A curvature optimal sharp corner smoothing algorithm for high-speed feed motion generation of NC systems along linear tool paths

Burak Sencer, Kosuke Ishizaki & Eiji Shamoto

The International Journal of Advanced Manufacturing Technology

ISSN 0268-3768

Int J Adv Manuf Technol
DOI 10.1007/s00170-014-6386-2

Springer

Int J Adv Manuf Technol
DOI 10.1007/s00170-014-6386-2

ORIGINAL ARTICLE

A curvature optimal sharp corner smoothing algorithm for high-speed feed motion generation of NC systems along linear tool paths

Burak Sencer · Kosuke Ishizaki · Eiji Shamoto

Received: 30 April 2014 / Accepted: 10 September 2014

Abstract Conventional tool paths for computer numerical-controlled (CNC) machine tools or NC positioning systems are mainly composed of linear motion segments, or the so-called G1 commands. This approach exhibits serious limitations in terms of achieving the desired part of geometry and productivity in high-speed machining. Velocity and acceleration discontinuities occur at the junction points of consecutive segments. In order to generate smooth and continuous feed motion, a geometric corner smoothing algorithm is proposed in this paper, which fits quintic B-splines to blend adjacent straight lines together. The proposed transition scheme ensures G^2 continuity transitions and optimal curvature geometry delivering fast cycle time without violating the axis acceleration limits. The cornering error is controlled analytically allowing the user to set the desired cornering tolerance. The feed profile along the corner-blended tool path is generated based on S-curve-type acceleration profile, and it is scheduled for minimum cycle time. At last, the corner-blended tool path is interpolated in real-time with minimum feed fluctuation for accurate and smooth feed motion. Proposed algorithms are implemented, and their effectiveness is tested on a CNC machine tool.

Keywords Machine tools · NC system · Trajectory generation · Bezier curve · Curvature

1 Introduction

With recent advances in part design and manufacturing technologies, computer-aided design (CAD) systems are utilized to design complex geometries compromised of smooth parametric splines such a NURBS or B-splines [1]. Direct interpolation of these curves is proven to be superior in terms of providing smoother and faster motion [2–4]. However, vast majority of numerical control (NC) units of machine tools or robotic systems do not accept reference tool paths defined by high order parametric curves. One main reason for this is the lack of an established convention for the data transfer between CAD and NC. Another reason is implementation issues within the NC. Real-time interpolation of parametric curves still exhibits some bottlenecks in terms of accurate computation of curve lengths [5] or the suppression of feed fluctuations [6, 7]. Instead, given the tolerance, computer-aided manufacturing (CAM) systems are used to generate cutter location files compromised of numerous short line segments, and NC systems simply interpret these linear "point-to-point" motion commands. Interpolating along those linear segments exhibits serious limitations in terms of achieving the desired productivity. The motion has to stop between each linear segment block due to the lack of higher order geometric continuity. This leads to elongated cycle times since the motion has to accelerate and decelerate along each linear segment. On the other hand, if linear segments are travelled in constant feed motion, there will be severe discontinuities in velocity and acceleration at junction points. This may excite motion systems' lightly damped structural modes and cause tracking errors, both of which severely deteriorate the positioning quality [8].

In an attempt to generate smooth continuous motion along series of discrete linear motion commands, two major techniques have been proposed for modern NC systems. The first one is geometric path smoothing where curve-fitting techniques are utilized to smoothen the discrete tool-path data. Advanced NC systems utilize approximate or exact spline interpolation techniques to generate continuous trajectories from batch of linear point-to-point motion commands [9, 10]. There exist several shortcomings when these methods

B. Sencer (✉) · K. Ishizaki · E. Shamoto
Nagoya University, Nagoya, Aichi, Japan
e-mail: burak0307@gmail.com

Published online: 02 October 2014

are applied. Firstly, higher order such as cubic (*3rd*) or quintic (*5th*) splines is necessary to generate C^2, i.e. acceleration continuous, tool paths from given discrete data. Nevertheless, higher order splines may exhibit "wiggles" or "curls" when fitted along densely placed short linear motion segments due to numerical conditioning. This could be addressed utilizing some heuristic methods [11, 12], or some energy functions based on pseudo *jerk* or the *curvature* can be considered [13–15] for smoothness. An alternative is to approximate the discrete tool path by parametric splines. However, in this case, the control of the fitting error becomes computationally expensive for real-time implementation.

