ELSEVIER

Computer-Aided Design 33 (2001) 1–15

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

A constant feed and reduced angular acceleration interpolation algorithm for multi-axis machining

R.V. Fleisig, A.D. Spence*

Department of Mechanical Engineering, McMaster University, 1280 Main St. W., Hamilton, Ont., Canada L8S 4L7

Received 27 September 1999; received in revised form 15 February 2000; accepted 23 February 2000

Abstract

Multi-axis tool paths are currently generated as a set of discrete data points consisting of a position vector, representing the tool tip, and an orientation unit vector, representing the tool axis. The CNC interpolator must convert these points into continuous machine tool axis motions. To achieve the highest quality parts, a constant feed and reduced angular acceleration must be maintained throughout the motion. This paper presents a new algorithm for off-line interpolation of the data points, followed by real-time axis command generation. The splines produced by the algorithm are C^2 continuous, and independent of machine tool kinematics. Hence the motions produced will be the same on any five-axis machine tool, hexapod, or robotic arm. Three splines are computed: position, orientation, and reparameterization. The position spline is a near arc-length parameterized quintic polynomial spline. The paper introduces a near arc-length parameterized quintic spherical Bézier spline as the orientation spline. Coordinated motion is accomplished with an orientation reparameterization spline. The proposed algorithm is demonstrated using a practical example. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Five-axis machining; Interpolation; Computer numerical control

1. Introduction

The majority of milling operations employ three-axis CNC machine tools. These machines have the virtue of a direct correspondence between the part centric CL-DATA tool path coordinates and the machine tool axis motions. Multi (four and five)-axis motions, however, can only be related to the CL-DATA coordinates by solving a set of machine specific kinematic equations. To use a different machine tool configuration, appropriate parameters must be determined, and the axes motions must again be off-line calculated using a commercial post processor such as IntelliPost [19]. This complication has impeded the introduction of multi-axis machines for many manufacturing operations, despite advantages such as increased metal removal rate, the improved repeatability of one setup, and improved surface finish [9].

Recent research on interpolators for three-axis machines has demonstrated that reduced cutting time, better accuracy, and improved surface finish can be achieved through the application of parametric splines. In this paper, algorithms are developed that extend the benefits of parametric spline interpolation to multi-axis machining.

In Section 2, interpolation, as it has been applied to three-axis tool paths, is reviewed. This is followed by a discussion of multi-axis interpolation. In Section 3, the proposed multi-axis interpolation algorithm is presented. A practical example of the proposed algorithm is provided in Section 4.

2. Previous work

Machining of mechanical parts requires generation of tool paths defining the motion of the tool with respect to the part [7]. Three-axis tool paths are represented by a set of Cartesian position vectors. Each vector describes the location of the tool with respect to the part. Feedrate information is also provided. Multi-axis tool paths also specify the orientation of the tool. Since the tool rotates about the spindle during cutting, only the remaining two orientation degrees of freedom are necessary. A five-degree of freedom tool path is therefore sufficient to specify the position and orientation.

In general, a multi-axis tool path is represented by a set of

* Corresponding author. Tel.: + 1-905-525-9140, ext. 27130; fax: + 1-905-572-7944.

E-mail address: adspence@mcmaster.ca (A.D. Spence).

Nomenclature

A, C, X, Y, Z	Five-axis CNC machine tool joint coordinates
a, b, c	Scalars used in calculation of reparameterization spline
$\mathbf{B}$	Spherical Bézier vector function
$\mathbf{c}$	Position spline vector polynomial coefficient
$\mathbf{d}$	Orientation spline vector control point
d_x, d_y, d_z	Workpiece offset coordinates
e, i	Spline segment index
$\mathbf{f}$, $\mathbf{g}$, $\mathbf{h}$	Vectors used in calculation of the spherical Bézier spline second derivatives
f	Feedrate (mm/s)
h	First derivative of reparameterization spline
j	Curve degree
k	Spherical Bézier curve control point index
l	Range of position spline segment
m	Sample
n	Number of segments in a spline
$\mathbf{P}$	Position spline vector function
$\mathbf{p} = [p_x \quad p_y \quad p_z]^{\mathrm{T}}$	Tool tip position vector and scalar components (mm)
$\mathbf{Q}$	Orientation spline vector function
$\mathbf{q} = [q_x \quad q_y \quad q_z]^{\mathrm{T}}$	Tool axis orientation unit vector and scalar components
$\mathbf{S}$	Spherical interpolation vector function
s	Arc-length parameter
T	Servo update period (s)
$\mathbf{t}$	Tool path displacement vector
t	Time (s)
u	Position spline parameter
V	Reparameterization spline function
v	Orientation spline parameter
w	Iteration index
λ	Range of orientation spline parameter
μ	Orientation spline knot
θ	Angle between two unit vectors

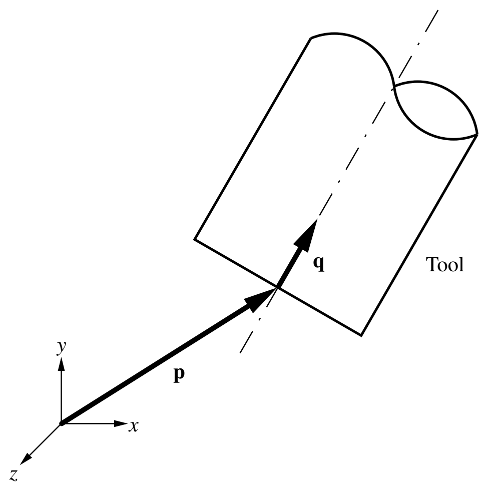

Fig. 1. The tool tip position vector, **p**, and the tool axis orientation vector, **q**.

tool path vectors, **t**

$$\mathbf{t} = [\mathbf{p} \quad \mathbf{q}]^{\mathrm{T}},\ \mathbf{p} = [p_x \quad p_y \quad p_z]^{\mathrm{T}},\ \mathbf{q} = [q_x \quad q_y \quad q_z]^{\mathrm{T}},\ |\mathbf{q}| = 1 \tag{1}$$

The tool tip position vector **p** is represented in Cartesian coordinates. The orientation of the tool axis is represented by a unit vector **q** (Fig. 1).

The challenge in converting tool paths into axis motions is to command the machine tool axes to move at the specified feedrate. Because it is impractical to specify the axis positions at the motion controller loop closing frequency (approximately 1 kHz), interpolatory splines are used. Maintaining a near constant feedrate minimizes cutting force variation, avoids chatter, and produces a smoother surface finish.

For multi-axis tool paths the angular velocity of the tool must also be taken into account since the effective feedrate along the tool-axis will vary with the angular velocity.

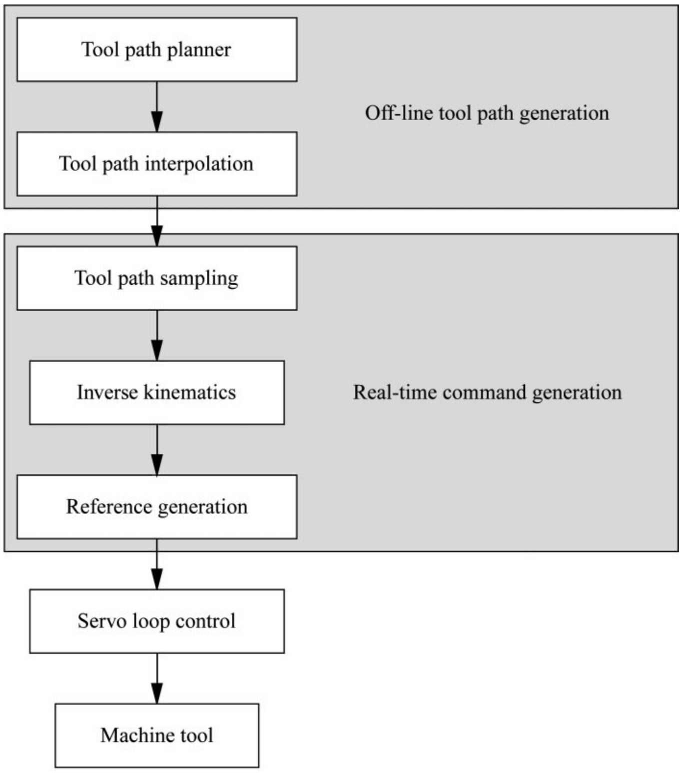

Fig. 2. Real-time multi-axis interpolator architecture proposed by Lin and Koren [12].

Commercial CAD/CAM systems and post processors first perform the inverse kinematics algorithm on the tool path points, and then discretize the transformed tool path into a series of short linear segments. The linear segments are discretized into real-time axis positions by the machine tool controller. Previous researchers, Koren and Lo [10], and Yang and Kong [24], have demonstrated the following limitations of linear interpolators:

- The velocity discontinuities at the linear segment junctions lead to higher accelerations, poor surface finish, lower surface accuracy and longer machining times.
- Generally, the linear segment length cannot be evenly subdivided by the position loop update period. As a result, the distance between the last two points will be shorter than the rest, resulting in undesirable accelerations and longer machining times.

Fitting parametric splines to meet the geometric requirement of interpolating the tool path points is a straightforward task. Most researchers have used polynomial splines of n segments and degree j which vary with the parameter, u

$$\mathbf{p} = \mathbf{P}_i(u) = \mathbf{c}_{1_i} + \mathbf{c}_{2_i}u + \cdots + \mathbf{c}_{(j+1)_i}u^j, \qquad i = 1, \ldots, n \tag{2}$$

The difficulty has been in obtaining the desired feedrate along the spline. Two approaches have evolved to solve this problem. The first method, developed by Shpitalni [18], is to discretize the spline in real-time by estimating the next point along the spline at a constant feedrate, f, using a truncated Taylor series expansion

$$u_m = u_{m-1} + \frac{fT}{\sqrt{\left(\frac{\mathrm{d}p_x}{\mathrm{d}u}\right)^2_{m-1} + \left(\frac{\mathrm{d}p_y}{\mathrm{d}u}\right)^2_{m-1} + \left(\frac{\mathrm{d}p_z}{\mathrm{d}u}\right)^2_{m-1}}} \tag{3}$$

where T is the sampling period. The second notion, developed by Renner [15] and improved successively by Wang and Yang [23], and Wang and Wright [22], is to introduce splines with the property of near unit tangency or near arc-length parameterization. With this property, the next point along such a spline is obtained with

$$u_m = u_{m-1} + fT \tag{4}$$

To reduce accelerations due to geometry the splines are constructed to be at least C^2 continuous.

