Int J Adv Manuf Technol (2013) 69:1373–1388
DOI 10.1007/s00170-013-5083-x

ORIGINAL ARTICLE

Time-optimal interpolation for five-axis CNC machining along parametric tool path based on linear programming

Wei Fan · Xiao-Shan Gao · Chen-Han Lee · Ke Zhang · Qiang Zhang

Received: 18 September 2012 / Accepted: 21 May 2013 / Published online: 19 June 2013

Abstract In this paper, the time-optimal velocity planning problem for five-axis computer numerical control machining along a given parametric tool path under chord error, acceleration, and jerk constraints is studied. The velocity planning problem under confined chord error, feedrate, and acceleration is reduced to an equivalent linear programming problem by discretizing the tool path and other quantities. As a consequence, a polynomial time algorithm with computational complexity $O(N^{3.5})$ is given to find the optimal solution, where N is the number of discretized segments of the tool path. The velocity planning problem under confined chord error, feedrate, acceleration, and jerk is reduced to a linear programming program by using a linear function to approximate the nonlinear jerk constraint. As a consequence, a polynomial time algorithm is given to find the approximate time-optimal solution. Simulation results are used to show the efficiency and effectiveness of the algorithms.

Keywords Time-optimal interpolation · Velocity planning · Parametric tool path · Jerk constraint · Linear programming algorithm · Polynomial time algorithm

W. Fan · X.-S. Gao (✉) · K. Zhang · Q. Zhang
KLMM, Institute of Systems Science, AMSS,
Chinese Academy of Sciences, Beijing 100190, China
e-mail: xgao@mmrc.iss.ac.cn

W. Fan · C.-H. Lee
School of Mechanical Science and Engineering,
Huazhong University of Science and Technology,
Wuhan 430074, China

Q. Zhang
China University of Petroleum, Qingdao 266555, China

1 Introduction

Interpolation and velocity planning algorithms, which determine how the machine tool moves along a given tool path, play a key role in high-speed and high-precision computer numerical control (CNC) machining and, hence, are widely studied in the literature. In order to achieve high-speed machining, the planned feedrate is required to be as large as possible. In order to achieve high-quality machining, it is required that the planned feedrate satisfies constraints such as confined feedrate, confined acceleration, confined jerk, confined chord error, etc. Therefore, time optimization velocity planning under confined feedrate, accelerations, jerk, and chord error along a given parametric tool path becomes a standard problem in CNC interpolation, which is not completely solved both theoretically or practically.

The existing velocity planning algorithms for parametric tool paths can be roughly divided into four classes: the phase space analysis methods [1–6], the direct sampling methods [7–16], the numerical optimization methods [22–26], and the critical point methods [17–21].

With the phase space analysis methods, closed-form optimal solutions were obtained using the concepts of velocity limiting curve and integration trajectory in the case of confined acceleration and chord error by Bobrow et al. [1], Shiller [2], Farouki et al. [3], Zhang et al. [4], and Yuan et al. [5]. In the case of confined jerk, Zhang et al. [6] gave a greedy algorithm based on the concept of velocity limiting surface [6]. The main drawback of this approach is that the computation costs could be high for complicated tool paths due to the need to solve high-degree algebraic equations, and time-optimal algorithms under confined jerk do not exist.

In the direct sampling methods, the velocity at every sampling time $kT, k = 0, 1, \ldots$ is computed based on certain

strategies. For instance, Bedi et al. [7] and Yang-Kong [8] used a constant feedrate, and Yeh and Hsu used a chord error bound to control the feedrate if needed and used a constant feedrate in other places [9]. These methods are simple and efficient. However, the machine acceleration capabilities were not considered. Feng et al. [13] and Sun et al. [14] proposed a method to generate chord error and acceleration-limited velocity. Li et al. [29] provided a variable period feed interpolation algorithm for five-axis machining, but the jerk constraint of machine tool was not considered. Xu [28], Park [30], and Wang et al. [31] presented real-time interpolation methods for nonuniform rational basis spline (NURBS) paths. However, the acceleration ability of each axis was not considered. Nam and Yang [10] proposed a recursive method to generate jerk-limited velocity. Emami and Arezoo [11] and Lai et al. [12] proposed time-optimal velocity planning methods with confined acceleration, jerk, and chord error by adjusting the velocity if any of the bounds is violated through backtracking. In [15], Beudaert et al. improved the above procedure by searching the backtracking point with dichotomy when any of the bounds is violated. In [16], the idea of phase space analysis is adopted to plan the velocity under confined acceleration and chord error, and the computational complexity of the algorithm is $O(M)$ where M is the size of the discretization grid.

In the numerical optimization methods, the velocity planning problem is discretized as a nonlinear optimization problem which is solved with standard numerical methods. Nonlinear optimization-based interpolation methods under confined jerk and chord error were given by Erkorkmaz-Altintas [22] and Sencer-Altintas-Croft [24]. Dong et al. [23] gave a discrete greedy algorithm under confined feedrate, acceleration, and jerk based on a series of single variable optimization subproblems. Gasparetto et al. [26] proposed to use a linear combination of the machining time and the total jerk as the objective function in order to minimize vibration. Numerical optimization methods are very general and powerful, but solving nonlinear programming problems is generally time-consuming, and the obtained solutions are not guaranteed to be globally optimal.

In the critical point methods, critical points of the tool path with extremal curvatures are identified, and maximum feedrates at the critical points are determined according to chord error or acceleration constraints. Furthermore, a feedrate function for each tool path segment between two critical points is planned with various velocity profiles such as S-shape profiles by Narayanaswami and Yong [17], trigonometric profiles by Lee et al. [21], and jounce confined profiles by Fan et al. [20]. Lin et al. [18] and Tsai et al. [19] further introduced dynamics constraints. This approach is very practical, but it is not time-optimal, and the chord error is guaranteed only at the critical points.

In this paper, the time-minimum velocity planning problem for five-axis CNC machining along a given parametric tool path $\eta(u), u \in [0, 1]$ under confined feedrate, chord error, acceleration, and jerk is studied. The numerical optimization approach is adopted, but instead of nonlinear optimization methods, the linear programming method is used to solve the velocity planning problem. The main contribution of this paper is to give computationally efficient time-minimum velocity planning algorithms whose computational complexity is $O(N^{3.5})$ in terms of floating point operations, where N is the number of discretized segments of the tool path.

By properly discretizing the differential quantities as finite differences, the velocity planning problem under confined acceleration and chord error is reduced to an equivalent linear programming problem. It is proved that the linear programming problem has a unique solution which is the solution to the original velocity planning problem. As a consequence, a polynomial time algorithm with complexity $O(N^{3.5})$ is given to find the optimal solution, where N is the number of discretized segments of the tool path.

In the case of confined jerk, the nonlinear jerk constraint is replaced with a stronger linear constraint under which the jerk constraint is still valid, and the velocity planning problem under confined jerk, acceleration, and chord error is also reduced to a linear programming program problem. Simulation results show that the solution to the linear programming problem gives nice approximate solutions to the velocity planning problem.

Comparing with the phase space analysis methods, the proposed method is more efficient, and the computational complexity is the same for any tool path, making the method efficient for velocity planning along complex tool paths. Comparing with the other three types of methods, the proposed method can generate global optimal or approximate global optimal solution with a nice computational complexity bound. Comparing with the efficient method given in [16], the new method can handle jerk constraints. Overall, the major advantage of the method is that global optimal or approximate global optimal (in the case of confined jerk) solution for any tool path can be generated with guarantied computational complexity, which makes the method a practical one for CNC velocity planning. Simulation data are used to validate the abovementioned claims.

The rest of this paper is organized as follows. In Section 2, the kinematic constraints and chord error constraints of CNC machining are presented. In Section 3, velocity planning under confined acceleration and chord error is studied. In Section 4, velocity planning under confined jerk, acceleration, and chord error is studied. In Section 5, simulation results for tool paths from real CNC models are used to show the effectiveness of the algorithms. In Section 6, the conclusion is presented.

2 Kinematic and chord error constraints

In this section, the kinematic and chord error constraints of CNC machining will be presented, including maximal feedrate, maximal accelerations of the five axes, and maximal chord error. The jerk constraint will be given in Section 4.

2.1 The acceleration constraints

The tool path of the five-axis CNC machine is given by a set of parametric functions with at least C^1 continuity $\eta(u) = (x(u),\ y(u),\ z(u),\ a(u),\ c(u))^T, (u \in [0,1])$, where $r(u) = (x(u), y(u), z(u))$ is the machining path, and $a(u)$ and $c(u)$ are rotary angles around the X-axis and Z-axis, respectively. The parametric functions could be NURBS, B-spline curves, etc. Note that these coordinates are in the machine coordinate system (MCS).

In this paper, we use "$\cdot$" to denote "$\frac{d}{dt}$" and use "$'$" to denote "$\frac{d}{du}$." Denote the machining velocity to be $\mathbf{v} = (v_x, v_y, v_z)$ and the tangential feedrate to be $v = \sqrt{v_x^2 + v_y^2 + v_z^2}$. Let s be the arc length of $r(u)$ and $\sigma(u) = ds/du = \sqrt{x'(u)^2 + y'(u)^2 + z'(u)^2}$ the parametric speed. Introduce the following important quantity

$$q = \left(\frac{v}{\sigma}\right)^2, \tag{1}$$

which will be used as the optimization variable in our velocity planning problem. As will be shown later, the acceleration constraints can be written as linear inequalities about q and its derivative, which is one of the reasons allowing us to reduce the velocity planning to a linear programming problem. Note

$$\dot{u} = \frac{du}{dt} = \frac{du}{ds}\frac{ds}{dt} = \frac{v}{\sigma} = \sqrt{q}. \tag{2}$$

For the X-axis,

$$v_x = \dot{x} = \frac{dx}{du}\frac{du}{dt} = x'\sqrt{q} \tag{3}$$

$$a_x = \dot{v_x} = \frac{dv_x}{dx}\frac{dx}{dt} = v_x\frac{dv_x}{dx} = \frac{1}{2}\frac{dv_x{}^2}{dx} = \frac{1}{2}\frac{dqx'^2}{dx} \tag{4}$$

For CNC machines, the maximal acceleration limits of each axis, reflecting the ability to accelerate, are important parameters. The following formulas are the acceleration constraints of the five axes:

$$|a_\tau(u)| \le A_\tau, \tau \in \{x, y, z, a, c\}, u \in [0, 1]. \tag{5}$$

where A_τ is the maximal acceleration of τ-axis. From Eq. 4, the axis acceleration constraints (5) are equivalent to the following inequalities about q:

$$\left|\frac{1}{2}\frac{d\left(q\tau'^2\right)}{d\tau}\right| \le A_\tau, \tau \in \{x, y, z, a, c\}, u \in [0, 1]. \tag{6}$$

2.2 Feedrate and chord error constraint for five-axis CNC machines

Different from three-axis CNC machines, for five-axis CNC machines, the feedrate and chord error are measured in the workpiece coordinate system (WCS) instead of the MCS. Therefore, a coordinate transformation between WCS and MCS is needed. Such a transformation depends on the structure of the CNC machines. The widely used table-tilting (Fig. 1) five-axis CNC machine is adopted in this paper. Other types of five-axis CNC machines can be treated similarly.