Next, "corner blending," also called the "corner smoothing," techniques have been proposed to achieve nonstop continuous motion. The idea is straightforward. Sharp corner is replaced with a smooth parametric curve, as has already been applied in robotics literature [16]. Jouaneh et al. [17] replaced the corner with a circular arc for fast cornering. However, a circular arc only delivers velocity continuous motion transition. They later used a double (two) clothoid curves to further smoothen the motion transition [18]. These techniques are computationally less stringent and lead to more efficient real-time implementation in the NC system. But, the order of the blending curve must be increased to control the fitting tolerance and the corner geometry at the same time. Yutkowitz and Chester [19] utilized two *4th* order polynomials to realize corner blending for Cartesian motion systems. This method delivers C^2 continuous corner transition in the expense of significant computational load. Pateloup et al. [20] proposed a 2-D path-smoothing method, which employs cubic B-splines with eight control points to round pocket profiles. Similarly, Zhang et al. [21] proposed a more general method utilizing double cubic B-splines. Others [22, 23] also utilized similar transition methods to achieve acceleration continuous feed motion.

One of the vital aspects of corner smoothing is the design of the corner blend so that it can be traveled at high speed and thus the total cycle time could be minimized. In practice, a conventional tool path may contain thousands of linear segments. Minimizing the cornering time spent at each corner actually presents a great potential to reduce overall machining time. However, limited effort has been directed toward the design of optimal corner geometry. Erkorkmaz et al. [24] considered dynamics of the servo system while designing the corner blend. They found that over-corner spline delivers less cornering error but a longer cornering time. Beudaert et al. [25] addressed the problem of sharp corners in 5-axis motion utilizing a pair of B-spline curves. Zhao et al. [23] proposed a real-time cornering scheme where the curvature extremum could be computed for fitted corner splines. They concluded that optimization of the curvature in real-time is computationally stringent and unfeasible. Similarly, authors [26] have tried to find real-time efficient optimization techniques. This paper proposes a "curvature optimal cornering scheme" where the maximum curvature of the corner is minimized in a computationally efficient strategy that is suitable for real-time implementation. This results in a high-speed cornering scheme and thus significant reduction in the total cycle time.

Once the discrete tool path is blended with corner splines, it becomes a mixture of linear segments continuously connected with blending splines. The next problem is scheduling of the feedrate profile along the tool path. Due to curved corner profile, tangential feed must be lowered so that axis velocity and acceleration limits are not violated at corner sections. Therefore, high-speed feed sections of linear segments must be stitched together with low-speed cornering segments while ensuring that the kinematic compatibility conditions between position, velocity, acceleration, and jerk are not violated. Several feed profiles have been suggested in the literature to smoothly schedule the tangential speed [27, 28]. Comprehensive feedrate optimization techniques have also been introduced for complex spline tool paths [29–31] to minimize the cycle time while respecting physical limits of the drives. As compared to the current literature, the work in this paper utilizes smooth *S-curve-type* acceleration transitions while considering the kinematic compatibility conditions between adjacent feed segments. An efficient algorithm is proposed for federate scheduling along the entire tool path considering the maximum cornering speed while respecting the acceleration limits of the drives.

At last, the tool path must be interpolated with respect to the feedrate profile in real-time to generate axis reference commands. In order to smoothly interpolate the reference trajectory without any feedrate fluctuations, the recursive interpolation scheme developed in a previous study [7] for parametric splines is applied on quintic Bezier corner blends.

This paper presents methods to overcome some of the inefficiencies in current corner smoothing techniques and presents a computationally efficient *curvature optimal corner smoothing algorithm* for high-speed machine tools and motion systems to attain minimum cycle time motion. The overall corner smoothing scheme is summarized in Fig. 1. Once the discrete tool path is generated by CAM systems, proposed corner smoothing algorithm utilizes a single quintic Bezier spline to ensure G^2 continuous motion transition between linear segments. The corner curvature is optimized to deliver minimum curvature extremum and thus realizes rapid cornering speed using less acceleration/torque capacities of the axes in motion. Next, S-curve-type acceleration transients are designed to stitch feed segments smoothly along multi-segmented tool path, and a feedrate-scheduling algorithm is