Parametric splines reduce or eliminate the two drawbacks of linear interpolation: velocity discontinuity between segments and end of segment accelerations. As a consequence, the machining times are reduced while improving part surface finish and accuracy.

There is little work on interpolators for multi-axis machining. Lin and Koren [12] applied the Taylor series approximation technique to the ruled surface subset of tool paths. A significant contribution in the work, however, is a new architecture for multi-axis interpolation (Fig. 2). Unlike common practice, discretization of the tool path is performed first, in part coordinates, and then inverse kinematics is performed in real-time. This distinction avoids the need to repeat the interpolation step if there is a change in the location or orientation of the part, or a change in tool length.

A more general approach would be to interpolate the tool path position and orientation vectors, regardless of the surface type. The primary obstacle is the interpolation of the orientation unit vectors, while ensuring that the entire interpolated spline remains on the surface of the unit sphere. A method of interpolating the orientation as well as the position has been suggested by Ge and Srinivasan [5,20] and Jüttler [8]. Their work is based on the notion that tool path vectors are transformed into image space where a spline is interpolated through all coordinates of the transformed vector simultaneously. The spline may then be sampled in the real-time controller of the machine tool using the Taylor series approximation described above. This method ensures an acceptable constant feed. However, the angular velocity and therefore the effective feedrate will be uncontrolled because Eq. (3) does not account for the parameterization of the tool orientation. This effect is greatest where there is a small feed relative to the angular velocity. The algorithm developed in this paper overcomes this shortcoming.

3. Proposed algorithm

The proposed multi-axis interpolation algorithm aims to extend the three-axis near arc-length parameterized quintic spline algorithm by Wang and Yang [23] to multi-axis machining. In particular, the interpolated tool path will be C^2 continuous with a near constant feed and an angular velocity with reduced angular acceleration. This scheme is arranged in the manner suggested by Lin and Koren [12], in which part coordinate tool path interpolation is performed off-line, and sampling and inverse kinematics are performed in real-time on the machine tool controller.

The interpolation algorithm is divided into three stages. In the first stage, a near arc-length parameterized quintic polynomial spline is interpolated through the position vectors. In the second stage, a near arc-length parameterized quintic spherical Bézier spline is interpolated through the orientation unit vectors. The resultant spline lies on the unit sphere. If this spline were sampled in real-time a near constant angular velocity would result. To synchronize the position and orientation splines a reparameterization spline is constructed during the third stage. This spline reparameterizes the orientation spline, but with reduced deviation from a constant angular velocity. Details of the algorithm follow.

3.1. Position spline

The first stage in interpolating a tool path is calculation of the near arc-length quintic polynomial position spline through the set of position vectors, $\mathbf{p}_1, \ldots, \mathbf{p}_{n+1}$. The *i*th segment of the position spline is expressed in the form

$$\mathbf{P}_i(u) = \mathbf{c}_{1_i} + \mathbf{c}_{2_i}u + \mathbf{c}_{3_i}u^2 + \mathbf{c}_{4_i}u^3 + \mathbf{c}_{5_i}u^4 + \mathbf{c}_{6_i}u^5, \quad u \in [0, l_i] \tag{5}$$

where u is the position parameter, and l_i is the range of u.

Determination of the quintic coefficients, $\mathbf{c}_{1_i}, \ldots, \mathbf{c}_{6_i}$, in Eq. (5) is based on the technique described in Ref. [23], but with an improvement. In their work, Wang and Yang first estimate l_i with

$$l_i = |\mathbf{p}_i - \mathbf{p}_{i+1}| \tag{6}$$

and then interpolate a C^2 cubic polynomial spline with natural parameterization through the data points with

$$\begin{bmatrix} \frac{l_1}{3} & \frac{l_1}{6} & & & & & & 0 \\ \frac{l_1}{6} & \frac{l_1+l_2}{3} & \frac{l_2}{6} & & & & & \\ & & \ddots & & & & & \\ & & \frac{l_{i-1}}{6} & \frac{l_{i-1}+l_i}{3} & \frac{l_i}{6} & & & \\ & & & & \ddots & & & \\ & & & & \frac{l_n}{6} & \frac{l_n+l_{n+1}}{3} & \frac{l_n+1}{6} \\ 0 & & & & & \frac{l_{n+1}}{6} & \frac{l_{n+1}}{3} \end{bmatrix} \begin{bmatrix} (\mathbf{p}''_1)^{\mathrm{T}} \\ \vdots \\ (\mathbf{p}''_i)^{\mathrm{T}} \\ \vdots \\ (\mathbf{p}''_{n+1})^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \left(\frac{\mathbf{p}_2-\mathbf{p}_1}{l_1} - \mathbf{p}'_1\right)^{\mathrm{T}} \\ \left(\frac{\mathbf{p}_3-\mathbf{p}_2}{l_2} - \frac{\mathbf{p}_2-\mathbf{p}_1}{l_1}\right)^{\mathrm{T}} \\ \vdots \\ \left(\frac{\mathbf{p}_{i+1}-\mathbf{p}_i}{l_i} - \frac{\mathbf{p}_i-\mathbf{p}_{i-1}}{l_{i-1}}\right)^{\mathrm{T}} \\ \vdots \\ \left(\frac{\mathbf{p}_{n+1}-\mathbf{p}_n}{l_n} - \frac{\mathbf{p}_n-\mathbf{p}_{n-1}}{l_{n-1}}\right)^{\mathrm{T}} \\ \left(\mathbf{p}'_{n+1} - \frac{\mathbf{p}_{n+1}-\mathbf{p}_n}{l_n}\right)^{\mathrm{T}} \end{bmatrix} \tag{7}$$

The first and second derivatives of the cubic spline at the data points are extracted and used to construct a quintic near arc-length parameterized spline. In doing so, the first derivative is normalized in order that the quintic spline is arc-length parameterized at the data points. However, the second derivative is not modified and is applied directly to the quintic spline. The coefficients of the quintic spline are then calculated from the normalized cubic spline first and second derivatives, using the equations

$$\mathbf{c}_{1_i} = \mathbf{p}_i, \qquad \mathbf{c}_{2_i} = \mathbf{p}'_i, \qquad \mathbf{c}_{3_i} = \frac{\mathbf{p}''_i}{2},$$

$$\mathbf{c}_{4_i} = \frac{10(\mathbf{p}_{i+1} - \mathbf{p}_i)}{l_i^3} - \frac{2(2\mathbf{p}'_{i+1} + 3\mathbf{p}'_i)}{l_i^2} + \frac{\mathbf{p}''_{i+1} - 3\mathbf{p}''_i}{2l_i},$$

$$\mathbf{c}_{5_i} = -\frac{15(\mathbf{p}_{i+1} - \mathbf{p}_i)}{l_i^4} + \frac{7\mathbf{p}'_{i+1} + 8\mathbf{p}'_i}{l_i^3} - \frac{2\mathbf{p}''_{i+1} - 3\mathbf{p}''_i}{2l_i^2},$$

$$\mathbf{c}_{6_i} = \frac{6(\mathbf{p}_{i+1} - \mathbf{p}_i)}{l_i^5} - \frac{3(\mathbf{p}'_{i+1} + \mathbf{p}'_i)}{l_i^4} - \frac{\mathbf{p}''_{i+1} - \mathbf{p}''_i}{2l_i^3} \tag{8}$$

Finally, the l_i are recalculated, with the condition

$$\left|\frac{\mathrm{d}}{\mathrm{d}u}\mathbf{P}_i\left(\frac{l_i}{2}\right)\right| = 1 \tag{9}$$

The improvement is made by finding the curvature of the cubic spline at the data points and applying these values in the construction of the quintic spline. This yields a spline that is significantly closer to arc-length parameterization. The improved algorithm is summarized in Fig. 3.

The curvature is found by taking the derivative of the tangent of a general curve. If $\mathbf{P}(u)$ is the general curve, its tangent is

$$\frac{\mathrm{d}}{\mathrm{d}s}\mathbf{P}(u) = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}s} = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\frac{\mathrm{d}u}{\mathrm{d}s} \tag{10}$$

By finding the length of the tangent vector $\mathrm{d}\mathbf{p}/\mathrm{d}s$ in Eq. (10) and noting that the length of the tangent vector must be unity the following result is obtained:

$$\frac{\mathrm{d}u}{\mathrm{d}s} = \frac{1}{\left|\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\right|} \tag{11}$$

Combining Eq. (10) and Eq. (11) results in the well-known equation for calculating the tangent:

$$\frac{\mathrm{d}}{\mathrm{d}s}\mathbf{P}(u) = \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}s} = \frac{\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}}{\left|\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\right|}. \tag{12}$$

The curvature is obtained by differentiating Eq. (12). This yields

$$\frac{\mathrm{d}^2}{\mathrm{d}s^2}\mathbf{P}(u) = \frac{\mathrm{d}^2\mathbf{p}}{\mathrm{d}s^2} = \frac{\dfrac{\mathrm{d}^2\mathbf{p}}{\mathrm{d}u^2}\dfrac{\mathrm{d}u}{\mathrm{d}s}}{\left|\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\right|} - \frac{\left(\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\dfrac{\mathrm{d}^2\mathbf{p}}{\mathrm{d}u^2}\right)\dfrac{\mathrm{d}u}{\mathrm{d}s}\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}}{\left|\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\right|^3}. \tag{13}$$

Combining Eq. (13) with Eqs. (11) and (12) results in a relation for curvature depending only on the first and second derivative of the general curve.

$$\frac{\mathrm{d}^2\mathbf{p}}{\mathrm{d}s^2} = \frac{\left(\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\cdot\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\right)\dfrac{\mathrm{d}^2\mathbf{p}}{\mathrm{d}u^2} - \left(\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\cdot\dfrac{\mathrm{d}^2\mathbf{p}}{\mathrm{d}u^2}\right)\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}}{\left(\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\cdot\dfrac{\mathrm{d}\mathbf{p}}{\mathrm{d}u}\right)^2} \tag{14}$$