In a table-tilting machine shown in Fig. 1, C is the workbench rotary angle around the Z-axis, and A is the rotary angle around the X-axis. Denote $r_w(u) = (x_w(u), y_w(u), z_w(u))$ as the coordinates of tool path curve $r(u)$ in WCS. The initial coordinate of the origin of MCS is denoted as $O_w = (x_{w_0}, y_{w_0}, z_{w_0})$.

Let $Rot(a, x)$ and $Rot(c, z)$ be the rotary matrices of rotating angle a around X-axis and angle c around Z-axis, respectively, which have following forms:

$$Rot(a, x) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$Rot(c, z) = \begin{pmatrix} \cos c & -\sin c & 0 & 0 \\ \sin c & \cos c & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Denote the translation matrix to be $Tran(O_w)$ which has the form

$$Tran(O_w) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_{w_0} & y_{w_0} & z_{w_0} & 1 \end{pmatrix}.$$

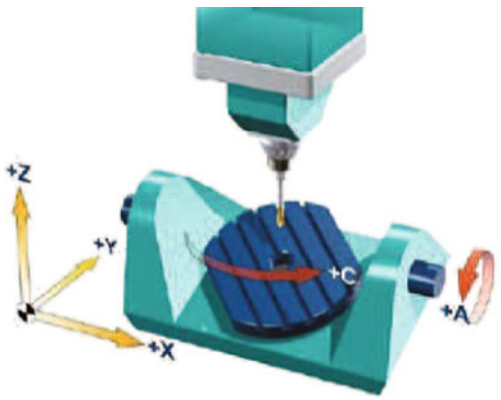


Fig. 1 The table-tilting five-axis machine

The relationship between the coordinates (x_w, y_w, z_w) in WCS and the coordinate (x, y, z) in MCS is given below.

$$(x_w \ y_w \ z_w \ 1) = (x \ y \ z \ 1) \cdot Tran(O_w) \cdot Rot(a, x) \cdot Rot(c, z)$$

which can be written as follows:

$$\begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} = \begin{pmatrix} \cos c & \cos a \sin c & \sin a \sin c \\ -\sin c & \cos a \cos c & \sin a \cos c \\ 0 & -\sin a & \cos a \end{pmatrix} \cdot \begin{pmatrix} x + x_{w_0} \\ y + y_{w_0} \\ z + z_{w_0} \end{pmatrix} \tag{7}$$

By Eq. 7, the coordinate transformation from MCS to WCS is obtained.

Let $\mu = (x_w, y_w, z_w)^T$ and $\eta = (x, y, z, a, c)^T$. Differentiate two sides of the Eq. 7 and obtain the following differential relationship

$$\frac{d\mu}{du} = M_t \cdot \frac{d\eta}{du} \tag{8}$$

where

$$M_t = \begin{pmatrix} \cos c & \cos a \sin c & \sin a \sin c & z_w \sin c & y_w \\ -\sin c & \cos a \cos c & \sin a \cos c & z_w \cos c & -x_w \\ 0 & -\sin a & \cos a & -(y + y_{w_0}) \cos a - (z + z_{w_0}) \sin a & 0 \end{pmatrix}.$$

Differentiate two sides of Eq. 8 to obtain the relationship for the second-order derivatives:

$$\frac{d^2\mu}{du^2} = \frac{dM_t}{du} \cdot \frac{d\eta}{du} + M_t \cdot \frac{d^2\eta}{du^2} \tag{9}$$

where

$$\frac{dM_t}{du} = \begin{pmatrix} -c' \sin c & -c' \cos c & 0 \\ -a' \sin a \sin c + c' \cos a \cos c & -a' \sin a \cos c - c' \cos a \sin c & -a' \cos a \\ a' \cos a \sin c + c' \sin a \cos c & a' \cos a \cos c - c' \sin a \sin c & -a' \sin a \\ c' z_w \cos c + z'_w \sin c & -c' z_w \sin c + z'_w \cos c & -y' \cos a - z' \sin a - z_w a' \\ y'_w & -x'_w & 0 \end{pmatrix}^T$$

where x'_w, y'_w, z'_w can be determined from Eq. 8.

Now, we can give the feedrate and chord error constraints. The maximal feedrate along the tool path is an important CNC parameter reflecting the machining ability. For a given maximal feedrate V_m, the feedrate v_w must satisfy the following condition:

$$v_w(u) \le V_m, u \in [0, 1]. \tag{10}$$

Let $\sigma_w = \sqrt{(x'_w)^2 + (y'_w)^2 + (z'_w)^2}$. Then

$$v_w(u) = \sigma_w(u)\dot{u} = \sigma_w(u)\sqrt{q(u)}. \tag{11}$$

By Eq. 8, the feedrate constraint $v_w(u) \le V_M$ is reduced to the following constraint about q:

$$q \le \frac{V_M^2}{\sigma_w^2} = \frac{V_M^2}{(|M_t \cdot \eta'|)^2}. \tag{12}$$

Chord error is the error caused by machining the tool path within one sampling period. Only the start point and end point for one sampling period are given to the CNC machine, and the actual machining path is different from the tool path. Chord error is used to measure the difference between these two paths and is defined to be the distance between the line segment connecting the start point and end point and the tool path.

Since one sampling period is very small, the tool path during one sampling period can be approximated as a piece of circle arc. If the sampling period is T and and tool path is $r_w(u) = (x_w(u), y_w(u), z_w(u))$ in WCS, then the chord error $\delta_w(u)$ has the following well-known relationship with velocity $v_w(u)$ in WCS [9, 11, 12]:

$$\delta_w(u) = \rho_w(u) - \sqrt{\rho_w(u)^2 - \frac{v_w(u)^2 T^2}{4}}$$

where $\rho_w(u)$ is the curvature radius of the tool path in WCS and $\rho_w(u) = \frac{|r'_w(u)|^3}{|r'_w(u) \times r''_w(u)|}$. Let δ_m be the maximal chord error. So the chord error constraint $\delta_w(u) \le \delta_m$ becomes

$$\rho_w(u) - \sqrt{\rho_w(u)^2 - \frac{v_w(u)^2 T^2}{4}} \le \delta_m.$$

That is

$$v_w(u) \le \frac{\sqrt{8\rho_w(u)\delta_m - 4\delta_m^2}}{T}.$$

Since δ_m is a tiny quantity compared with $\rho_w(u)$, we can omit the second-order small quantity δ_m^2 to obtain

$$v_w(u) \le \frac{\sqrt{8\delta_m \rho_w(u)}}{T}. \tag{13}$$

Due to Eq. 11, 13 is reduced to

$$q \le \frac{8\delta_m \rho_w(u)}{\sigma_w^2 T^2} = \frac{8\delta_m \sigma_w}{|r'_w(u) \times r''_w(u)|T^2}. \tag{14}$$

From Eqs. 8, 9, and 14, the chord error constraint $\delta_w(u) \le \delta_m$ can be written as the following inequality about q:

$$q \le \frac{8\delta_m |M_t \eta'|}{|M_t \eta' \times (M'_t \eta' + M_t \eta'')|T^2}. \tag{15}$$

3 Optimal velocity planning under acceleration constraint

In this section, we give a polynomial time algorithm to find the time-optimal solution to the velocity planning problem under acceleration and chord error constraints.

3.1 Formulation of the problem

In this section, the velocity planning problem is formulated as an optimal problem under differential constraints.

At the start and end points of the tool path, the feedrates are assumed to be zero, that is

$$q(0) = q(1) = 0. \tag{16}$$

The machining time is $t = \int_0^1 \frac{du}{\dot{u}} = \int_0^1 \frac{du}{\sqrt{q}}$. So, the time-optimal velocity planning problem under the confined feedrate, acceleration, and chord error can be described as the following form:

$$\min_q \int_0^1 \frac{du}{\sqrt{q}} \tag{17}$$

subject to constraints (6), (12), and (15). From Eq. 1, after q is obtained, the optimal feedrate can be obtained as $v(u) = \sigma(u)\sqrt{q(u)}$.

The above optimal problem has a very important property given in the following theorem, which is proved in many cases and using different methods [1–3, 5, 6, 25].

Theorem 3.1 *The optimal feedrate $v_o(u), u \in [0, 1]$ of problem (17) is maximum at any parameter value. That is, let $v_f(u)$ be another feedrate satisfying constraints (16), (12), (6), and (15). Then, $v_f(u) \le v_o(u)$ for all $u \in [0, 1]$. Furthermore, the optimal feedrate is bang-bang-singular in the sense that for any parameter u, the equality sign holds at least in one of the three constraints (12), (6), and (15).*

From Theorem 3.1, it is easy to see that the optimal solution to problem (17) is unique.

3.2 Discrete form of the optimization problem

The three-axis version for the above optimization problem can be solved analytically as shown in [1–3, 5]. Although the analytical solution approach can be theoretically extended to the five-axis case, it is difficult to obtain a practically effective method due to the complicated expression introduced in constraint (15). Therefore, to develop efficient numerical method is inevitable for the five-axis CNC velocity planning with chord error constraint. In order to use numerical methods to solve problem (17), in this section, we give a discrete version for the problem.

We divide the parametric interval $[0, 1]$ into N equal parts at knots $u_i = \frac{i}{N}$, $i = 0, \ldots, N$. The length of each subinterval is $\Delta = 1/N$. Since the velocities at the two endpoints of tool path are zero, we have $q_0 = q_N = 0$. Since Δ is very small, the constraints (12), (6), and (15) can be approximately transformed into the following linear inequalities:

$$0 \le q_i \le \frac{V_M^2}{\left(\left|M_{ti} \cdot \eta'_i\right|\right)^2} \tag{18}$$

$$\left|q_{i+1}\tau_{i+1}'^2 - q_i \tau_i'^2\right| \le 2A_\tau \left|\tau_{i+1} - \tau_i\right|, \quad \tau \in \{x, y, z, a, c\} \tag{19}$$

$$0 \le q_i \le \frac{8\delta_m \left|M_{ti}\eta'_i\right|}{\left|M_{ti}\eta'_i \times \left(M'_{ti}\eta'_i + M_{ti}\eta''_i\right)\right| T^2}, \tag{20}$$

where $q_i = q(u_i)$, $\sigma_i = \sigma(u_i)$, $\tau_i = \tau(u_i)$, $\tau'_i = \tau'(u_i)$, and the vectors η'_i, M_{t_i} and M'_{t_i} are the values of $\eta'(u)$, $M'_t(u)$ and $M_t(u)$ at u_i, respectively. Correspondingly, the optimization problem (17) is transformed into the following *nonlinear programming problem* with objective function

$$\min_{\{q_i\}} \frac{1}{N} \sum_{i=1}^{N-1} \frac{1}{\sqrt{q_i}} \tag{21}$$

and constraints (16), (18), (19), and (20).