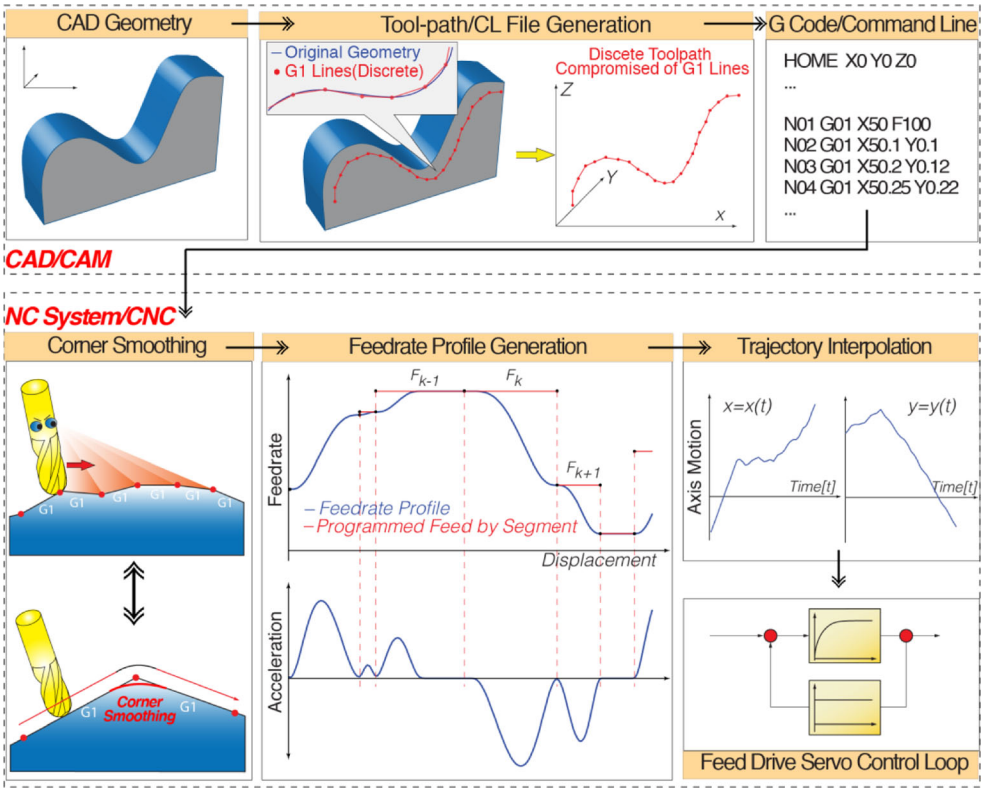

Fig. 1 Proposed corner smoothing scheme

presented to plan the feed motion along multi-segmented tool path. Finally, it is interpolated in real-time and validated experimentally on a 3-axis Cartesian machine tool.

2 Smoothening of sharp corners

Two consecutive linear segments are blended with Bezier splines as shown in Fig. 2a to deliver a smooth transition of feed motion from one linear segment to the next. A quintic Bezier segment (see in Fig. 2b) in Cartesian coordinates can be defined parametrically as follows:

$$\mathbf{B}(u) = \begin{bmatrix} B_x(u) \\ B_y(u) \\ B_z(u) \end{bmatrix} = \left\{ \begin{array}{l} (1-u)^5\mathbf{P}_0 + 5u(1-u)^4\mathbf{P}_1 + 10u^2(u-1)^3\mathbf{P}_2 \\ +10u^3(1-u)^2\mathbf{P}_3 + 5u^4(1-u)\mathbf{P}_4 + u^5\mathbf{P}_5 \end{array} \right\} \text{ and}$$

$$\mathbf{P}_0 = \begin{bmatrix} P_{0,x}(u) \\ P_{0,y}(u) \\ P_{0,z}(u) \end{bmatrix}, \mathbf{P}_1 = \begin{bmatrix} P_{0,x}(u) \\ P_{0,y}(u) \\ P_{0,z}(u) \end{bmatrix}, \ldots, \mathbf{P}_5 = \begin{bmatrix} P_{5,x}(u) \\ P_{5,y}(u) \\ P_{5,z}(u) \end{bmatrix} \quad (1)$$

where B_x, B_y, and B_z are the axis position polynomials. $\mathbf{P}_0$, $\mathbf{P}_1$,…, $\mathbf{P}_5$ are the vectors containing control points for each axis coordinate. u is the Bezier curve parameter, $u \in [0,1]$. As observed from Eq. 1, there are six control points in the proposed Bezier blend that needs to be determined to ensure the following:

(i) Smooth transition between linear segments, namely G^2 continuous motion
(ii) Minimum curvature extremum for the fastest cornering speed
(iii) Accurate control of the cornering error to respect the manufacturing tolerances