Wright et al. [17,21,22] also made an improvement to the

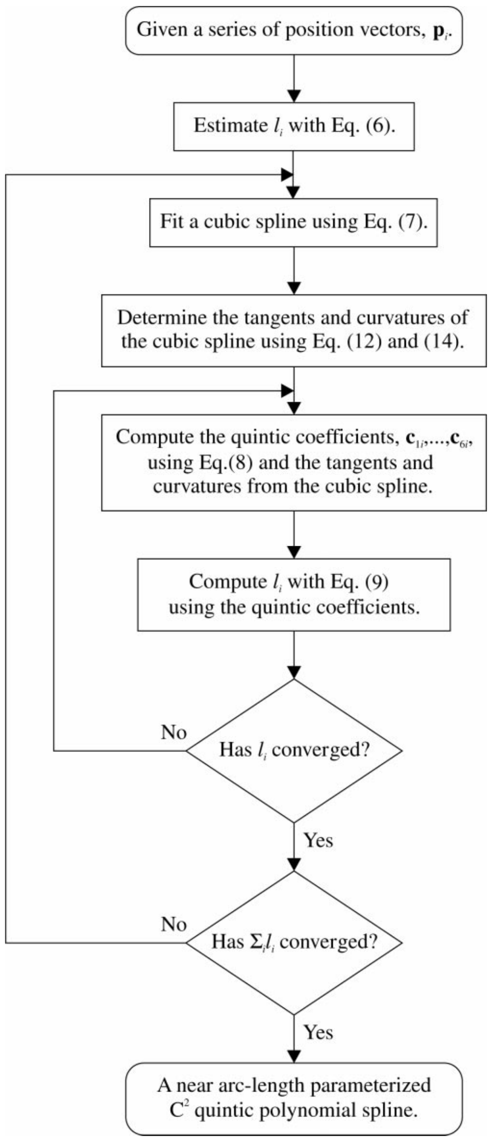
Given a series of position vectors, $\mathbf{p}_i$.
Estimate l_i with Eq. (6).
Fit a cubic spline using Eq. (7).
Determine the tangents and curvatures of the cubic spline using Eq. (12) and (14).
Compute the quintic coefficients, $\mathbf{c}_{1i},\ldots,\mathbf{c}_{6i}$, using Eq.(8) and the tangents and curvatures from the cubic spline.
Compute l_i with Eq. (9) using the quintic coefficients.
Has l_i converged?
No
Yes
Has $\Sigma_i l_i$ converged?
No
Yes
A near arc-length parameterized C^2 quintic polynomial spline.

Fig. 3. Proposed algorithm for computing near arc-length parameterized C^2 continuous polynomial splines.

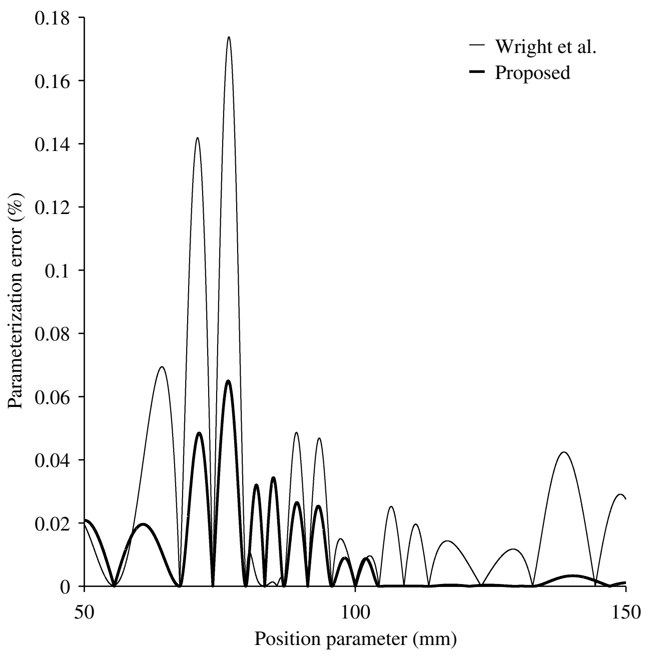

Fig. 4. Comparison of the deviation from arc-length parameterization of the proposed quintic C^2 and Wright et al. C^3 continuous splines.

arc-length parameterization of the quintic spline by enforcing C^3 continuity at the junctions of the spline. Their results are compared to the above-proposed method in Fig. 4. The example shown is for a portion of the tool path shown in Fig. 5. The parameterization error is calculated as $|1 - \|\mathrm{d}\mathbf{p}/\mathrm{d}u\| \times 100\%$. In Fig. 4 the average parameterization error for the C^3 continuous spline is 0.0091% and the average error for the improved C^2 continuous spline is 0.0048%, or approximately half as large.

3.2. Orientation spline

The orientation spline interpolates the orientation portion of the tool path vectors. Since the orientation of the tool is defined by a unit vector, the orientation spline must lie on the surface of the unit sphere. Construction of a near arc-length parameterized orientation spline is discussed in two parts below. First the spherical Bézier spline, which lies on a unit sphere, is introduced. Interpolating a set of unit vectors with a near arc-length parameterized spherical Bézier spline is then discussed.

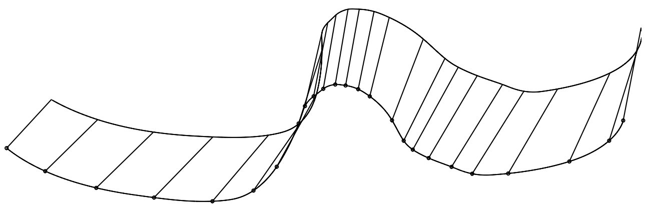

Fig. 5. Isometric view of the example tool path generated by the proposed algorithm. The dots indicate the tool tip position and straight line segments indicate the tool axis orientation.

3.2.1. Spherical Bézier spline

To construct a curve which lies on the unit sphere using the de Casteljau form of the Bézier curve [2,13], we replace the linear interpolation with a spherical interpolation. The spherical interpolation curve, $\mathbf{S}(v)$, interpolates the two unit vectors, $\mathbf{q}_1$ and $\mathbf{q}_2$ as follows [3]:

$$\mathbf{S}(v) = \frac{\mathbf{q}_1 \sin(\theta(1-v)) + \mathbf{q}_2 \sin(\theta v)}{\sin(\theta)}, \qquad \theta = \arccos(\mathbf{q}_1 \cdot \mathbf{q}_2), \qquad |\mathbf{q}_1| = |\mathbf{q}_2| = 1. \tag{15}$$

It can be shown that

$$|\mathbf{S}(v)| = 1, \qquad v \in [0,1] \tag{16}$$

and

$$\mathbf{S}(0) = \mathbf{q}_1, \qquad \mathbf{S}(1) = \mathbf{q}_2 \tag{17}$$

Given $j+1$ control points $\mathbf{d}_k$, a spherical Bézier curve, $\mathbf{B}_k^j(v)$, of degree j is given by

$$\mathbf{B}_k^j(v) = \begin{cases} \mathbf{d}_k, & \text{if } j = 0 \\ \dfrac{\mathbf{B}_k^{j-1}(v)\sin(\theta(1-v)) + \mathbf{B}_{k+1}^{j-1}(v)\sin(\theta v)}{\sin(\theta)}, & \text{if } j > 0, \end{cases}$$

$$|\mathbf{d}_k| = 1, \qquad \theta = \arccos(\mathbf{B}_k^{j-1}(v) \cdot \mathbf{B}_{k+1}^{j-1}(v)). \tag{18}$$

The ith segment of a spherical Bézier spline is given by

$$\mathbf{Q}_i^j(v) = \mathbf{B}_{1_i}^j\left(\frac{v}{\lambda_i}\right), \; v \in [0, \lambda_i], \; i = 1, \ldots, n \tag{19}$$

where v is the orientation parameter, and λ_i is the range of v. This equation is the orientation analogue of the position spline, Eq. (5). It has six unknown vectors $\mathbf{d}_{1_i}, \ldots, \mathbf{d}_{6_i}$ and an segment length λ_i. Similar to the Bézier curve in Euclidean space, the spherical Bézier spline has the following properties:

1. Eq. (17) ensures endpoints of a segment (at $v = 0$ and $v = \lambda$) correspond to the first and last control point, $\mathbf{d}_1$ and $\mathbf{d}_{j+1}$, respectively.
2. The first derivatives at the endpoints of a segment depend on only the nearest two control points, $\mathbf{d}_1$ and $\mathbf{d}_2$ at $v = 0$, and $\mathbf{d}_j$ and $\mathbf{d}_{j+1}$ at $v = \lambda$. To ensure C^1 continuity at the junctions, the four local control points $\mathbf{d}_{j_i}$, $\mathbf{d}_{(j+1)_i} = \mathbf{d}_{1_{i+1}}$ and $\mathbf{d}_{2_{i+1}}$ must lie on a great circle of the unit sphere.
3. Similarly, the second derivatives at the endpoints of a segment depend only on the nearest three control points.

Equations for determining the first and second derivatives at the endpoints are given in Appendix A.

3.2.2. Interpolating the orientation spline

The object of the orientation spline is to interpolate the set of $n+1$ orientation vectors, $\mathbf{q}_1, \ldots, \mathbf{q}_{n+1}$, with a quintic ($j = 5$) spherical Bézier spline of Eq. (19). As previously