Instead of solving problem (21), we will solve the following *linear programming problem* with objective function

$$\max_{\{q_i\}} \sum_{i=1}^{N-1} q_i \tag{22}$$

and constraints (16), (18), (19), and (20).

In the following, it is shown that the solution to the linear programming problem (22) provides an approximate solution to the original problem (17). The key idea is to show that the optimal solution q_i^*, $(i = 1, \ldots, N-1)$ to the linear programming problem (22) is unique, and each

q_i^* achieves the maximal value among all possible feasible solutions. Combing this property and Theorem 3.1, the claim can be proven.

Theorem 3.2 *The unique solution to problem* (17) *can be sufficiently approximated with the unique solution of the linear programming problem* (22) *when N is large enough.*

Proof The feasible region for $q_i = 0, (i = 1, \ldots, N-1)$ determined by constraints (18), (19), and (20) is clearly a finite compact set. Then, the linear programming program always has a solution $Q^* = \{q_i^*, (i = 1, \ldots, N-1)\}$.

We claim that for each i, q_i^* is maximal among all feasible solutions. Assume the contrary. Then there exists another feasible solution $\{q_k, (k = 1, \ldots, N-1)\}$ such that $q_j > q_j^*$ holds for at least one j. Since for a linear programming problem, the optimal value for the objective function is unique, there must exist a k, such that $q_k < q_k^*$. Notice that $q_0 = q_0^* = q_N = q_N^* = 0$. Then, there exists $i < j$ such that $q_i^* \geq q_i$ and $q_k^* < q_k$, $(k = i+1, \ldots, j-1)$, and $q_j^* \geq q_j$. Since q_i, q_{i+1}, and q_i^*, q_{i+1}^* satisfy the constraint (19), we have

$$\left|q_{i+1}\tau_{i+1}'^2 - q_i\tau_i'^2\right| \leq 2A_\tau\,|\tau_{i+1} - \tau_i|\,,$$
$$\left|q_{i+1}^*\tau_{i+1}'^2 - q_i^*\tau_i'^2\right| \leq 2A_\tau\,|\tau_{i+1} - \tau_i|\,.$$

Let $a_1 = q_{i+1}\tau_{i+1}'^2$, $a_2 = q_i\tau_i'^2$, $b_1 = q_{i+1}^*\tau_{i+1}'^2$, $b_2 = q_i^*\tau_i'^2$, $B = 2A_\tau|\tau_{i+1} - \tau_i|$. So, $|a_1 - a_2| \leq B$, $|b_1 - b_2| \leq B$. By the assumption $q_i^* \geq q_i$ and $q_{i+1}^* < q_{i+1}$, we have $a_1 > b_1$ and $a_2 \leq b_2$. Then

$$a_1 - b_2 = a_1 - b_1 + b_1 - b_2 \geq b_1 - b_2 \geq -B,$$
$$a_1 - b_2 = a_1 - a_2 + a_2 - b_2 < a_1 - a_2 \leq B.$$

Thus, $|a_1 - b_2| \leq B$. That is

$$\left|q_{i+1}\tau_{i+1}'^2 - q_i^*\tau_i'^2\right| \leq 2A_\tau\,|\tau_{i+1} - \tau_i|\,.$$

Therefore, q_i^* and q_{i+1} satisfy the constraint (19). Similarly, it can be shown that q_{j-1} and q_j^* also satisfy the constraint (19). Since constrains (18) and (20) are automatically satisfied, by replacing q_k^*, $(k = i+1, \ldots, j-1)$ with q_k, $(k = i+1, \ldots, j-1)$, we obtain a new feasible solution of the linear programming problem, which has a larger value for the objective function, leading a contradiction. Thus, the claim is proven.

From the claim, the linear programming problem (22) has a unique solution $q_i^*, (i = 0, \ldots, N)$. Furthermore, since the solution q_i^* is maximal for each i among all feasible solutions, a solution to problem (22) is also a solution to problem (21). And the objective function (21) is the discrete form of (17). Therefore, the solution to linear problem (22) can approximate the solution to problem (17) as good as possible when N becomes large enough. □

3.3 Optimization velocity planning based on linear programming

In the preceding section, we show that the problem of velocity planning is transformed into a linear optimization problem. In this section, the algorithm is given.

In order to use the standard solver from MatLab, we rewrite the linear programming problem (22) as the following standard form.

$$\max_Q cQ \tag{23}$$

with constraints $MQ \leq R$ and $Q \geq 0$, where $Q = (q_1, \ldots, q_{N-1})^T$, $q_0 = q_N = 0$, the coefficient vector c is $(1, \ldots, 1)$ with size $1 \times (N-1)$, R and M will be given below.

Denote $h(u) = \min\left\{\frac{V_m^2}{\sigma_w(u)^2}, \frac{8\delta_m|M_t\eta'(u)|}{|M_t\eta'(u)\times(M_t'\eta'(u)+M_t\eta''(u))|T^2}\right\}$ and $\lambda_i = \left(x_i'^2, y_i'^2, z_i'^2, a_i'^2, c_i'^2\right)^T$, where $x_i' = x'(u_i)$. Let $\pi_i = \left(2A_x|x_{i+1} - x_i|, 2A_y|y_{i+1} - y_i|, 2A_z\,|\,z_{i+1} - z_i\,|, 2A_a|a_{i+1} - a_i|, 2A_c|c_{i+1} - c_i|\right)$. Then $R = (\pi_0, \pi_0, h(u_0), \ldots, \pi_{N-2}, \pi_{N-2}, h(u_{N-2}), \pi_{N-1}, \pi_{N-1})^T$ whose size is $(11N-1) \times 1$. The coefficient matrix M has the following form whose size is $(11N-1) \times (N-1)$.

$$M = \begin{pmatrix}
\lambda_1 & 0 & 0 & \ldots & 0 & 0 \\
-\lambda_1 & 0 & 0 & \ldots & 0 & 0 \\
1 & 0 & 0 & \ldots & 0 & 0 \\
-\lambda_1 & \lambda_2 & 0 & \ldots & 0 & 0 \\
\lambda_1 & -\lambda_2 & 0 & \ldots & 0 & 0 \\
0 & 1 & 0 & \ldots & 0 & 0 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
0 & 0 & 0 & \ldots & -\lambda_{N-2} & \lambda_{N-1} \\
0 & 0 & 0 & \ldots & \lambda_{N-2} & -\lambda_{N-1} \\
0 & 0 & 0 & \ldots & 0 & 1 \\
0 & 0 & 0 & \ldots & 0 & -\lambda_{N-1} \\
0 & 0 & 0 & \ldots & 0 & \lambda_{N-1}
\end{pmatrix} \tag{24}$$

After solving the linear programming problem (23), we obtain a set of optimal solution $Q^* = \left(q_1^*, \ldots, q_{N-1}^*\right)^T$ with $q_0^* = q_N^* = 0$. From Eq. (1), the optimal feedrate can be obtained $v_i^* = v(u_i) = \sigma(u_i)\sqrt{q_i^*}, (i = 0, \ldots, N)$. We can use standard techniques to fit the discrete values v_i to obtain the optimal feedrate function $v^*(u)$, $u \in [0, 1]$ [16].

Based on the above analysis, the following velocity planning algorithm is given.

Algorithm 3.3 ***VPA_LP***

Input: *Five-axis tool path* $\eta(u)$,$u \in [0, 1]$ *in MCS, the sampling period* T*, the maximal feedrate* V_m*, five-axis maximal acceleration bounds* $(A_x, A_y, A_z, A_a, A_c)$*, the chord*

error bound δ_m, the number of discretization N, and the initial position of the origin of MCS $O_w = (x_{w_0}, y_{w_0}, z_{w_0})$.

Output: *The approximate optimal velocity function $v(u), u \in [0, 1]$ of problem (17).*

1. *Compute $\eta_i = \eta(u_i)$, λ_i, and π_i defined in this section, where $u_i = \frac{i}{N}$, $i = 0, \ldots, N$ to obtain the coefficient matrix M in (24).*
2. *Compute the error limit function*

$$h(u_i) = \min\left\{\frac{V_m^2}{|M_t(u_i)\eta'(u_i)|^2}, \frac{8\delta_m|M_t(u_i)\eta'(u_i)|}{|M_t(u_i)\eta'(u_i) \times (M_t'(u_i)\eta'(u_i) + M_t(u_i)\eta''(u_i))|T^2}\right\}, \quad i = 1, \ldots, N-1.$$

Obtain the right-hand side vector

$$R = (\pi_0, \pi_0, h(u_1), \ldots, \pi_{N-2}, \pi_{N-2}, h(u_{N-1}), \pi_{N-1}, \pi_{N-1})^T$$

3. *Solve the linear programming problem (23) to obtain the optimal solution $Q^* = \left(q_1^*, \ldots, q_{N-1}^*\right)^T$. Let $q_0^* = q_N^* = 0$.*
4. *Compute the velocity function. For instance, in the simplest case, we can give a piecewise linear representation as follows:*

$$v(u) = v_i^* \frac{u - u_{i+1}}{u_i - u_{i+1}} + v_{i+1}^* \frac{u_i - u}{u_i - u_{i+1}}$$

where $v_i^* = \sigma(u_i)\sqrt{q_i^*}$ and $u \in [u_i, u_{i+1}], i = 0, \ldots, N-1$. In order to use a smaller number of functions to represent the velocity curve, quadratic B-splines can be used to fit the sequence of velocity values with the method given in [4].

A nice feature to Algorithm 3.3 is that we can give its worst case computational complexity, which is rare for most velocity planning algorithms.

Theorem 3.4 *For a given N, the worst case computational complexity for Algorithm 3.3 is $O\left(N^{3.5}\right)$ in terms of floating point arithmetic operations.*

Since the dominate step of Algorithm 3.3 is step 3, we need only to estimate the complexity of this step. Note that the number of variables and the number of constraints for the linear programming problem (23) are both $O(N)$. Based on Karmarkar's famous algorithm [32] to solve linear programming problems, if using floating point number computations, the algorithm requires $O(N)$ steps, and each step requires $O(N^{2.5})$ arithmetic operations. Thus, we have Theorem 3.4.

In step 1 and step 2 of Algorithm 3.3, we need only to compute the values of certain formulas about the tool path $\eta(u)$ at the parameter values $u_k, k = 0, \ldots, N$. Suppose that $\eta(u)$ is a B-spline curve of degree d. Using Horner's trick, it is easy to show that the computation complexity for this procedure is $O(dN)$. Since d is very small comparred to N, the computational complexity of the algorithm can be considered the same for any such tool paths. This is a nice feature compared to the phase analysis methods [1–6] which involve integrations and nonlinear algebraic equation solving, and the equations to be solved depend on $\eta(u)$. As a consequence, the computational complexity of these methods also depends on $\eta(u)$.