2.1 Design of the quintic Bezier blend

The quintic Bezier blend is applied in transition of two point-to-point linear motion segments as shown in Fig. 2c. $\mathbf{P}_{\text{start}}$ is the starting point of the k^{th} linear segment, $\mathbf{P}_{\text{trans}}$ is the ending point, which becomes the transition point to the $(k+1)^{\text{th}}$ segment, and θ is the cornering angle. In blending two consecutive linear segments, G^2 continuity is ensured so that two successive segments share the same tangent and normal directions at the junction points. Combined with accurate interpolation presented in later sections, this allows the realization of C^2 motion. Differentiating Eq. 1 with respect to the spline parameter u gives the *1st* and *2nd* derivative profiles along the Bezier blend:

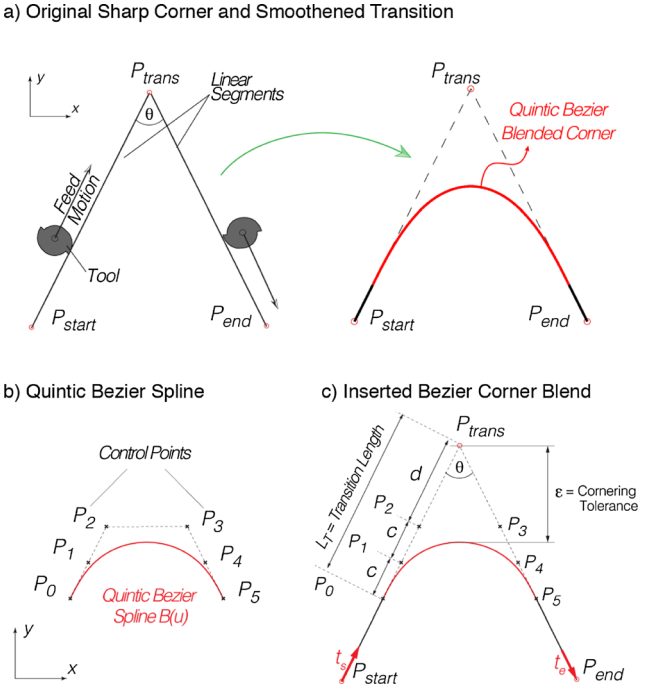

Fig. 2 Quintic Bezier corner blending

$$\left.\begin{aligned}\mathbf{B}_u = \frac{d\mathbf{B}(u)}{du} &= 5(1-u)^4(\mathbf{P}_1-\mathbf{P}_0) + 20u(1-u)^3(\mathbf{P}_2-\mathbf{P}_1) + 30u^2(u-1)^2(\mathbf{P}_3-\mathbf{P}_2) \\ &\quad +20u^3(1-u)(\mathbf{P}_4-\mathbf{P}_3) + 5u^4(\mathbf{P}_5-\mathbf{P}_4) \\ \mathbf{B}_{uu} = \frac{d^2\mathbf{B}(u)}{du^2} &= 20(1-u)^3(\mathbf{P}_2-2\mathbf{P}_1+\mathbf{P}_0) + 60u(1-t)^2(\mathbf{P}_3-2\mathbf{P}_4+\mathbf{P}_5)\end{aligned}\right\} \tag{2}$$

Evaluating the profiles from Eqs. 1 and 2, at the start $u=0$ and end $u=1$ of the Bezier blend reveals

$$\left.\begin{aligned}\mathbf{B}|_{u=0} &= \mathbf{P}_0, \\ \mathbf{B}_u|_{u=0} &= 5(\mathbf{P}_1-\mathbf{P}_0) \\ \mathbf{B}_{uu}|_{u=0} &= 20(\mathbf{P}_2-2\mathbf{P}_1+\mathbf{P}_0)\end{aligned}\right\} \text{ and } \left.\begin{aligned}\mathbf{B}|_{u=1} &= \mathbf{P}_5, \\ \mathbf{B}_u|_{uu=1} &= 5(\mathbf{P}_5-\mathbf{P}_4) \\ \mathbf{B}_{uu}|_{u=1} &= 20(\mathbf{P}_5-2\mathbf{P}_4+\mathbf{P}_3)\end{aligned}\right\} \tag{3}$$

