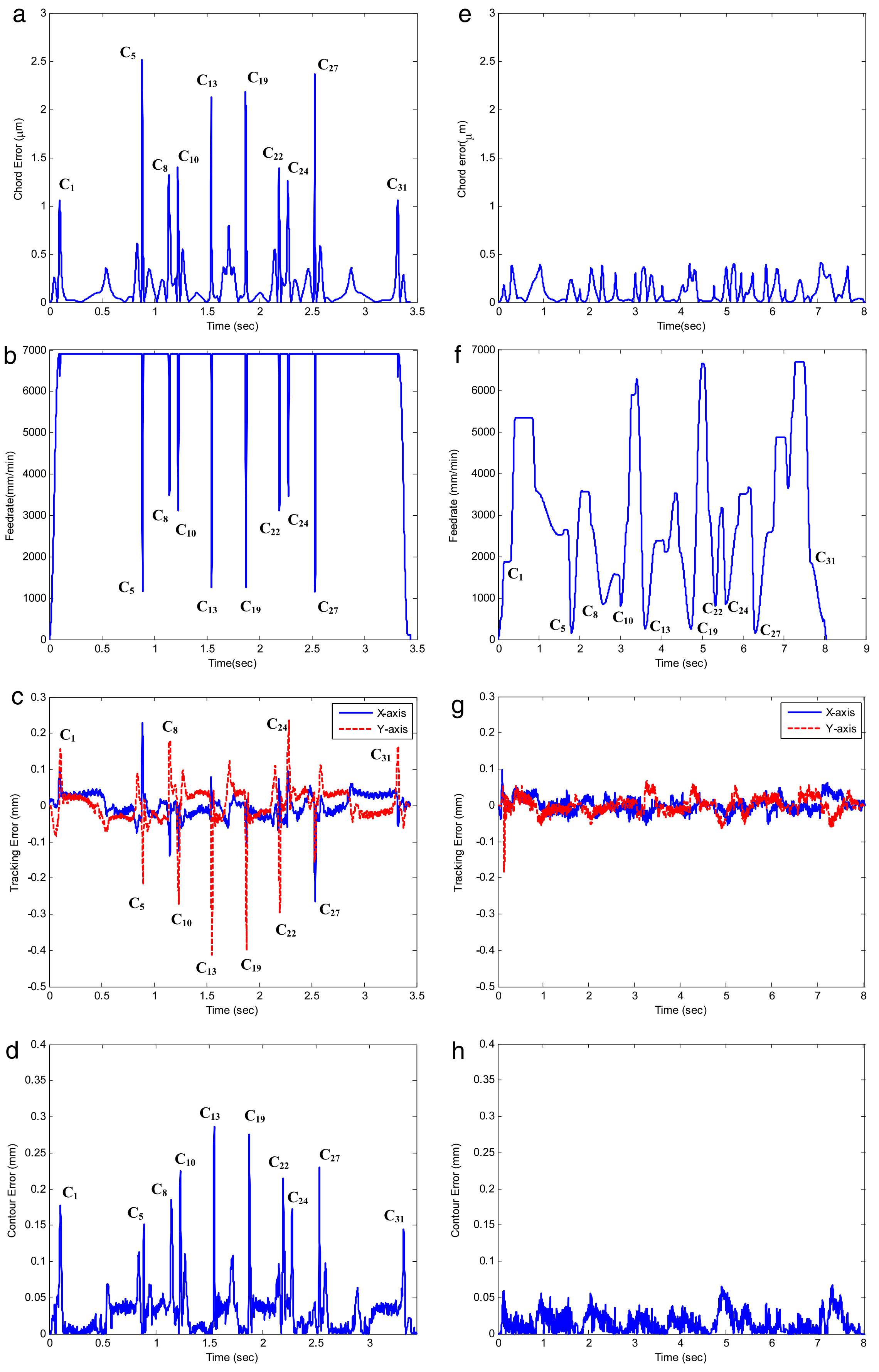


**Fig. 16.** Experimental results of ADA and FS interpolation algorithms for the butterfly curve.

## 6. Conclusions

In this study, an off-line feedrate scheduling method constrained by chord tolerance, acceleration and jerk limitations is proposed to construct a jerk-limited feedrate profile of NURBS curve. In off-line process, scanning a NURBS curve in advance is performed to find out the breakpoints with $G^0$ continuity, and the curve is split into several NURBS sub-curves. The proposed method also detects the critical points with large curvatures, and the curve is further divided into small blocks. Furthermore, the suitable feedrates at breakpoints and critical points are determined based on the constraints of chord tolerance, acceleration and jerk limitations, and velocity variation. The length of each NURBS block is estimated and stored as the scanning data. The feedrate profile for each NURBS block is constructed by utilizing the results of curve scanning. In addition, feedrate compensation method for short NURBS block is established to make the whole feedrate profile more continuous and precise. The proposed method is realized as a preprocessor, which avoids the complicated and heavy calculations in real time. In real-time process, it is simple and efficient to perform the interpolation algorithm and to generate the curve point. Finally, simulations and experiments are conducted to verify the feasibility and applicability of the proposed feedrate scheduling method.