3.4 An illustrative example

An illustrative example is given, where the tool path is a set of cubic polynomial curve:

$$\eta(u) = \left(15u^3, 10u^2, 20u, 2u^2 + 5u - 68, 7.5u^2 + 2u - 27\right), u \in [0, 1].$$

The path curve and the curve of two rotary axes are illustrated, respectively, in Figs. 2 and 3.

The CNC parameters are given as follows: $A_x = 1{,}000$ mm/s^2, $A_y = 1{,}000$ mm/s^2, $A_z = 1{,}000$ mm/s^2, $A_a = 500°$/s^2, $A_c = 500°$/s^2, $V_m = 110$ mm/s, $\delta_m = 0.05$ μm, $T = 1$ ms, $O_w = (1, 1, 1)$, $N = 200$.

The velocity curve obtained with Algorithm **VPA_LP** and the theoretical chord error of the velocity curve are shown in Figs. 4 and 5, respectively, where theoretical chord error is calculated by

$$\delta(u) = \frac{q(u)\left|M_t\eta' \times \left(M_t'\eta' + M_t\eta''\right)\right| T^2}{8\left|M_t\eta'\right|}. \tag{25}$$

The acceleration curves of X-, Y-, and Z-axes for this velocity curve are shown in Fig. 6a. The acceleration curves of two rotary axes are illustrated in Fig. 6b.

In Figs. 4, 5, and 6, it is easy to see that the planned velocity function is bang-bang-singular, meaning that at least one of accelerations for the five axes, the feedrate, or the chord error reaches its boundary value at any time. The whole machining process is divided into five phases. When $u \in [0, 0.155]$, the acceleration of Z-axis reaches a maximum value of 1,000 mm/s^2. When $u \in [0.155, 0.71]$, the chord error reaches maximum value. When $u \in [0.71, 0.905]$, the feedrate reaches maximal velocity V_m. When $u \in [0.905, 0.925]$, the acceleration of C-axis reaches minimum value $-500°$/s^2. Finally, when $u \in [0.925, 1]$, the acceleration of X-axis reaches minimum value $-1{,}000$ mm/s^2.

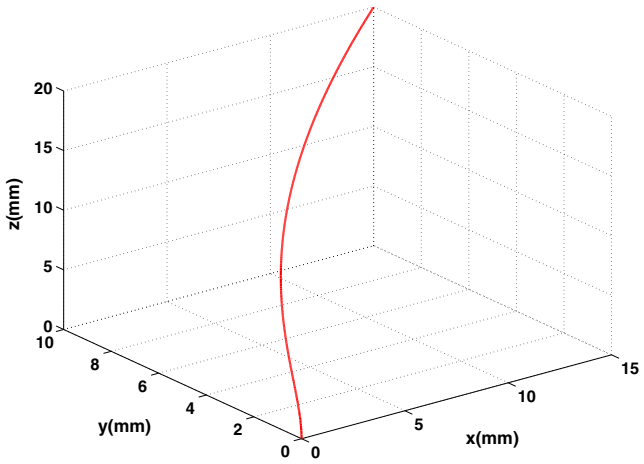


Fig. 2 The tool path curve

From Theorem 3.1, the bang-bang-singular control is a necessary condition of this optimal problem. The fact that our solutions are bang-bang-singular implies that these solutions provide nice approximation to the optimal solution of the original problem.

However, in Fig. 4, it is not difficult to find that the feedrate function is not smooth at some points such as $u = 0.155$, $u = 0.71$, and $u = 0.905$. At these points, the large vibration of the machine tool could lead to poor machining quality. This problem is addressed in the next section by introducing jerk bounds.

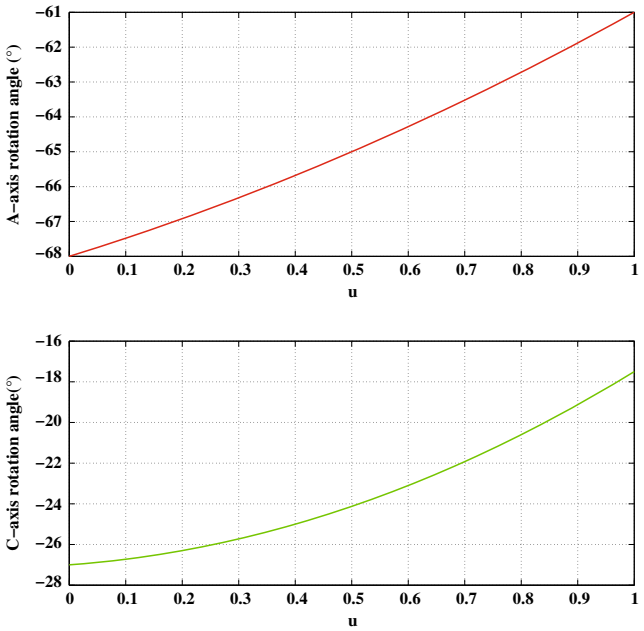


Fig. 3 The A-axis and C-axis curve

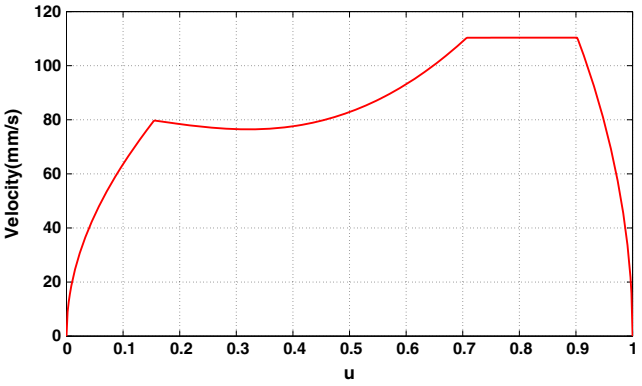


Fig. 4 The planned velocity

4 Velocity planning under jerk constraint with linear programming

In this section, we will consider time-optimal velocity planning under jerk constraints and give an approximate algorithm based on linear programming.

4.1 Formulation of the problem

Assume that the tool path $\eta(u) = (x(u),\ y(u),\ z(u),\ a(u),\ c(u))^T, (u \in [0, 1])$ have C^2 continuity. The maximal jerks of the five axes $\mathbf{J}_m = (J_x, J_y, J_z, J_a, J_c)$ are used to describe the maximal change rate of accelerations that the five axes can sustain. So the jerk of every axis in the machining process should satisfy the following constraints:

$$|j_\tau(u)| \le J_\tau, u \in [0, 1], \tau \in \{x, y, z, a, c\}. \tag{26}$$

Use the notations in Section 2. Since the acceleration $\mathbf{a} = (a_x, a_y, a_z, a_a, a_c)$ can be expressed in the following form

$$a_\tau = \ddot{\tau} = \frac{d\left(\sqrt{q}\tau'\right)}{dt} = \frac{d\left(\sqrt{q}\tau'\right)}{du}\sqrt{q} = \tau'' q + \frac{\tau'}{2}q',$$

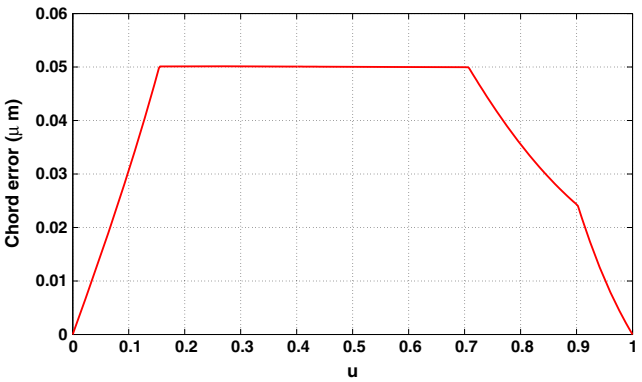
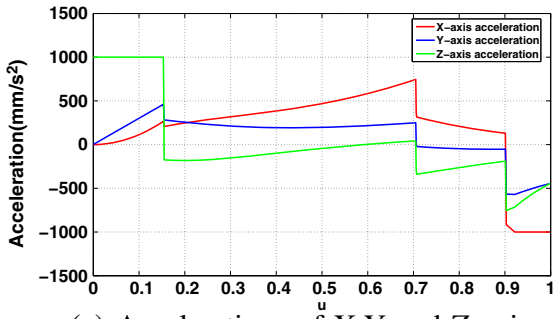
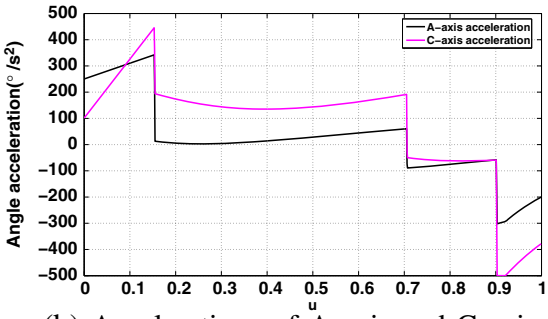


Fig. 5 The chord error



(a) Accelerations of X,Y and Z-axis



(b) Accelerations of A-axis and C-axis

Fig. 6 Accelerations for five axes

the jerk $\mathbf{j} = \left(j_x, j_y, j_z, j_a, j_c\right)$ can be written as

$$j_\tau = \dot{a}_\tau = a'_\tau \sqrt{q} = \left(\tau''' q + \frac{3\tau''}{2} q' + \frac{\tau'}{2} q''\right) \sqrt{q}, \tag{27}$$

where $\tau \in \{x, y, z, a, c\}$. Note that (27) is nonlinear about q, q', and q''.

We assume that the velocities and accelerations at the beginning and end are zero:

$$q(0) = q(1) = 0, \mathbf{a}(0) = \mathbf{a}(1) = \mathbf{0}. \tag{28}$$

Then, the time-optimal velocity planning problem becomes

$$\min_q \int_0^1 \frac{du}{\sqrt{q}} \tag{29}$$

subject to constraints (28), (12), (6), (15), and (26).

Note that after adding jerk constraints, Theorem 3.1 is not fully proved yet. It is not known whether an optimal solution to problem (29) is maximum at any parameter value. But, we still know that an optimal solution must be bang-bang-singular [6]:

Theorem 4.1 *The optimal feedrate* $v_o(u), u \in [0, 1]$ *of problem* (29) *is bang-bang-singular in the sense that for any parameter u, the equality sign holds at least in one of the four constraints* (12), (6), (15), *and* (26).