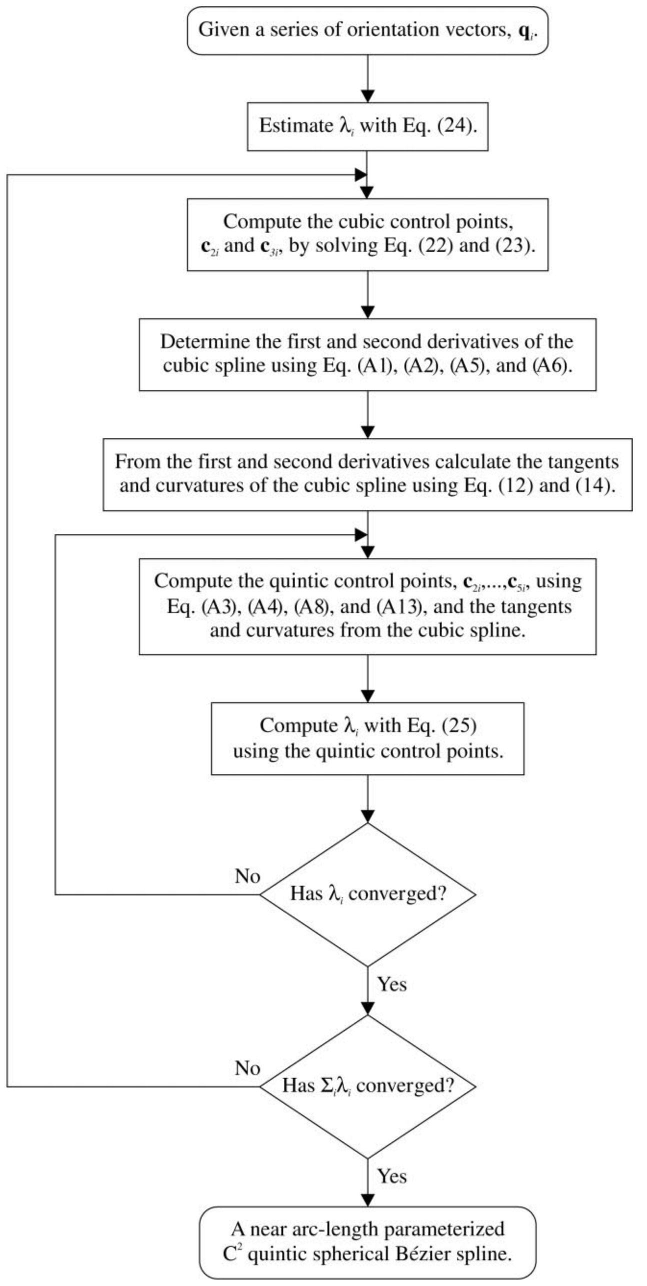
Given a series of orientation vectors, $\mathbf{q}_i$.
Estimate λ_i with Eq. (24).
Compute the cubic control points, $\mathbf{c}_{2i}$ and $\mathbf{c}_{3i}$, by solving Eq. (22) and (23).
Determine the first and second derivatives of the cubic spline using Eq. (A1), (A2), (A5), and (A6).
From the first and second derivatives calculate the tangents and curvatures of the cubic spline using Eq. (12) and (14).
Compute the quintic control points, $\mathbf{c}_{2i}$,...,$\mathbf{c}_{5i}$, using Eq. (A3), (A4), (A8), and (A13), and the tangents and curvatures from the cubic spline.
Compute λ_i with Eq. (25) using the quintic control points.
Has λ_i converged?
No
Yes
Has $\Sigma_i\lambda_i$ converged?
No
Yes
A near arc-length parameterized C^2 quintic spherical Bézier spline.

Fig. 6. Proposed algorithm for computing near arc-length parameterized C^2 continuous quintic spherical Bézier splines.

discussed, the interpolated spline must have both the near arc-length parameterization property and C^2 continuity. The same algorithm used for the position spline is applied to the orientation spline, with adjustment for the curve type.

The algorithm for computing the orientation spline is summarized in Fig. 6. It combines two distinct procedures. The first is the fitting of a C^2 quintic spline. The second is the improvement of the parameterization of the spline such that the parameterization approaches arc-length parameterization.

3.2.3. Fitting the quintic spline

Fitting the C^2 quintic spline involves two steps: computing a C^2 cubic spline and then using information from the cubic spline to construct the quintic spline.

The cubic spline is computed by solving the set of nonlinear equations

$$\mathbf{Q}_i^j(0) = \mathrm{q}_i, \qquad i = 1, \dots, n \tag{20}$$

$$\mathbf{Q}_i^j(\lambda_i) = \mathbf{q}_{i+1}, \qquad i = 1, \dots, n \tag{21}$$

$$\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{Q}_i^j(\lambda_i) = \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{Q}_{i+1}^j(0), \qquad i = 1, \dots, n-1 \tag{22}$$

$$\frac{\mathrm{d}^2}{\mathrm{d}v^2}\mathbf{Q}_i^j(\lambda_c) = \frac{\mathrm{d}^2}{\mathrm{d}v^2}\mathbf{Q}_{i+1}^j(0), \qquad i = 1, \dots, n-1 \tag{23}$$

for the unknown control points $\mathbf{d}_{2_i}$ and $\mathbf{d}_{3_i}$. The control points $\mathbf{d}_{1_i}$ and $\mathbf{d}_{4_i}$ correspond to the data points $\mathbf{q}_i$ and are therefore known. Broyden's method [14] has been successfully used to solve this set of equations, but it requires the first derivatives at the ends of the spline and initial estimates of the control points. If estimates of the tangents, $\mathbf{q}_i'$, can be obtained first then the control points can be determined from Eqs. (A3) and (A4).

Estimates for these first derivatives are obtained by interpolating successive sets of three data points with a quadratic spherical Bézier curve. The quadratic Bézier curve has three control points. The first and third control point corresponds to the first and third data point. The middle control point is computed using Broyden's method with the middle data point as the initial estimate. After each iteration of this method the middle control point is normalized to a unit vector. From these quadratic curves the tangents necessary to compute the cubic spline are extracted.

Once a cubic spline has been fitted, the first and second derivatives of the cubic spline at the data points are extracted. These are used to compute the tangents and curvatures in the same way as for the position spline using Eqs. (12) and (14). Using the tangents and curvatures from the cubic spline the unknown quintic control points, $\mathbf{d}_{2_i}$, $\mathbf{d}_{3_i}$, $\mathbf{d}_{4_i}$, and $\mathbf{d}_{5_i}$, are found (see Appendix A).

3.2.4. Parameterization improvement

An initial estimate of λ_i is obtained in a similar manner to the chord length for position splines

$$\lambda_i = \arccos(\mathbf{q}_i \cdot \mathbf{q}_{i+1}) \tag{24}$$

This equation computes the geodesic distance on a unit sphere between the two unit vectors $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$.

The first improvement to the parameterization is made in the selection of the tangents and curvatures of the quintic spline by modifying first and second derivatives extracted from the cubic spline using Eq. (12) and Eq. (14).

The second improvement to the parameterization is made once the quintic spline control points are known. By forcing the unit tangency property at the middle of each segment, the segment lengths are obtained and parameterization of the spline is improved:

$$\left|\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{Q}_i^5\left(\frac{\lambda_i}{2}\right)\right| = 1 \tag{25}$$

This equation can be solved efficiently for λ_i with Brent's method [14].

However, the quintic control points depend on λ_i. Once λ_i is recomputed with Eq. (25) the control points are no longer valid and Eqs. (22) and (23) are violated. Therefore it is necessary to iterate through these two steps until the change in λ_i is less than a specified threshold value.

Similarly, the cubic and quintic splines are refitted until $\sum_i \lambda_i$ changes by less than a specified threshold.

3.3. Orientation reparameterization spline

The orientation reparameterization spline ensures coordinated motion by the position and orientation splines. In three-axis machining, near arc-length parameterized splines are used to achieve constant feedrate coordinated motion. The length of the spline therefore determines the distance the tool travels, and machining time required. In multi-axis machining, maintaining a constant feedrate along the position spline and a constant angular velocity along the orientation spline would result in uncoordinated motion. This is because the two splines have different lengths. Merely scaling the length of the orientation or position spline is insufficient since the spacing between matching points on the two splines is unlikely to be proportional. Consequently, the orientation spline must be reparameterized to meet the coordinated motion requirement.[1]

Coordinated motion is achieved by reparameterizing the orientation spline with an orientation reparameterization spline. The reparameterization spline relates the position spline parameter, u, to the orientation spline parameter, v, for each segment:

$$v = V_i(u) \tag{26}$$

$$h_i^{(w+1)} = \frac{c_i - a_{i-1}h_{i-1}^{(w+1)} - a_i h_{i+1}^{(w)} + [(c_i - a_{i-1}h_{i-1}^{(w+1)} - a_i h_{i+1}^{(w)})^2 + 4(a_{i-1}+a_i)b_i]^{1/2}}{2(a_{i-1}+a_i)}, \qquad i = 2,\ldots,n \tag{36}$$

For each segment of the spline coordinated motion is ensured with

$$V_i(0) = 0 \tag{27}$$

$$V_i(l_i) = \lambda_i. \tag{28}$$

The reparameterization spline is formulated as a C^2 monotonic rational quadratic spline based on the work of Fritsch and Carlson [4], and Delbourgo and Gregory [1,6]. The spline is interpolated through the knots of the position and orientation splines and is given as

$$V_i(u) = \frac{\mu_{i+1}u^2 + l_i\lambda_i^{-1}(\mu_{i+1}h_i + \mu_i h_{i+1})u(l_i - u) + \mu_i(l_i - u)^2}{u^2 + l_i\lambda_i^{-1}(h_i + h_{i+1})u(l_i - u) + (l_i - u)^2} - \mu_i,$$

$$i = 1,\ldots,n, \tag{29}$$

$$\mu_1 = 0,$$

$$\mu_i = \sum_{e=1}^{i-1}\lambda_e \qquad i = 2,\ldots,n+1.$$

The h_i are the first derivatives of the spline at the segment junctions and are unknown. They are solved for in the following procedure. Let

$$a_i = \frac{1}{\lambda_i}, \qquad i = 1,\ldots,n \tag{30}$$

$$b_i = \frac{\lambda_{i-1}}{l_{i-1}^2} + \frac{\lambda_i}{l_i^2}, \qquad i = 2,\ldots,n \tag{31}$$

$$c_i = \frac{1}{l_{i-1}} + \frac{1}{l_i}, \qquad i = 2,\ldots,n. \tag{32}$$

Initialize the h_i with

$$h_1 = \frac{\lambda_1 l_2(l_1 + l_2 + 1) - \lambda_2 l_1}{l_1 l_2(l_1 + l_2)} \tag{33}$$

$$h_i = \left(\frac{b_i}{a_{i-1} + a_i}\right)^{1/2}, \qquad i = 2,\ldots,n \tag{34}$$

$$h_{n+1} = \frac{\lambda_n l_{n-1}(l_n + l_{n-1} + 1) - \lambda_{n-1} l_n}{l_n l_{n-1}(l_n + l_{n-1})}. \tag{35}$$

Using Gauss–Seidel iterate until

$$\max_i |h_i^{(w+1)} - h_i^{(w)}| \le \epsilon. \tag{37}$$

4. Performance evaluations

A multi-axis tool path based on a fan profile [22,23] is used as an example. The original fan profile is scaled by three, and then offset towards the center of the fan profile by the tool radius, 12.7 mm. The resulting spline is duplicated, translated in the positive z direction by 25.4 mm, and rotated by 0.1745 rad (10°). These two splines form a ruled surface.

[1] Alternatively the position spline could be reparametrized should a constant angular velocity be desired.