## References

[1] Piegl L, Tiller W. The NURBS Books. second ed. Berlin, Heidelberg: Springer; 1997.
[2] Xu XW, Newman ST. Making CNC machine tools more open, interoperable and intelligent—a review of the technologies. Computers in Industry 2006;57(2):141–52.
[3] Xu XW. Realization of STEP-NC enabled machining. Robotics and Computer-Integrated Manufacturing 2006;22(2):144–53.
[4] Shipitalni M, Koren Y, Lo CC. Real-time curve interpolators. Computer-Aided Design 1994;26(11):832–8.
[5] Yang DCH, Kong T. Parametric interpolator versus linear interpolator for precision CNC machining. Computer-Aided Design 1994;26(3):225–34.
[6] Farouki RT, Tsai YF. Exact Taylor series coefficients for variable-feedrate CNC curve interpolators. Computer-Aided Design 2001;33(2):155–65.
[7] Tsai MC, Cheng CW. A real-time predictor–corrector interpolator for CNC machining. ASME Transaction Journal of Manufacturing Science and Engineering 2003;125(3):449–60.
[8] Erkorkmaz K, Altintas Y. Quintic spline interpolation with minimal feed fluctuation. ASME Transaction Journal of Manufacturing Science and Engineering 2005;127(2):339–49.
[9] Lei WT, Sung MP, Lin LY, Huang JJ. Fast real-time NURBS path interpolation for CNC machine tools. International Journal of Machine Tools and Manufacture 2007;47(10):1530–41.
[10] Yeh SS, Hsu PL. Adaptive-feedrate interpolation for parametric curves with a confined chord error. Computer-Aided Design 2002;34(3):229–37.
[11] Zhiming X, Jincheng C, Zhengjin F. Performance evaluation of a real-time interpolation algorithm for NURBS curves. International Journal of Advanced Manufacturing Technology 2002;20(4):270–6.
[12] Yong T, Narayanaswami R. A parametric interpolator with confined chord errors, acceleration and deceleration for NC machining. Computer-Aided Design 2003;35(13):1249–59.
[13] Sun Y, Jia Z, Ren F, Guo D. Adaptive feedrate scheduling for NC machining along curvilinear paths with improved kinematic and geometric properties. International Journal of Advanced Manufacturing Technology 2008;16(1–2):60–8.
[14] Nam SH, Yang MY. A study on a generalized parametric interpolator with real-time jerk-limited acceleration. Computer-Aided Design 2004;36(1):27–36.
[15] Liu X, Ahmad F, Yamazaki K, Mori M. Adaptive interpolation scheme for NURBS curves with the integration of machining dynamics. International Journal of Machine Tools and Manufacture 2005;45(4–5):433–44.
[16] Xu RZ, Xie L, Li CX, Du DS. Adaptive parametric interpolation scheme with limited acceleration and jerk values for NC machining. International Journal of Advanced Manufacturing Technology 2008;36(3–4):343–54.
[17] Lai JY, Lin KY, Tseng SJ, Ueng WD. On the development of a parametric interpolator with confined chord error, feedrate, acceleration and jerk. International Journal of Advanced Manufacturing Technology 2008;37(1–2):104–21.
[18] Lin MT, Tsai MS, Yau HT. Development of a dynamics-based NURBS interpolator with real-time look-ahead algorithm. International Journal of Machine Tools and Manufacture 2007;47(15):2246–62.
[19] Tsai MS, Nien HW, Yau HT. Development of an integrated look-ahead dynamics-based NURBS interpolator for high precision machinery. Computer-Aided Design 2008;40(5):554–66.
[20] Dong J, Stori JA. Optimal feed-rate scheduling for high-speed contouring. ASME Transaction Journal of Manufacturing Science and Engineering 2007;129(1):63–76.
[21] Tikhon XM, Ko TJ, Lee SH, Kim HS. NURBS interpolator for constant material removal rate in open NC machine tools. International Journal of Machine Tools and Manufacture 2004;44(2–3):237–45.
[22] Choi IH, Yang MY, Hong WP, Jung TS. Curve interpolation with variable feedrate for surface requirement. International Journal of Advanced Manufacturing Technology 2005;25(3–4):325–33.
[23] Fleisig RV, Spence AD. A constant feed and reduced angular acceleration interpolation algorithm for multi-axis machining. Computer-Aided Design 2001;33(1):1–15.
[24] Mohan S, Kweon SH, Lee DM, Yang SH. Parametric NURBS curve interpolators: a review. International Journal of Precision Engineering and Manufacturing 2008;9(2):84–92.
[25] Kincaid D, Cheney W. Numerical Analysis: Mathematics of Scientific Computing. 3th ed. Brooks Cole; 2002.
[26] FANUC Corporation. FANUC Series 30i/300i/300is-MODEL A. Series 31i/310i/310is-MODEL A5. Series 31i/310i/310is-MODEL A. Series 32i/320i/320is-MODEL A—Parameter Manual. 2004.
[27] Jeon JW, Ha YY. A generalized approach for the acceleration and deceleration of industrial robots and CNC machine tool. IEEE Transactions on Industrial Electronics 2000;47(1):133–9.
[28] Intel Corporation. Intel®64 and IA-32 Architectures—Optimization Reference Manual. 2009.
[29] Yau HT, Lin MT, Tsai MS. Real-time NURBS interpolation using FPGA for high speed motion control. Computer-Aided Design 2006;38(10):1123–33.