4.2 Discrete form of the velocity planning problem

Similar to Section 3, the interval $[0, 1]$ is divided into N equal intervals at $u_i = i/N, (i = 0, \ldots, N)$. The first-order and second-order derivatives of q can be approximated as follows:

$$q'_i \approx \frac{q_{i+1} - q_{i-1}}{2\Delta u},$$
$$q''_i \approx \frac{q_{i+1} + q_{i-1} - 2q_i}{\Delta u^2}, i = 1, \ldots, N-1.$$

From Eq. 27, constraint (26) is discretized to the following form:

$$\left|\left(\alpha_{\tau_i} q_{i-1} + \beta_{\tau_i} q_i + \gamma_{\tau_i} q_{i+1}\right) \sqrt{q_i}\right| \le J_\tau, i = 1, \ldots, N-1 \tag{30}$$

where

$$\alpha_{\tau_i} = \frac{\tau'_i}{2\Delta u^2} - \frac{3\tau''_i}{4\Delta u}, \beta_{\tau_i} = \tau'''_i - \frac{\tau'_i}{\Delta u^2}, \gamma_{\tau_i} = \frac{\tau'_i}{2\Delta u^2} + \frac{3\tau''_i}{4\Delta u}$$

and $\tau'_i = \tau'(u_i)$, $\tau''_i = \tau''(u_i)$ and $\tau'''_i = \tau'''(u_i)$.

Since

$$j_\tau = \frac{da_\tau}{dt} = a_\tau \frac{da_\tau}{dv_\tau} = \frac{1}{2}\frac{da_\tau^2}{dv},$$

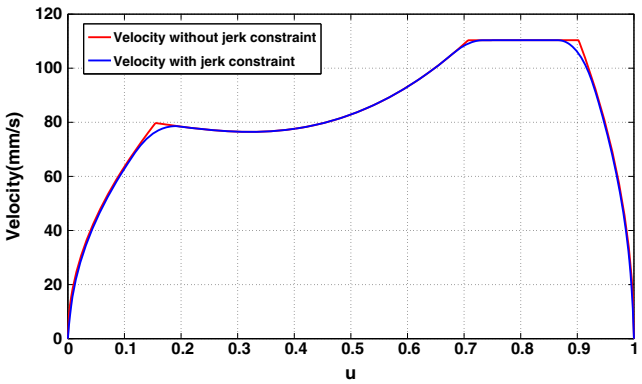


Fig. 7 Feedrates

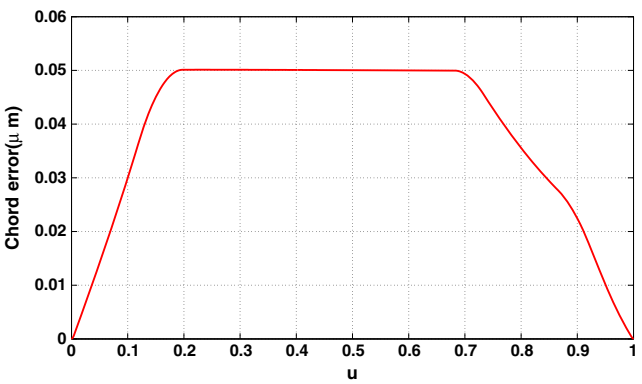


Fig. 8 Chord error with jerk bound

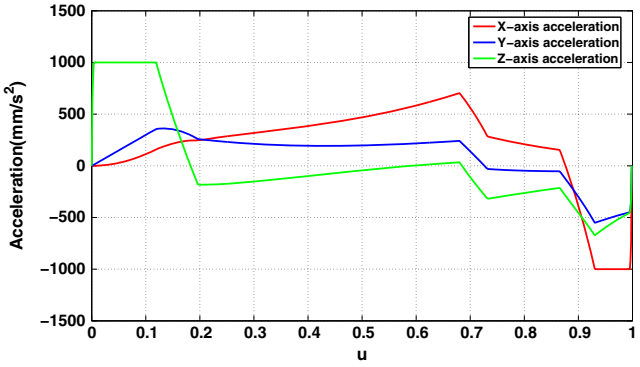


Fig. 9 Accelerations of X-, Y-, and Z-axes

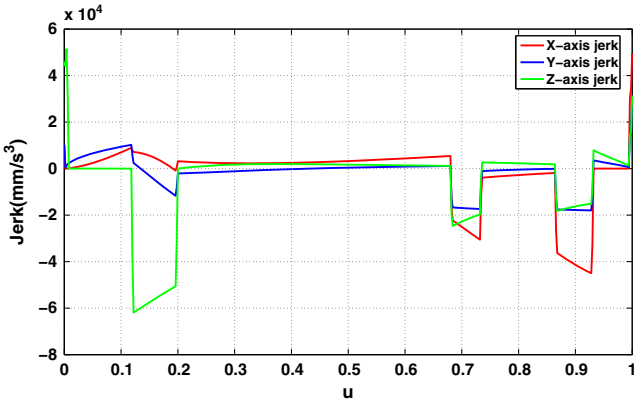


Fig. 11 Jerks of X-, Y-, and Z-axes

for the first interval $[0, u_1]$, $|j_{\tau_1}| = \left|\left(a_{\tau_1}^2 - a_{\tau_0}^2\right) / \left(2v_{\tau_1} - 2v_{\tau_0}\right)\right| = \left|a_{\tau_1}^2 / \left(2\tau_1' \sqrt{q_1}\right)\right| = \left|N^2 \tau_2'^2 \sqrt{q_1} q_2 / 8\tau_1'\right| \leq J_\tau$. Similarly, for the last interval $[u_{N-1}, 1]$, $|j_{\tau_N}| = \left|N^2 \tau_{N-2}'^2 \sqrt{q_{N-1}} q_{N-2} / 8\tau_{N-1}'\right| \leq J_\tau$.Therefore, the jerk constraints in the first and last intervals can be written as follows:

$$|N^2 \tau_2'^2 \sqrt{q_1} q_2 / 8\tau_1'| \leq J_\tau \tag{31}$$

$$|N^2 \tau_{N-2}'^2 \sqrt{q_{N-1}} q_{N-2} / 8\tau_{N-1}'| \leq J_\tau \tag{32}$$

Correspondingly, the optimization problem (29) is transformed into the following *nonlinear programming problem* with objective function

$$\min_{\{q_i\}} \frac{1}{N} \sum_{i=1}^{N-1} \frac{1}{\sqrt{q_i}} \tag{33}$$

under constraints (28), (18), (19), (20), (30), (31), and (32).

We could use existing algorithms to solve the above nonlinear programming problem. But, the solving procedure is generally quite time-consuming, and the solutions thus obtained is not guaranteed to be globally optimal. In the next section, we try to reduce the problem into a linear programming problem.

4.3 Reduce the problem to a linear programming problem

In this section, we will reduce problem (33) into a linear programming problem using a relaxation technique.

The linear programming problem has the following objective function

$$\max_{Q} cQ \tag{34}$$

and constraints $\widehat{M}Q \leq \widehat{R}$, $Q \geq 0$, where $Q = (q_1, \ldots, q_{N-1})$, $c = (1, \ldots, 1)$, and $\widehat{M}$ will be given below. The objective function is the same as Eq. 23, which means to make the total feedrate maximized.

Note that constraints (18), (19), and (20) are already linear in Q. In the following, we will show how to reduce constraints (30), (31), and (32) into a linear form.

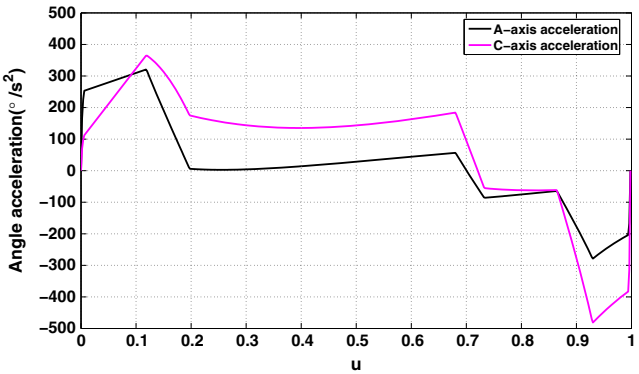


Fig. 10 Accelerations on A- and C-axes

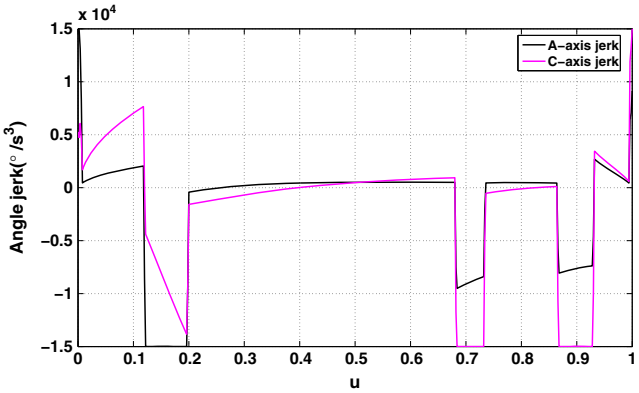
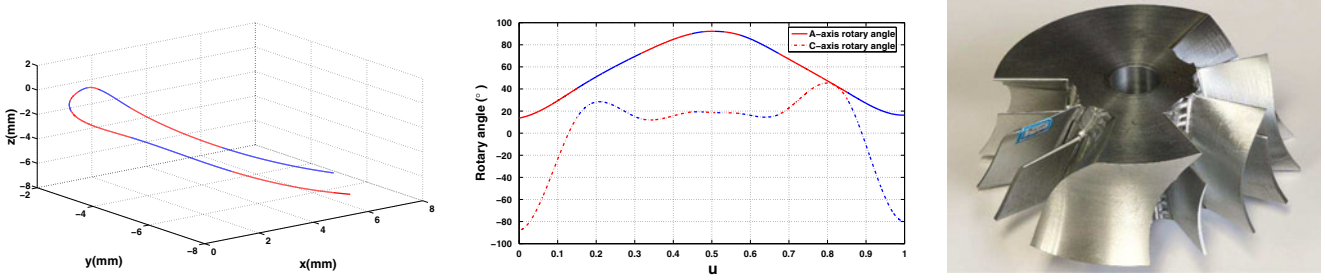


Fig. 12 Jerks of A- and C-axes



(a) Tool path (b) Rotary angles (c) The impeller

Fig. 13 The tool path and rotary angles for a segment of an impeller

Let the optimal solution of problem (23) be $\{q_i^\star\}_{i=0,\ldots,N}$. Multiplying the the corresponding $\sqrt{\frac{q_i^\star}{q_i}}$ on both sides of constraints (30)–(32), we have

$$\left|\left(\sqrt{q_i^\star}\alpha_{\tau_i} q_{i-1} + \sqrt{q_i^\star}\beta_{\tau_i} q_i + \sqrt{q_i^\star}\gamma_{\tau_i} q_{i+1}\right)\right| \leq J_\tau \sqrt{\frac{q_i^\star}{q_i}}, i = 1, \ldots, N-1 \tag{35}$$

$$\left|\frac{N^2\tau_2'^2}{8\tau_1'}\sqrt{q_1^\star} q_2\right| \leq J_\tau \sqrt{\frac{q_1^\star}{q_1}} \tag{36}$$

$$\left|\frac{N^2\tau_{N-2}'^2}{8\tau_{N-1}'}\sqrt{q_{N-1}^\star} q_{N-2}\right| \leq J_\tau \sqrt{\frac{q_{N-1}^\star}{q_{N-1}}} \tag{37}$$

where $\tau \in \{x, y, z, a, c\}$. The left-hand sides of (35)–(37) are linear in q_i. We will show how to relax the right-hand sides of these inequalities into linear forms.