Table 1
Interpolatory data for the example tool path

p_x (mm)	p_y (mm)	p_z (mm)	q_x	q_y	q_z
113.560775	7.735266	−2.209314	−0.107258	0.624902	0.773300
117.864949	−10.950074	−0.974065	−0.003002	0.653008	0.757345
115.502860	−34.808781	0.779567	0.135034	0.648777	0.748902
104.086026	−55.840098	2.682644	0.263922	0.596758	0.757776
95.225652	−63.959571	4.073524	0.317154	0.551822	0.771301
88.820589	−65.972318	6.021818	0.329520	0.516103	0.790603
80.162816	−65.096397	7.031464	0.326785	0.478040	0.815285
72.478246	−61.109537	6.863103	0.313696	0.446069	0.838223
65.985754	−54.666365	6.143037	0.288698	0.416448	0.862105
54.251993	−39.568096	4.931174	0.217010	0.357492	0.908354
38.038952	−23.111532	3.536073	0.129479	0.261821	0.956391
31.679054	−18.711329	3.017623	0.105499	0.220895	0.969575
26.192567	−16.781277	2.454873	0.096827	0.184942	0.977968
22.337845	−16.486398	1.907427	0.097931	0.159739	0.982290
18.769843	−17.937940	0.258718	0.110602	0.137889	0.984253
17.004164	−21.333422	−1.375463	0.133457	0.127499	0.982819
16.763098	−27.082862	−2.573817	0.171104	0.127727	0.976939
20.059322	−38.210737	−3.573619	0.238991	0.153346	0.958837
26.991221	−67.786318	−5.570258	0.399932	0.201303	0.894165
28.080737	−86.648047	−6.205088	0.487937	0.208205	0.847684
21.813110	−103.571377	−4.442501	0.567908	0.182153	0.802683
6.156323	−114.179587	−2.044104	0.628723	0.098457	0.771372
−12.661488	−117.988771	−0.872310	0.654180	−0.006643	0.756310
−31.816238	−116.324006	0.514512	0.651997	−0.117213	0.749107
−49.438878	−108.784390	2.089537	0.618930	−0.223905	0.752856

The tool path is generated by offsetting points in a direction normal to the design surface for a distance equal to the tool radius. Because of symmetry, only one quarter of the fan profile was used. The resulting ruled surface tool path is shown in Fig. 5. Twenty-five points were extracted from the tool path for the position, orientation, and reparameterization splines as shown in Table 1 and Fig. 7.

The position spline (see Fig. 7) is almost identical to the spline generated by the algorithm of Wang and Wright [22] except that there is some displacement in the z direction. As shown in Fig. 8 the average parameterization error (calculated as earlier) has been reduced to 0.013% from 0.038% by the proposed algorithm. The average parameterization error of the orientation spline is even less 0.0033%.

The example tool path of Fig. 5 was simulated using a constant feedrate of 400 mm/min (6.67 mm/s). To evaluate the tool path at a constant feedrate the feedrate must be related to u:

$$u = ft. \tag{38}$$

The equations for the motion of the tool relative to the part reference frame at constant feedrate are given by the following equations:

$$\mathbf{t}(t) = \begin{bmatrix} \mathbf{P}_i(ft) \\ \mathbf{Q}_i(V_i(ft)) \end{bmatrix} \tag{39}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{t}(t) = \begin{bmatrix} \dfrac{\mathrm{d}}{\mathrm{d}t}\mathbf{P}_i(ft) \\ \dfrac{\mathrm{d}}{\mathrm{d}V_i}\mathbf{Q}_i(V_i(ft))\dfrac{\mathrm{d}}{\mathrm{d}t}V_i(ft) \end{bmatrix} \tag{40}$$

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{t}(t) = \begin{bmatrix} \dfrac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{P}_i(ft) \\ \dfrac{\mathrm{d}^2}{\mathrm{d}V_i^2}\mathbf{Q}_i(V_i(ft))\left(\dfrac{\mathrm{d}}{\mathrm{d}t}V_i(ft)\right)^2 + \dfrac{\mathrm{d}}{\mathrm{d}V_i}\mathbf{Q}_i(V_i(ft))\dfrac{\mathrm{d}^2}{\mathrm{d}t^2}V_i(ft) \end{bmatrix}. \tag{41}$$

The position spline components are shown in Fig. 9 and orientation spline components are shown in Fig. 10. These figures demonstrate that the displacement, and the first and second derivatives of the two splines are continuous. The level of continuity is comparable to Ref. [23].

Fig. 11 shows the feedrate and angular velocity for the example tool path. The feedrate illustrates the approximation to near constant feedrate within a range of approximately 1 mm/min or no greater than 0.018% error for the feedrate of 400 mm/min.

By contrast, the angular velocity is overwhelmed by the reparameterization and is therefore non-constant. However, in Fig. 12 the proposed algorithm is compared to the work of Ge and Srinivasan [5,20] which essentially assumes $\lambda_i = l_i$. The orientation spline based on this assumption therefore need only be a cubic spherical spline and consequently is not near arc-length parameterized. Hence fluctuations

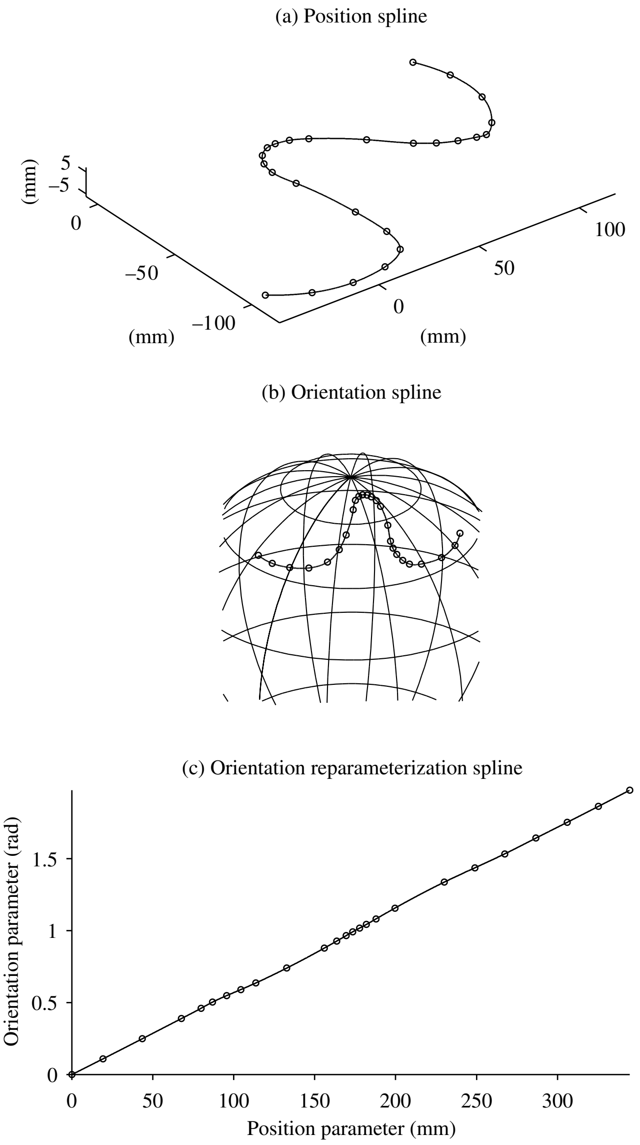

Fig. 7. Component splines of the example tool path: (a) tool tip position spline; (b) tool axis orientation spline and (c) orientation reparameterization spline.

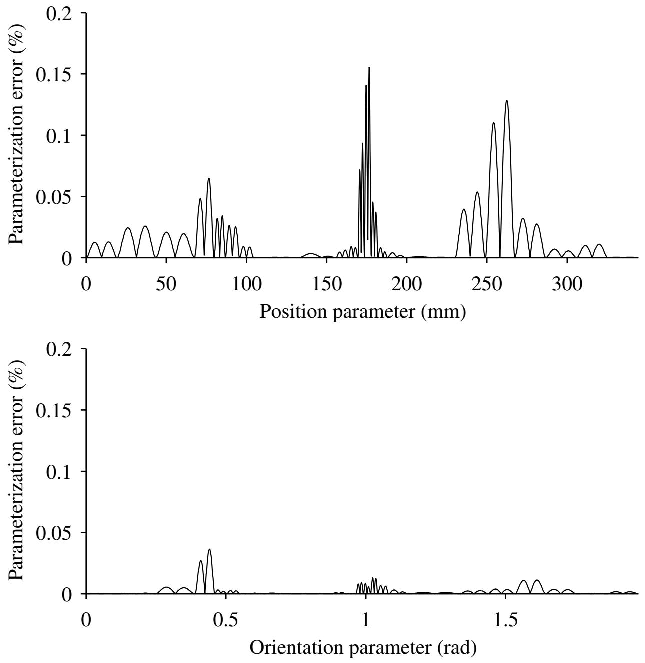

Fig. 8. Deviation from arc-length parameterization of position and orientation splines for the example tool path.

between interpolatory points appear in the angular velocity. If instead a near arc-length parameterized orientation spline is used along with a reparameterization spline the result is an improvement in the smoothness of the angular velocity and on average a reduction in angular acceleration of 11.6% and a reduction of 39.3% at the peak near 12 s (see Fig. 12). The reason for the improvement is the reduction of the fluctuations between interpolatory points in the speed of the orientation spline by the application of the near arc-length parameterization technique.