Any q_i satisfying the constraints of problem (33) must also satisfy the constraints of Theorem 3.2. By Theorem 3.2, any such q_i must satisfy $q_i \leq q_i^*$, $(i = 0, \ldots, N)$ or $\sqrt{\frac{q_i^\star}{q_i}} \geq 1$. The following lemma gives a better estimation for $\sqrt{\frac{q_i^\star}{q_i}}$.

Lemma 4.2 *If $\{q_i^\star\}_{i=1,\ldots,N-1}$ is the optimal solution of Eq. (23) and $\{q_i\}_{i=1,\ldots,N-1}$ is the optimized solution of problem (33), then*

$$\sqrt{\frac{q_i^\star}{q_i}} \geq \frac{3}{2} - \frac{q_i}{2q_i^\star} \geq 1.$$

Proof From Theorem 4.2, we have $q_i \leq q_i^\star$, and hence $\frac{3}{2} - \frac{q_i}{2q_i^\star} \geq 1$. Let $t = \sqrt{q_i/q_i^\star}$. Then

$$\sqrt{\frac{q_i^\star}{q_i}} - \left(\frac{3}{2} - \frac{q_i}{2q_i^\star}\right) = \frac{t^2}{2} + \frac{1}{t} - \frac{3}{2} = \frac{t^2}{2} + \frac{1}{2t} + \frac{1}{2t} - \frac{3}{2} \geq 3\sqrt[3]{\frac{t^2}{2}\cdot\frac{1}{2t}\cdot\frac{1}{2t}} - \frac{3}{2} = 0.$$

□

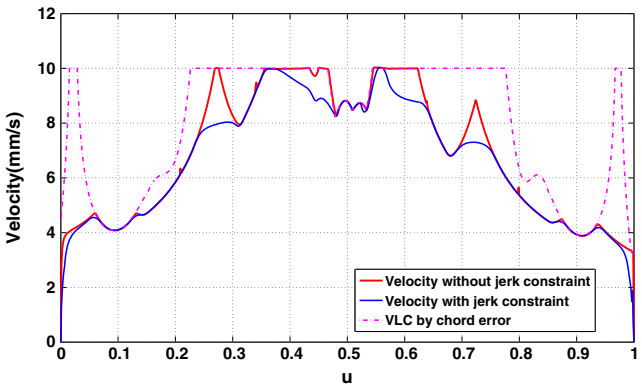


Fig. 14 Velocity curves

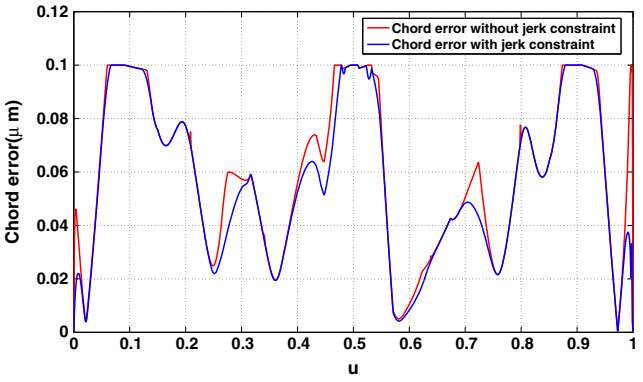


Fig. 15 Chord error curves

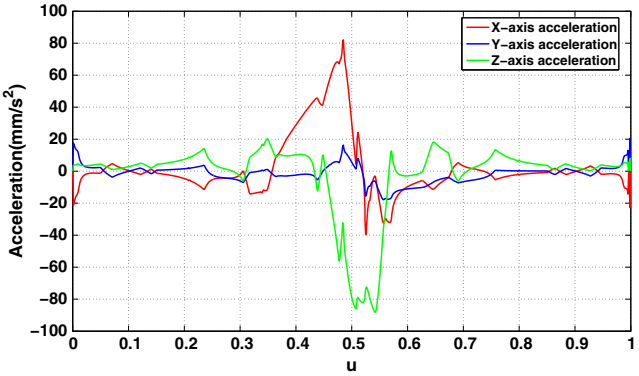


Fig. 16 Accelerations of X-, Y-, and Z-axes

By Lemma 4.2, if q_i satisfy the following constraints, then they also satisfy constraints (35)–(37).

$$\left|\left(\sqrt{q_i^\star}\alpha_{\tau_i}q_{i-1}+\sqrt{q_i^\star}\beta_{\tau_i}q_i+\sqrt{q_i^\star}\gamma_{\tau_i}q_{i+1}\right)\right| \le J_\tau\left(\frac{3}{2}-\frac{q_i}{2q_i^\star}\right) \tag{38}$$

$$\left|\frac{N^2\tau_2'^2}{8\tau_1'}\sqrt{q_1^\star}q_2\right| \le J_\tau\left(\frac{3}{2}-\frac{q_1}{2q_1^\star}\right) \tag{39}$$

$$\left|\frac{N^2\tau_{N-2}'^2}{8\tau_{N-1}'}\sqrt{q_{N-1}^\star}q_{N-2}\right| \le J_\tau\left(\frac{3}{2}-\frac{q_{N-1}}{2q_{N-1}^\star}\right) \tag{40}$$

The final linear programming problem to be solved has objective function (34) and constraints (28), (18), (19), (20), (38), (39), and (40).

Now, we will give the $\widehat{M}$ in problem (34). Since M in (24) represents constraints (18), (19), and (20), the $11N-1$ rows of $\widehat{M}$ are the same as that of M.

Let $\widehat{\alpha}_{\tau_0} = J_\tau/(2q_1^\star)$, $\widehat{\beta}_{\tau_0} = \sqrt{q_1^\star}N^2\tau_2'^2/(8\tau_1')$, $\widehat{\alpha}_{\tau_N} = \sqrt{q_{N-1}^\star}N^2\tau_{N-2}'^2/(8\tau_{N-1}')$, $\widehat{\beta}_{\tau_N} = J_\tau/(2q_{N-1}^\star)$, $\widehat{\alpha}_{\tau_i} = \alpha_{\tau_i}\sqrt{q_i^\star}$, $\widehat{\beta}_{\tau_i} = \beta_{\tau_i}\sqrt{q_i^\star} + J_\tau/(2q_i^\star)$, $\widehat{\gamma}_{\tau_i} = \gamma_{\tau_i}\sqrt{q_i^\star}$, $\theta_{\tau_i} = J_\tau/q_i^\star - \widehat{\beta}_{\tau_i}$, $i = 1,\ldots,N-1$, $\widehat{\alpha}_i = (\widehat{\alpha}_{x_i},\widehat{\alpha}_{y_i},\widehat{\alpha}_{z_i},\widehat{\alpha}_{a_i},\widehat{\alpha}_{c_i})^T$, $\widehat{\beta}_i = (\widehat{\beta}_{x_i},\widehat{\beta}_{y_i},\widehat{\beta}_{z_i},\widehat{\beta}_{a_i},\widehat{\beta}_{c_i})^T$, $\widehat{\gamma}_i = (\widehat{\gamma}_{x_i},\widehat{\gamma}_{y_i},\widehat{\gamma}_{z_i},\widehat{\gamma}_{a_i},\widehat{\gamma}_{c_i})^T$, and $\theta_i = (\theta_{x_i},\theta_{y_i},\theta_{z_i},\theta_{a_i},\theta_{c_i})^T$. The inequality (38)–(40) is added to M to obtain $\widehat{M}$

$$\widehat{M}=\begin{pmatrix}
\lambda_1 & 0 & 0 & \ldots & 0 & 0\\
-\lambda_1 & 0 & 0 & \ldots & 0 & 0\\
\widehat{\beta}_1 & \widehat{\gamma}_1 & 0 & \ldots & 0 & 0\\
\theta_1 & -\widehat{\gamma}_1 & 0 & \ldots & 0 & 0\\
1 & 0 & 0 & \ldots & 0 & 0\\
-\lambda_1 & \lambda_2 & 0 & \ldots & 0 & 0\\
\lambda_1 & -\lambda_2 & 0 & \ldots & 0 & 0\\
\widehat{\alpha}_2 & \widehat{\beta}_2 & \widehat{\gamma}_2 & \ldots & 0 & 0\\
-\widehat{\alpha}_2 & \theta_2 & -\widehat{\gamma}_2 & \ldots & 0 & 0\\
0 & 1 & 0 & \ldots & 0 & 0\\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots\\
0 & 0 & 0 & \ldots & -\lambda_{N-2} & \lambda_{N-1}\\
0 & 0 & 0 & \ldots & \lambda_{N-2} & -\lambda_{N-1}\\
0 & 0 & 0 & \ldots & \widehat{\alpha}_{N-1} & \widehat{\beta}_{N-1}\\
0 & 0 & 0 & \ldots & -\widehat{\alpha}_{N-1} & \theta_{N-1}\\
0 & 0 & 0 & \ldots & 0 & 1\\
0 & 0 & 0 & \ldots & 0 & -\lambda_{N-1}\\
0 & 0 & 0 & \ldots & 0 & \lambda_{N-1}\\
\widehat{\alpha}_0 & \widehat{\beta}_0 & 0 & \ldots & 0 & 0\\
\widehat{\alpha}_0 & -\widehat{\beta}_0 & 0 & \ldots & 0 & 0\\
0 & 0 & 0 & \ldots & \widehat{\alpha}_N & \widehat{\beta}_N\\
0 & 0 & 0 & \ldots & -\widehat{\alpha}_N & \widehat{\beta}_N
\end{pmatrix} \tag{41}$$

The right-hand side vector in Eq. 34 is

$$\widehat{R}=(\pi_0,\pi_0,1.5\mathbf{J}_m,1.5\mathbf{J}_m,h(u_1),\ldots,\pi_{N-2},\pi_{N-2},1.5\mathbf{J}_m,1.5\mathbf{J}_m,h(u_{N-1}),\pi_{N-1},\pi_{N-1},1.5\mathbf{J}_m,1.5\mathbf{J}_m)^T.$$

Now, all the parameters in problem (34) are given, and we can give the velocity planning algorithm.