Below is an example of inverse kinematics that further demonstrates the effect of the proposed tool path interpolation algorithm on the commanded motion of a multi-axis machine tool. A typical tilting rotary table type five-axis CNC machine tool has three linear axes, X, Y, and Z, and two rotary axes A, and C. For this type of CNC machine tool the simplified equations for transforming the tool coordinates, $\mathbf{t}$, into joint space is given below [11,16]:

$$A = -\arccos(q_z) \tag{42}$$

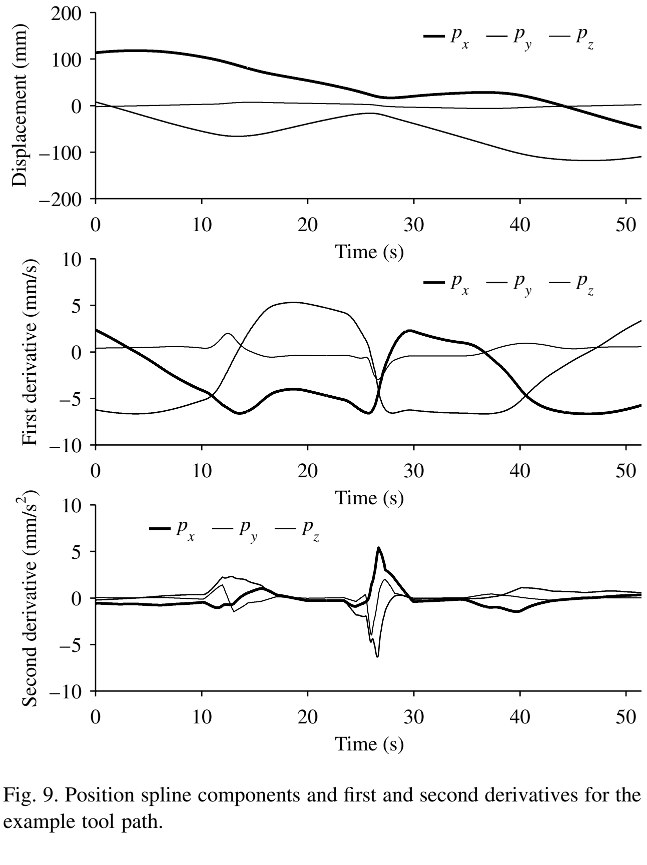

Fig. 9. Position spline components and first and second derivatives for the example tool path.

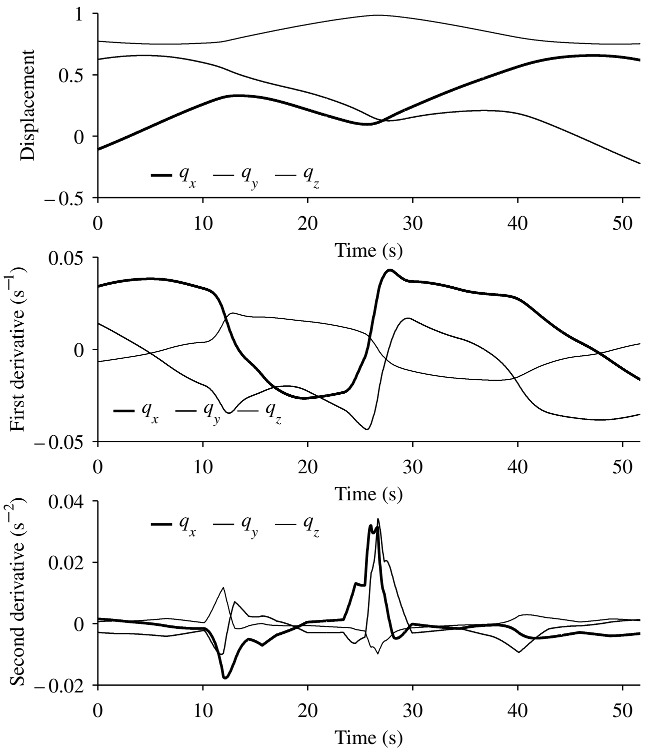

Fig. 10. Orientation spline components and first and second derivatives for the example tool path.

$$C = \arctan(q_x, q_y) + \pi \tag{43}$$

$$X = \frac{q_x(p_y - d_y) - q_y(p_x - d_x)}{\sqrt{1 - q_z^2}} \tag{44}$$

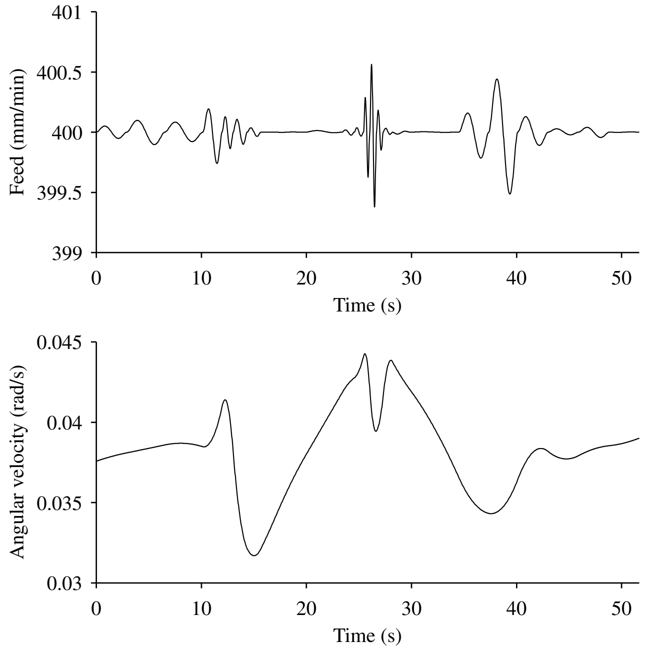

Fig. 11. Feed and angular velocity for the example tool path.

$$Y = \frac{(1 - q_z^2)(p_z - d_z) - q_z(q_x(p_y - d_y) + q_y(p_x - d_x))}{\sqrt{1 - q_z^2}} \tag{45}$$

$$Z = (q_x(p_y - d_y) + q_y(p_x - d_x)) + q_z(p_z - d_z). \tag{46}$$

The vector $[\,d_x \quad d_y \quad d_z\,]$ specifies the coordinates of the part relative to the tool in the workspace of the machine tool. For the example tool path, this vector was assigned the value $[\,0 \quad 0 \quad -140.8174\,]$ mm.

To compute the velocity and acceleration of the individual axes the chain rule of calculus is applied. For the axis the equations are:

$$\frac{dA}{dt} = \frac{dA}{dq_z}\dot{q}_z \tag{47}$$

$$\frac{d^2A}{dt^2} = \frac{d^2A}{dq_z^2}\dot{q}_z + \frac{dA}{dq_z}\ddot{q}_z. \tag{48}$$

Similar results are obtained for the remaining four axes.

Results are shown in Figs. 13 and 14 for the linear and rotary axes, respectively. It should be noted that the velocity and acceleration of the C axis demonstrates a marked spike. This occurs because the tool orientation nears a kinematic singularity in this region. These simulations demonstrate C^2 continuity for all axes which compares favorably with the three-axis results of Yang and Wang [25], and Wang and Wright [22].

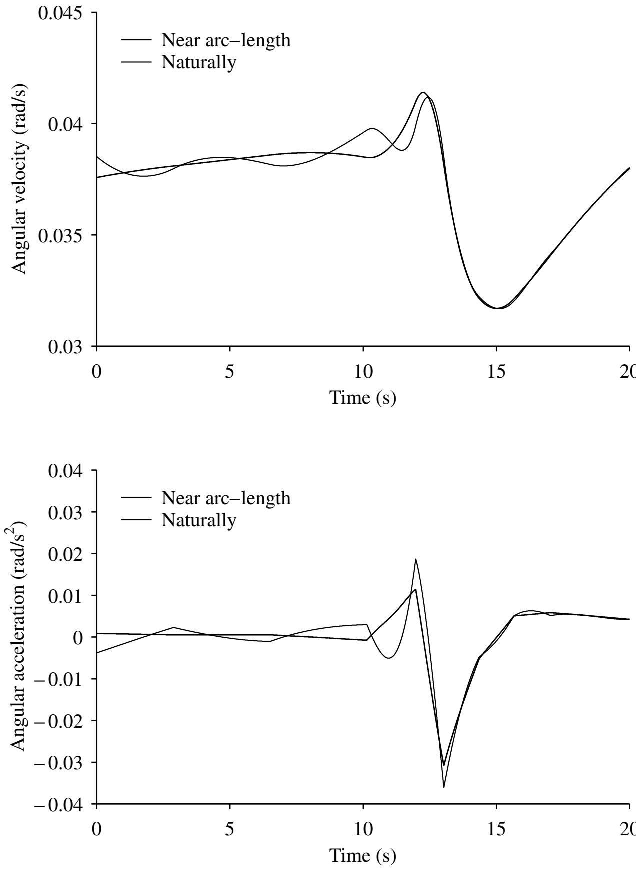

Fig. 12. Comparison of angular velocity and angular acceleration of a near arc-length and a naturally ($\lambda_i = l_i$) parameterized orientation spline (for clarity only the first 20 s of the example tool path is shown).

The acceleration of the linear and rotary axes does not exceed the range ± 10 mm/s^2 and ± 0.2 rad/s^2, respectively. This is a much smaller range than the accelerations of $\pm 30{,}000$ mm/s^2 reported by Yang and Wang [25] for linear interpolators. Fig. 15 shows photographs of the successfully machined part.

5. Conclusion

This paper introduces the spherical Bézier spline for interpolating the orientation unit vectors of a multi-axis tool path. This spline is combined with a position spline and reparameterization spline to produce a tool path interpolator. As shown, the proposed interpolator generates a tool path, which has the property of near constant feed and reduced angular acceleration. The axis commands are demonstrated to be C^2 continuous and display velocities and accelerations similar to previously published research on near arc-length parameterized three-axis interpolators, but with reduced angular acceleration. Additionally, an improvement in the unit tangency property of the position spline is demonstrated in comparison to previous work. By interpolating the tool path instead of the axes commands, a

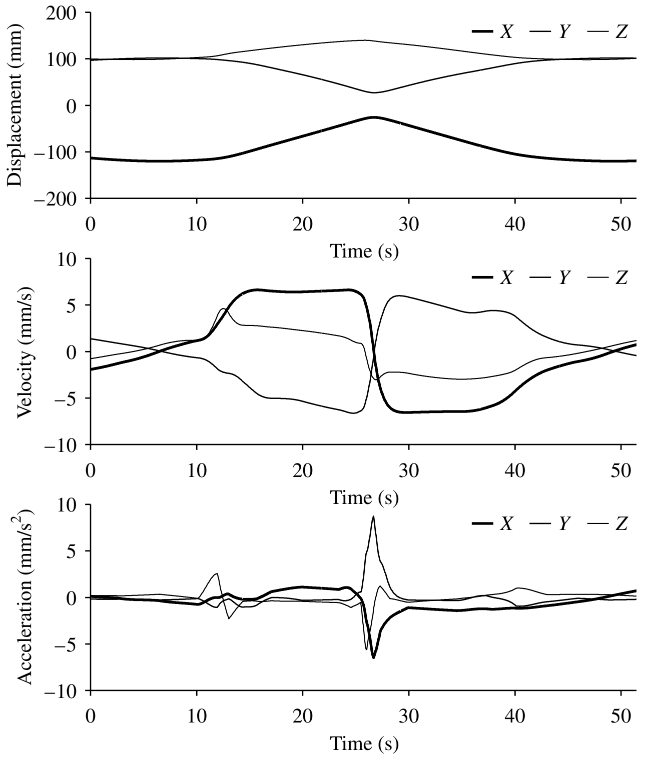

Fig. 13. Commanded motion, velocity, and acceleration, for linear axes for the example tool path.