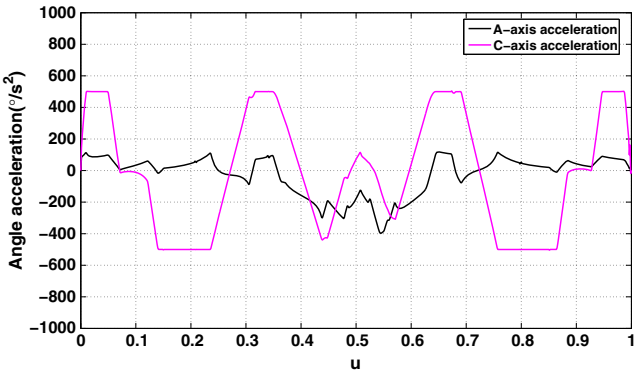


Fig. 17 Accelerations of A- and C-axes

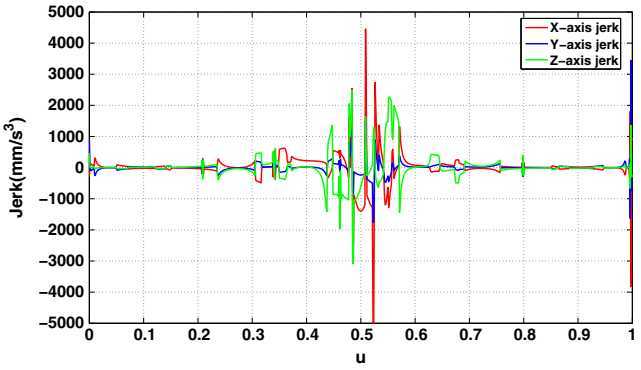


Fig. 18 Jerks of X-, Y-, and Z-axes

Algorithm 4.3 VPJ_LP

Input: *Five-axis tool path* $\eta(u)$,$u \in [0,1]$ *in MCS, the sampling period* T, *the feedrate bound* V_m, *five-axis acceleration bounds* $(A_x, A_y, A_z, A_a, A_c)$, *the chord error bound* δ_m, *five-axis jerk bounds* $(J_x, J_y, J_z, J_a, J_c)$, *the*

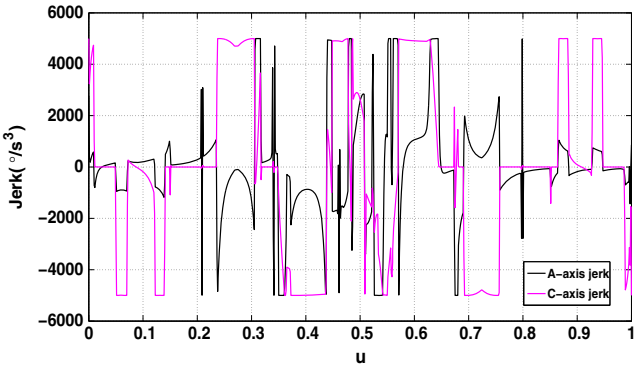


Fig. 19 Jerks of A- and C-axes

number of discretization N, and the initial position of the origin of MCS $O_w = (x_{w_0}, y_{w_0}, z_{w_0})$.

Output: *The approximate optimal velocity function $v(u)$, $u \in [0, 1]$ of problem (17).*

The algorithm is almost the same as Algorithm 3.3. The only difference is that instead of solving linear programming (23), we now solve linear programming (34). Since the matrix $\widehat{M}$ is of the size $(21N + 9) \times (N - 1)$, the computational complexity of Algorithm 3.3 is also $O\left(N^{3.5}\right)$ in terms of floating point arithmetical operations.

5 Simulation

In this section, simulation results are used to show that Algorithm 4.3 can be used to find the approximate optimal solution very efficiently.

5.1 Simulation results

For the tool path in Fig. 2, we use Algorithm 4.3 to find the velocity function obeying jerk constraints. Set $J_x = J_y = J_z = 100{,}000$ mm/s^3, $J_a = J_c = 15{,}000°/s^3$. Other parameters are the same as those given in Section 3.4. The velocity functions obtained with Algorithms 3.3 and 4.3 are given in Fig. 7. The "sharp corners" of the velocity curve under confined acceleration are smoothed by adding the jerk constraint.

Figure 8 shows the theoretical chord error of the velocity curve under confined jerk, which is calculated by (25). The chord error in Fig. 8 is smaller than that given in Fig. 5 at some positions.

Table 1 Computation times of Algorithm **VPA_LP** for different N

N	100	200	300	500	1,000
Fig. 2 (s)	0.328	0.625	0.922	2.187	3.594
Fig. 13 (s)	0.390	0.844	1.218	2.485	5.047

Table 2 Computation times of Algorithms **NP** and **VPJ_LP** for different N

	Fig. 2		Fig. 13	
Number (N)	VPJ_LP (s)	NP (min)	VPJ_LP (s)	NP (min)
100	1.609	42	2.740	67
200	3.766	149	5.000	175
300	5.797	Failure	6.641	Failure
500	10.875	Failure	10.813	Failure
1,000	32.640	Failure	27.047	Failure

The acceleration curves of five axes with jerk constraints are shown in Figs. 9 and 10. Different from the acceleration curves under confined acceleration given in Fig. 6, these curves are continuous. The jerk curves of five axes with jerk constraints are shown in Figs. 11 and 12, from which we can see that the jerk bounds are satisfied.

In Figs. 7 to 12, it is easy to see that the control profile of velocity is approximately bang-bang-singular. In fact, when $u \in [0, 0.001]$, A-axis jerk reaches J_a. When $u \in [0.001, 0.125]$, Z-axis acceleration reaches A_z. A-axis jerk reaches $-J_a$ when $u \in [0.125, 0.192]$. Chord error reaches maximum value when $u \in [0.192, 0.68]$. C-axis jerk reaches $-J_c$ when $u \in [0.68, 0.733]$. Velocity reaches V_m when $u \in [0.733, 0.86]$. Once again, C-axis jerk reaches $-J_c$ when $u \in [0.86, 0.93]$. The X-axis acceleration reaches $-A_x$ when $u \in [0.93, 0.997]$. The C-axis jerk reaches J_c when $u \in [0.997, 1]$.

Now, consider a more complicated tool path shown in Fig. 13. This five-axis curve is one piece of tool path for an "impeller" shown in Fig. 13c. Both the tool path curve and the rotary angle curve are cubic B-splines containing ten curve segments and having C^2 continuity. The parameters are set to be $V_m = 10$ mm/s, $A_x = A_y = A_z = 200$ mm/s^2, $A_a = A_c = 500°/s^2$, $J_x = J_y = J_z = 5{,}000$ mm/s^3, $J_a = J_c = 5{,}000°/s^3$, $\delta_m = 0.1$ μm, $T = 3$ ms, and $N = 500$.

The velocity curves obtained by Algorithm **VPA_LP** and Algorithm **VPJ_LP** are given in Fig. 14. The velocity limit curve by chord error defined in [5], which is the maximal feedrate value not violating the chord error, is also given. The chord error curves with and without jerk constraints are shown in Fig. 15. Accelerations on five axes are illustrated in Figs. 16 and 17. Figures 18 and 19 show the jerks of the X-, Y-, and Z-axes and rotary axes A and C respectively. It is easy to find that for the planned velocity, all constraints are satisfied, and the motion profile is nearly bang-bang-singular. Due to Theorem 4.1, the velocity is a tight approximate optimal solution to the original velocity planning problem (29).

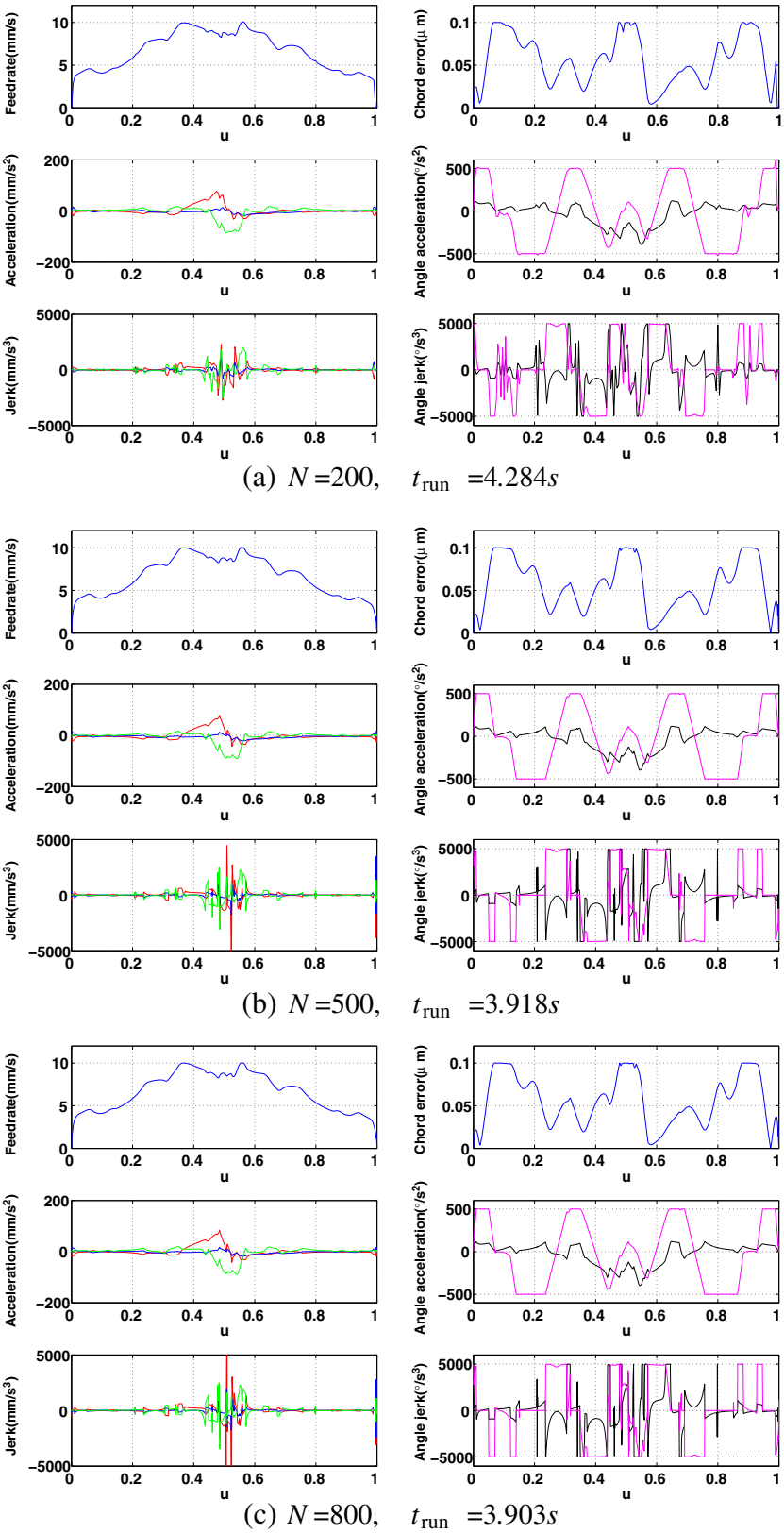


Fig. 20 The feedrate, chord error, acceleration, jerk, and the machining time under different N. For the acceleration and jerk profiles, the *black* and *purple curves* denote the kinematic profiles of A-axis and C-axis, respectively

5.2 Computational costs analysis

In Algorithm **VPA_LP**, the linear programming is used to obtain the optimal solution. The larger the number of refined intervals N is, the more closely the solution of Algorithm **VPA_LP** approximates the solution of the original problem (17). On the other hand, computational costs will rise as N becomes lager. In this section, we will use experimental data to show that for our problems, the computational complexity is much better than the worst case complexity $O\left(N^{3.5}\right)$.

In Table 1, the computation times using Algorithm **VPA_LP** to plan the velocities for the tool paths in Figs. 2 and 13 under different N are given. Both algorithms use MatLab on a PC with a 1.79-GHz CPU and 2-G memory.

In Table 1, we have the following observations. Firstly, the actual computational complexity of Algorithm **VPA_LP** is about $O(N)$, that is, the computational time is proportional to N. This is in accordance with Smale's result that the number of operations required to solve a linear programming problem grows in proportion to the number of variables on the average under certain conditions [33]. Secondly, for different tool paths, the computational times are of little difference, which is very significant for practical usage since tool path curves in the practical problem may be rather complicated.

Algorithm **VPJ_LP** computes approximate solutions of the time-optimal velocity planning problem under confined jerk using linear programming methods. The original velocity planning problem (33) under confined jerk is a nonlinear programming problem, which is denoted by **NP** and can be solved with the nonlinear programming software in MatLab. In Table 2, we compare the computation time for Algorithms **VPJ_LP** and **NP** for different N.

From Table 2, we can see that Algorithm **VPJ_LP** is much faster than nonlinear programming **NP**. When N exceeds 300, **NP** cannot solve the nonlinear problem, while Algorithm **VPJ_LP** stills works well.

5.3 Criterion of subdivision

It is apparent that the quality of the solution depends on N. In this section, we give a criterion for selecting N.

Take the tool path given in Fig. 13 as an example. The feedrate, acceleration, and jerk profiles computed with different N are illustrated in Fig. 20. In Fig. 20a, it can be seen that for $N = 200$, the C-axis acceleration violates the constraint at some positions, such as $u = 0.985$. Also, the jerk profiles for $N = 200$ perform not well in terms of the bang-bang-singular control. When $N = 500$, the kinematic profiles obey all the constraints, and the bang-bang-singular mode becomes more apparent. When $N = 800$, the results are almost the same as that of $N = 500$. This indicates that when N is small, the computed velocity curves may be of poor quality. For a sufficiently large N, the computed velocity curves give a nice approximation to the optimal solutions. This can also be seen from the machining times for different values of N.

The selection of N is closely related with the lengths $\Delta s_i = ||r(u_i) - r(u_{i-1})||$ of the subdivided curve segments. According to the theory of finite element method, the length of the refined interval is often required to satisfy $\Delta s_i \leq 0.1$ mm in order to achieve highly accurate computation [24]. So, a criterion for selecting N is to chose the least integer such that $\Delta s_i \leq 0.1$ mm, $i = 1...N$. For simplicity, the dichotomy could be used to find the proper N before starting the algorithm. Specifically, a lower bound of N is set to be $\max\{300, \lceil 10S_m \rceil\}$, where S_m denotes the path length with unit in millimeter, and $\lceil 10S_m \rceil$ is the least positive integer not less than $10S_m$. By the method of estimating Δu in the reference [20], an upper bound of N could be approximately taken to be $\max\{300, \lceil 10 \max_{u\in[0,1]} \sigma(u) \rceil\}$.

We finally remark that a nonuniform subdivision taking into consideration the local shape of the tool path, instead of the uniform one used in this paper, could result in better performance. This will be considered in our future work.

6 Conclusion

In this paper, the time-optimal velocity planning problem for five-axis CNC machining along a given parametric tool path under chord error, acceleration, and jerk constraints is studied.

The parameter interval $[0, 1]$ is divided into N equal parts and the differential quantities are discretized into finite differences. Then, the velocity planning problem under confined acceleration is reduced to an equivalent linear programming problem, and as a consequence, a polynomial time algorithm with complexity $O\left(N^{3.5}\right)$ is given to find the optimal solution.

In the case of confined jerk, the nonlinear jerk constraint is approximated with a linear function, and the velocity planning problem under confined jerk is approximated with a linear programming program. As a consequence, a

polynomial time algorithm is given to find the approximate optimal solution under confined jerk.

By using linear programming methods, efficient polynomial time algorithms are given for time-optimal velocity planning of five-axis CNC machining along any parametric tool path under confined chord error, acceleration, and jerk. Simulation results and complexity analysis are used to show that the proposed algorithm can find very efficiently the approximate time-optimal velocity for complicated five-axis tool paths. A criterion for selection of N is also given.

We finally remark that the sequence of velocity values obtained with Algorithms **VPA_LP** and **VPJ_LP** can be fitted into a continuous function in u and used in CNC machining similar to what we did in [16].

Acknowledgments This study is partially supported by the National Key Basic Research Project of China (2011CB302400) and by a grant from NSFC (60821002).

References

1. Bobrow JE, Dubowsky S, Gibson JS (1985) Time-optimal control of robotic manipulators along specified paths. Int J Robot Res 4(3):3–17
2. Shiller Z (1994) On singular time-optimal control along specified paths. IEEE Trans Robot Autom 10:561–566
3. Timar SD, Farouki RT (2007) Time-optimal traversal of curved paths by Cartesian CNC machines under both constant and speed-dependent axis acceleration bounds. Robot Comput Integr Manuf 23(5):563-579
4. Zhang M, Yan W, Yuan CM, Wang DK, Gao XS (2011) Curve fitting and optimal interpolation on CNC machines based on quadratic B-splines. Sci China Ser E 54(7):1407–1418
5. Yuan CM, Zhang K, Fan W, Gao XS (2011) Time-optimal interpolation for CNC machining along curved tool paths with confined chord error. MM Res Prepr 30:57–89
6. Zhang K, Gao XS, Li HB, Yuan CM (2012) A greedy algorithm for feed-rate planning of CNC machines along curved tool paths with confined jerk for each axis. Robot Comput Integr Manuf 28:472–483
7. Bedi D, Ali I, Quan N (1993) Advanced techniques for CNC machines. J Eng Ind Trans 115:329–336
8. Yang DCH, Kong T (1994) Parametric interpolator versus linear interpolator for precision CNC machining. Comput Aided Des 26(3):225–234
9. Yeh SS, Hsu PL (2002) Adaptive-feedrate interpolation for parametric curves with a confined chord error. Comput Aided Des 34:229–237
10. Nam SH, Yang MY (2004) A study on a generalized parametric interpolator with real-time jerk-limited acceleration. Comput Aided Des 36:27–36
11. Emami MM, Arezoo B (2010) A look-ahead command generator with control over trajectory and chord error for NURBS curve with unknown arc length. Comput Aided Des 4(7):625–632
12. Lai JY, Lin KY, Tseng SJ, Ueng WD (2008) On the development of a parametric interpolator with confined chord error, feedrate, acceleration and jerk. Int J Adv Manuf Technol 37:104–121
13. Feng J, Li Y, Wang Y, Chen M (2010). Int J Adv Manuf Technol 48:227–241
14. Sun Y, Jia Z, Ren F, Guo D (2008) Adaptive feedrate scheduling for NC machining along curvilinear paths with improved kinematic and geometric properties. Int J Adv Manuf Technol 36:60–68
15. Beudaert X, Lavernhe S, Tournier C (2012) Feedrate interpolation with axis jerk constraints on 5-axis NURBS and G1 tool path. Int J of Mach Tools and Manu:5773–82
16. Zhang K, Yuan CM, Gao XS (2013) Efficient algorithm for feedrate planning and smoothing with confined chord error and acceleration for each axis. Int J Adv Manuf Technol 66:1685–1697
17. Yong T, Narayanaswami R (2003) A parametric interpolator with confined chord errors, acceleration and deceleration for NC machining. Comput Aided Des 35:1249–1259
18. Lin MT, Tsai MS, Yau HT (2007) Development of a dynamics-based NURBS interpolator with real-time look-ahead algorithm. Int J of Mach Tools and Manu 47:2246–2262
19. Tsai MS, Nien HW, Yau HT (2008) Development of an integrated look-ahead dynamics-based NURBS interpolator for high precision machinery. Comput Aided Des 40:554–566
20. Fan W, Gao XS, Yan W, Yuan CM (2011) Interpolation of parametric CNC machine tool path under confined jounce. Int J Adv Manuf Technol 62:719–739
21. Lee AC, Lin MT, Pan YR, Lin WY (2011) The feedrate scheduling of NURBS interpolator for CNC machine tools. Comput Aided Des 43:612–628
22. Erkorkmaz K, Altintas Y (2001) High speed, CNC system design. Part I, jerk limited trajectory generation and quintic spline interpolation. Int J of Mach Tools and Manu 41:1323–1345
23. Dong JY, Ferreiraa PM, Stori JA (2007) Feedrate optimization with jerk constraints for generating minimum-time trajectories. Int J of Mach Tools and Manu 47:1941–1955
24. Sencer B, Altintas Y, Croft E (2008) Feed optimization for five-axis CNC machine tools with drive constraints. Int J of Mach Tools and Manu 48:733–745
25. Li SR, Zhang Q, Gao XS, Li HB (2012) Minimum time trajectory planning for five-axis machining with general kinematic constraints. MM Res Prepr 31:1–20
26. Gasparetto A, Lanzutti A, Vidoni R, Zanotto V (2012) Experimental validation and comparative analysis of optimal time-jerk algorithms for trajectory planning. Robot Comput Integr Manuf 28:164–181
27. Zhang LX, Sun RY, Gao XS, Li HB (2011) High speed interpolation for micro-line trajectory and adaptive real-time lookahead in CNC machining. Sci China - Inf Sci 54:1481–1495
28. Xu ZM, Chen JC, Feng ZJ (2002) Performance evaluation of a real-time interpolation algorithm for NURBS curves. Int J Adv Manuf Technol 20:270–276
29. Li YY, Feng JC, Wang YH, Yang JG (2009) Variable-period feed interpolation algorithm for high-speed five-axis machining. Int J Adv Manuf Technol 40:769–775
30. Park J, Nam SH, Yang MY (2005) Development of a real-time trajectory generator for NURBS interpolation based on the two-stage interpolation method. Int J Adv Manuf Technol 26:359–365
31. Wang JB, Yau HT (2008) Real-time NURBS interpolator: application to short linear segments. Int J Adv Manuf Technol 41:1169–1185
32. Karmarkar N (1984) A new polynomial time algorithm for linear programming. Combinatorica 4(4):373–395
33. Smale S (1983) On the average number of steps of the simplex method of linear programming. Math Program 27(3):241–262