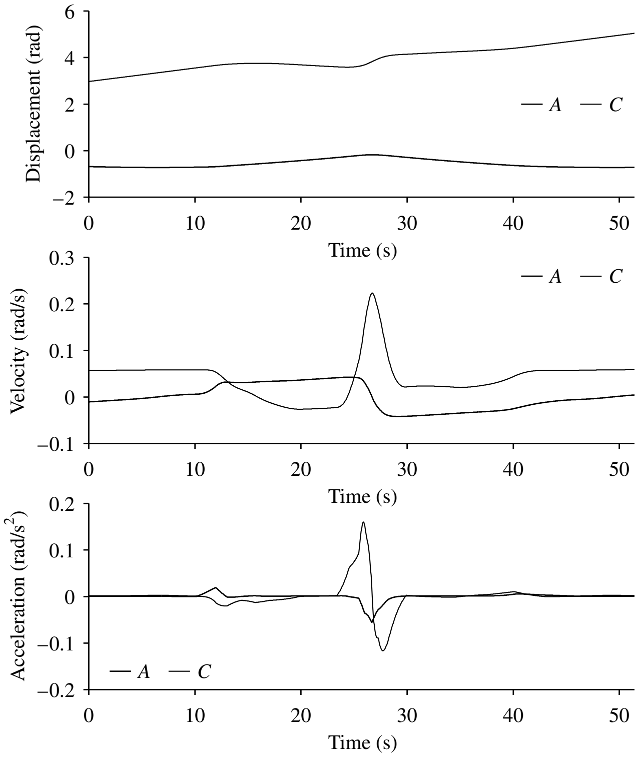

Fig. 14. Commanded motion, velocity, and acceleration, for rotary axes for the example tool path.

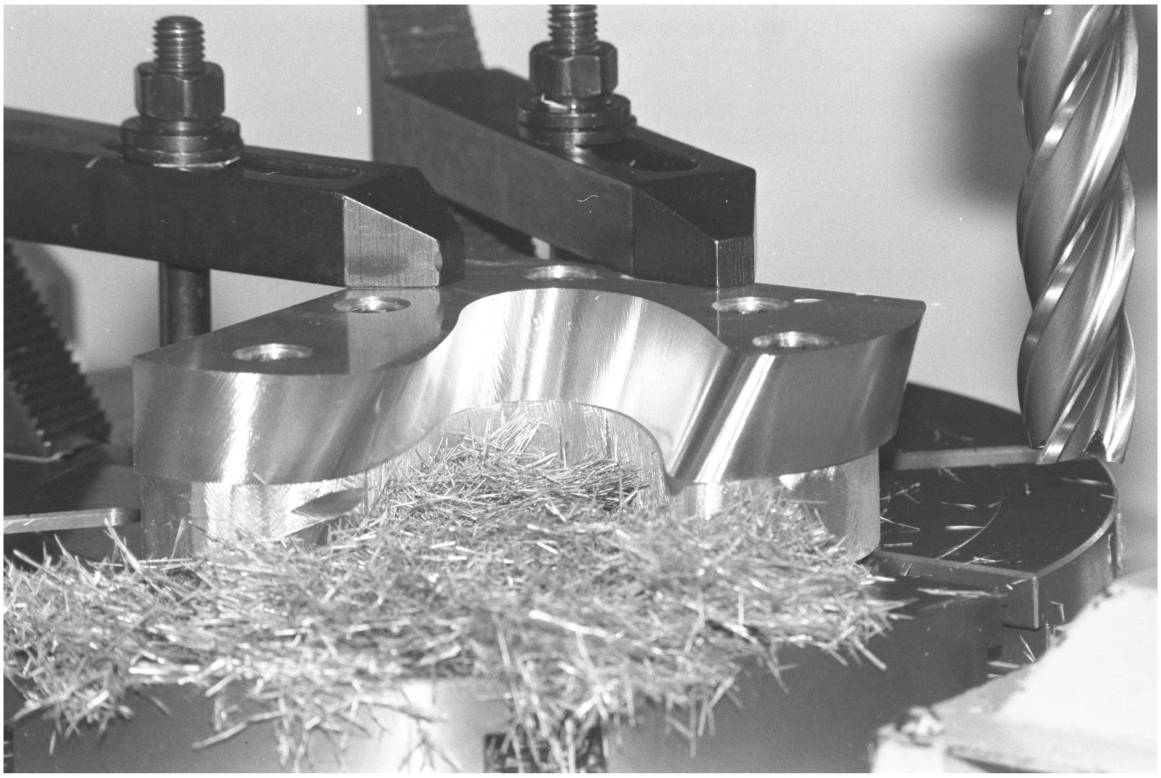
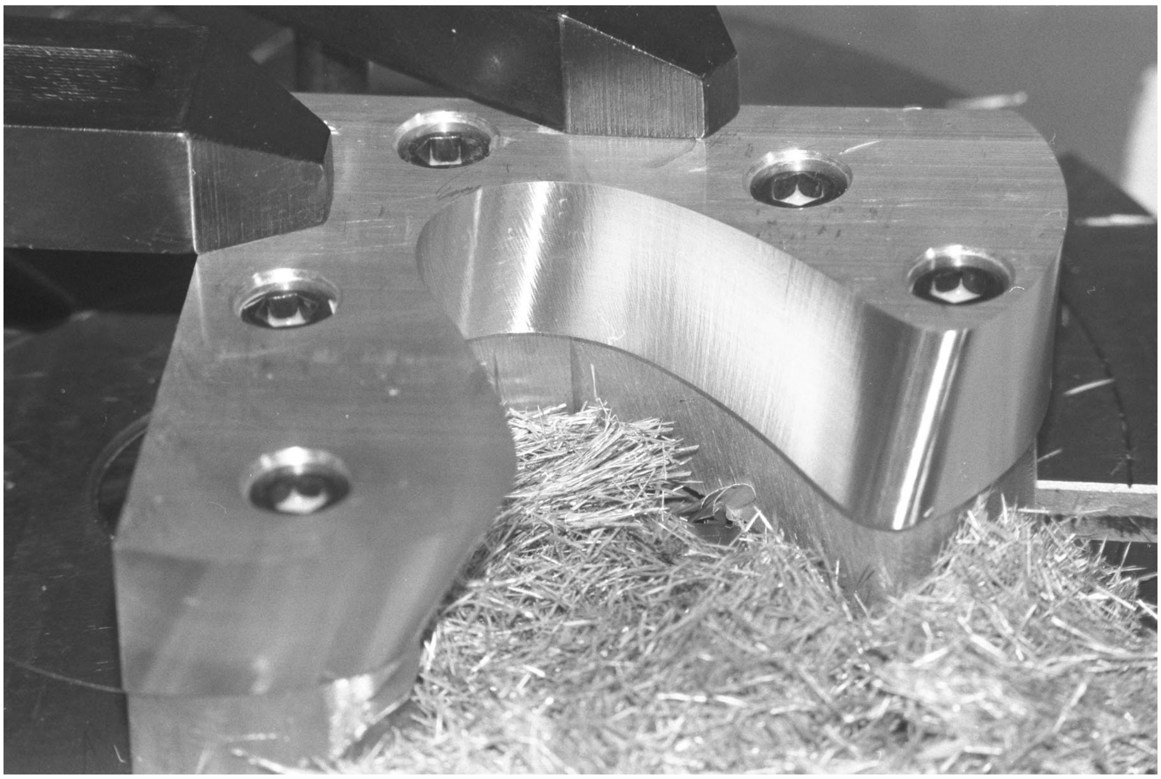

Fig. 15. Photographs of the machined part and 25.4 mm diameter flat end-mill cutter.

kinematics independent solution that can be used with five-axis machine tools, hexapods, and robotic arms, is achieved.

Acknowledgements

The authors gratefully acknowledge financial support in the form of an individual research grant from the Natural Sciences and Engineering Research Council of Canada (NSERC), by an NSERC Postgraduate Scholarship, and by an Ontario Graduate Scholarship from the Ontario Ministry of Education and Training. The helpful suggestions and additional references provided by the reviewers is very much appreciated.

Appendix A

In computing the cubic and quintic spherical Bézier spline it is necessary to calculate the first and second derivatives at the ends of each segment of the spline. It also necessary to calculate the control points in the vicinity of the end points of each segment given the tangent and curvature at the end points.

For Bézier curves the first derivative at $v = 0$ is affected by only the first and second control point. Hence, after differentiating Eq. (19) with respect to v and evaluating the result at zero (the subscript i has been dropped for clarity)

$$\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{Q}^j(0) = \mathbf{q}' = \frac{j\theta}{\lambda \sin(\theta)}(\mathbf{d}_2 - \mathbf{d}_1 \cos(\theta)), \tag{A1}$$

$$\theta = \arccos(\mathbf{d}_1 \cdot \mathbf{d}_2)$$

Similarly, when the first derivative of the spherical Bézier spline segment is evaluated at λ

$$\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{Q}^j(\lambda) = \mathbf{q}' = \frac{j\theta}{\lambda \sin(\theta)}(\mathbf{d}_{j+1} \cos(\theta) - \mathbf{d}_j), \tag{A2}$$

$$\theta = \arccos(\mathbf{d}_j \cdot \mathbf{d}_{j+1}).$$

Eqs. (A1) and (A2) can be inverted to yield

$$\mathbf{d}_2 = \mathbf{d}_1 \cos\left(\frac{\lambda|\mathbf{q}'|}{j}\right) + \frac{\mathbf{q}'}{|\mathbf{q}'|} \sin\left(\frac{\lambda|\mathbf{q}'|}{j}\right) \tag{A3}$$

and

$$\mathbf{d}_j = \mathbf{d}_{j+1} \cos\left(\frac{\lambda|\mathbf{q}'|}{j}\right) - \frac{\mathbf{q}'}{|\mathbf{q}'|} \sin\left(\frac{\lambda|\mathbf{q}'|}{j}\right), \tag{A4}$$

respectively.

Given the second derivative and the first two control points the third control point can be found because the second derivative at the end $v = 0$ is only affected by the first three control points as evidenced when Eq. (19) is differentiated twice and evaluated first at zero

$$\frac{\mathrm{d}^2}{\mathrm{d}v^2}\mathbf{Q}^j(0) = \mathbf{q}''$$

$$= \frac{j}{\lambda^2}\left(\frac{\theta}{\sin(\theta)}\left(\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_2^{j-1}(0) - \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^{j-1}(0)\cos(\theta)\right) + \frac{\mathrm{d}^2}{\mathrm{d}v^2}\mathbf{B}_1^1(0)\right.$$

$$\left. + \theta'\left(\frac{1}{\theta}\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^1(0) - \frac{1}{\sin(\theta)}\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^1(1)\right)\right), \theta$$

$$= \arccos(\mathbf{d}_j \cdot \mathbf{d}_2),$$

$$\theta' = -\frac{\mathbf{d}_1 \cdot \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_2^{j-1}(0) + \mathbf{d}_2 \cdot \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^{j-1}(0)}{\sin(\theta)} \tag{A5}$$

and then at λ

$$\frac{\mathrm{d}^2}{\mathrm{d}v^2}\mathbf{Q}^j(\lambda) = \mathbf{q}''$$

$$= \frac{j}{\lambda^2}\left(\frac{\theta}{\sin(\theta)}\left(\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_{j+1}^{j-1}(1)\cos(\theta) - \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^{j-1}(1)\right)\right.$$

$$\left. + \frac{\mathrm{d}^2}{\mathrm{d}v^2}\mathbf{B}_{j+1}^1(1) + \theta'\left(\frac{1}{\theta}\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_{j+1}^1(1) - \frac{1}{\sin(\theta)}\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_j^1(0)\right)\right), \theta$$

$$= \arccos(\mathbf{d}_j \cdot \mathbf{d}_{j+1}),$$

$$\theta' = -\frac{\mathbf{d}_j \cdot \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_{j+1}^{j-1}(1) + \mathbf{d}_{j+1} \cdot \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_j^{j-1}(1)}{\sin(\theta)}. \tag{A6}$$

Solving for the third control point is achieved in two steps. Firstly it is recognized that the unknown, $\mathbf{d}_3$, appears in Eq. (A5) only within the righthand side of

$$\mathbf{q}' = \frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_2^{j-1}(0). \tag{A7}$$

Consequently, the unknown control is obtained by first solving for $\mathbf{q}'$ and then solving for the unknown with

$$\mathbf{d}_3 = \mathbf{d}_2 \cos\left(\frac{|\mathbf{q}'|}{j-1}\right) + \frac{\mathbf{q}'}{|\mathbf{q}'|} \sin\left(\frac{|\mathbf{q}'|}{j-1}\right) \tag{A8}$$

The vector $\mathbf{q}'$ is obtained by rewriting Eq. (A5) as

$$\mathbf{q}' + (\mathbf{d}_1 \cdot \mathbf{q}')\mathbf{g} + \mathbf{h} = 0 \tag{A9}$$

$$\mathbf{g} = \frac{1}{\theta}\left(\frac{1}{\sin(\theta)}\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^1(1) - \frac{1}{\theta}\frac{\mathrm{d}}{\mathrm{d}v}\mathbf{B}_1^1(0)\right) \tag{A10}$$

$$\mathbf{h} = \frac{\sin(\theta)}{j\theta}\left(j\frac{d^2}{dv^2}\mathbf{B}_1^1(0) - \lambda^2\mathbf{q}''\right) + \mathbf{g}\left(\mathbf{d}_2\cdot\frac{d}{dv}\mathbf{B}_1^{j-1}(0)\right) - \cos(\theta)\frac{d}{dv}\mathbf{B}_1^{j-1}(0). \tag{A11}$$

To isolate $\mathbf{q}'$ note that $\mathbf{q}'$ and $\mathbf{d}_2$ must be orthogonal. Taking the dot product of $\mathbf{d}_2$ and Eq. (A7) eliminates the leftmost $\mathbf{q}'$. From the remaining equation $\mathbf{d}_1\cdot\mathbf{q}'$ is isolated and substituted back into Eq. (A7). This results in

$$\mathbf{q}' = \left(\frac{\mathbf{h}\cdot\mathbf{d}_2}{\mathbf{g}\cdot\mathbf{d}_2}\right)\mathbf{g} - \mathbf{h}. \tag{A12}$$

The result for $\mathbf{d}_{j-1}$ is obtained from the second derivative of the spherical Bézier spline evaluated at λ

$$\mathbf{d}_{j-1} = \mathbf{d}_j\cos\left(\frac{|\mathbf{q}'|}{j-1}\right) - \frac{\mathbf{q}'}{|\mathbf{q}'|}\sin\left(\frac{|\mathbf{q}'|}{j-1}\right) \tag{A13}$$

$$\frac{d}{dv}\mathbf{B}_j^{j-1}(1) = \mathbf{q}' = \left(\frac{\mathbf{h}\cdot\mathbf{d}_j}{\mathbf{g}\cdot\mathbf{d}_j}\right)\mathbf{g} - \mathbf{h} \tag{A14}$$

$$\mathbf{g} = \frac{1}{\theta}\left(\frac{1}{\theta}\frac{d}{dv}\mathbf{B}_{j+1}^1(1) - \frac{1}{\sin(\theta)}\frac{d}{dv}\mathbf{B}_j^1(0)\right) \tag{A15}$$

$$\mathbf{h} = \frac{\sin(\theta)}{j\theta}\left(\lambda^2\mathbf{q}'' - j\frac{d^2}{dv^2}\mathbf{B}_{j+1}^1(1)\right) + \mathbf{g}\left(\mathbf{d}_j\cdot\frac{d}{dv}\mathbf{B}_{j+1}^{j-1}(1)\right) - \cos\theta\frac{d}{dv}\mathbf{B}_{j+1}^{j-1}(1). \tag{A16}$$

References

[1] Delbourgo R, Gregory JA. C^2 rational quadratic spline interpolation to monotonic data. IMA J Numer Anal 1983;3:141–52.
[2] Farin G. Curves and surfaces for computer aided geometric design—a practical guide. 3rd ed.. New York: Academic Press, 1993.
[3] Foley TA, Lane DA, Nielson GM. Visualizing functions over a sphere. IEEE Comput Graphics Appl 1990;10(1):32–40.
[4] Fritsch FN, Carlson RE. Monotone piecewise cubic interpolation. SIAM J Numer Anal 1980;22(2):386–400.
[5] Ge QJ. Kinematics-driven geometric modeling: a framework for simultaneous NC tool-path generation and sculptured surface design. In: Proceedings of the 1996 13th IEEE International Conference on Robotics and Automation, 1996;2: p. 1819–24.
[6] Gregory JA, Delbourgo R. Piecewise rational quadratic interpolation to monotonic data. IMA J Numer Anal 1982;2:123–30.
[7] Jensen CG, Anderson DC. review of numerically controlled methods for finish-sculptured-surface machining. IIE Trans 1996;28:30–39.
[8] Jüttler B. Visualization of moving objects using dual quaternion curves. Comput Graphics 1994;18(3):315–26.
[9] Koren Y, Lin R-S. Five-axis surface interpolators. CIRP Ann 1995;44(1):379–82.
[10] Koren Y, Lo C-C, Shpitalni M. CNC interpolators: algorithms and analysis. In: Proc. of the ASME Prod Engng Div Manuf Sci and Engng, 1993, vol. 64. p. 83–92.
[11] Lin R-S, Koren Y. Real-time five-axis interpolators for machining ruled surfaces. In: Proc of the ASME Dyn Sys Con Div, 1994, vol. 55. p. 951–60.
[12] Lin R-S, Koren Y. Real-time interpolators for multi-axis CNC machine tools. CIRP J Mfg Sys 1996;25(2):145–9.
[13] Piegl L, Tiller W. The NURBS book. 2nd ed.. Berlin: Springer, 1997.
[14] Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical recipes in C the art of scientific computing. 2nd ed.. Cambridge: Cambridge University Press, 1992.
[15] Renner G. method of shape description for mechanical engineering practice. Comput Ind 1982;3:137–42.
[16] Rüegg A, Gygax P. Generalized kinematics model for three- to five-axis milling machines and their implementation in a CNC. CIRP Ann 1992;41(1):547–50.
[17] Schofield S, Wright PK. Open architecture controllers for machine tools, part 1: design principles. AMSE J Mfg Sci Engng 1998;120:417–24.
[18] Shpitalni S, Koren Y, Lo C-C. Realtime curve interpolators. Comput Aided Design 1994;26(11):832–8.
[19] Software Magic, Inc. IntelliPost Reference Manual, October 1994.
[20] Srinivasan LN, Ge QJ. Parametric continuous and smooth motion interpolation. ASME J Mech Design 1996;118(4):494–8.
[21] Wang F-C, Schofield S, Wright PK. Real time quintic spline interpolator for an open architecture machine tool. In: Proc of the ASME Dyn Sys Con Div, 1996, vol. 58. p. 291–97.
[22] Wang F-C, Wright PK. Open architecture controllers for machine tools, part 2: a real time quintic spline interpolator. AME J Mfg Sci Engng 1998;120:425–32.
[23] Wang F-C, Yang DCH. Nearly arc-length parameterized quintic-spline interpolation for precision machining. Comput Aided Design 1993;25(5):281–8.
[24] Yang DCH, Kong T. Parametric interpolator versus linear interpolator for precision CNC machining. Comput Aided Design 1994;26(3):225–34.
[25] Yang DCH, Wang F-C. A quintic spline interpolator for motion command generation of computer-controlled machines. ASME J Mech Design 1994;116:226–31.

Robert V. Fleisig received the BASc degree in Mechanical Engineering (1994) from the University of Waterloo and the MEng degree in Mechanical Engineering (1996) from McMaster University. He has been a PhD candidate in Mechanical Engineering at McMaster University since 1994. In his research, he has been involved in control of machine tools, and computer-aided design and manufacturing focusing on multi-axis machining.

Allan D. Spence received his BMath degree in Applied Mathematics from the University of Waterloo in 1984. He received his MASc degree in Mechanical Engineering, again from the University of Waterloo, in 1986. He received his PhD in Mechanical Engineering from the University of British Columbia in 1992. In 1994 he joined the Department of Mechanical Engineering at McMaster University, and was promoted to Associate Professor in 1999. He has industrial experience in mechanism design, vacuum and injection mould making, and computer-aided design and manufacturing, with particular emphasis on manufacturing process simulation and coordinate metrology.