As observed from Eq. 3, $\boldsymbol{P_0}$ and $\boldsymbol{P_5}$ define junction points of the Bezier blend to linear segments. In this design, they are placed at a Euclidian distance, $2c+d$ away from the corner transition point, $\mathbf{P}_{\text{trans}}$ as

$$\mathbf{P}_0 = \mathbf{P}_{trans}-(2c+d)t_s \quad and \quad \mathbf{P}_5 = \mathbf{P}_{trans} + (2c+d)\mathbf{t}_e\} \tag{4}$$

where $\mathbf{t}_s$ and $\mathbf{t}_e$ are the unit tangent vectors along linear segments,

$$\mathbf{t}_s = \frac{\mathbf{P}_{trans}-\mathbf{P}_{start}}{\|\mathbf{P}_{trans}-\mathbf{P}_{start}\|} \quad and \quad \mathbf{t}_e = \frac{\mathbf{P}_{end}-\mathbf{P}_{trans}}{\|\mathbf{P}_{end}-\mathbf{P}_{trans}\|} \tag{5}$$

Tangent vectors on the junction points of the Bezier can be aligned to linear segment by simply placing $\mathbf{P}_1$ and $\mathbf{P}_4$ anywhere along $\mathbf{t}_s$ and $\mathbf{t}_e$. They are placed at a distance of c from the junction points as

$$\mathbf{P}_1 = \mathbf{P}_0 + c\mathbf{t}_s \, and \, \mathbf{P}_4 = \mathbf{P}_5-c\mathbf{t}_e\} \tag{6}$$

Please note that this only ensures G^1 continuity since only the tangent directions are matched but not their magnitude. Next, in order to satisfy curvature continuity, normal vectors at the junction points, $\mathbf{B}_{uu}|_{u=0}$ and $\mathbf{B}_{uu}|_{u=1}$, must be zero. According to Eq. 3, this can be achieved by satisfying

$$\left.\begin{aligned}\mathbf{B}_{uu}|_{u=0} &= 20(\mathbf{P}_2-2\mathbf{P}_1+\mathbf{P}_0) = 0 \quad \rightarrow \quad \mathbf{P}_1-\mathbf{P}_0 = \mathbf{P}_2-\mathbf{P}_1 \\ \mathbf{B}_{uu}|_{u=1} &= 20(\mathbf{P}_5-2\mathbf{P}_4+\mathbf{P}_3) = 0 \quad \rightarrow \quad \mathbf{P}_4-\mathbf{P}_5 = \mathbf{P}_3-\mathbf{P}_4\end{aligned}\right\} \tag{7}$$

Substituting Eq. 7 into Eq. 4 yields

$$\mathbf{P}_0 + 2c\mathbf{t}_s = \mathbf{P}_2, \quad and \quad \mathbf{P}_5-2c\mathbf{t}_e = \mathbf{P}_3\} \tag{8}$$

which reveals that the control points $\mathbf{P}_1$ must be placed collinear with $\mathbf{P}_0$ and $\mathbf{P}_2$, and $\mathbf{P}_3$ with $\mathbf{P}_4$ and $\mathbf{P}_5$. From Eqs. 6 and 8, all six control points of the Bezier blend are determined as

$$\left.\begin{aligned}\mathbf{P}_0 &= \mathbf{P}_{trans}-(2c+d)\mathbf{t}_s \quad \mathbf{P}_5 = \mathbf{P}_{trans} + (2c+d)\mathbf{t}_e \\ \mathbf{P}_1 &= \mathbf{P}_0 + c\,\mathbf{t}_s \quad \mathbf{P}_4 = \mathbf{P}_5 - c\,\mathbf{t}_s \\ \mathbf{P}_2 &= \mathbf{P}_0 + 2\,c\,\mathbf{t}_s \quad \mathbf{P}_3 = \mathbf{P}_5 - 2\,c\,\mathbf{t}_s\end{aligned}\right\} \tag{9}$$

making the corner blend symmetric with respect to the angular bisector of the two lines $\overline{\mathbf{P}_{start}\mathbf{P}_{trans}}$ and $\overline{\mathbf{P}_{trans}\mathbf{P}_{end}}$, and G^2 continuous to the neighboring segments.

2.2 Curvature optimization

Curvature is solely a function of the path geometry, and it is integrally tied to the acceleration and jerk required to track the corner geometry. In high-speed machining, it is desirable to maximize the total cornering speed while still respecting the actuator acceleration. The curvature along the path is defined as follows [32]:

$$\kappa = \left\|\frac{d\mathbf{T}}{ds}\right\|, \quad \mathbf{T} = \left[t_x, t_y, t_z\right]^T_{3\times 1} \tag{10}$$

where $\mathbf{T}$ is the unit tangent vector and s is the arc displacement. The relationship between the differential arc displacement ds and the Bezier parameter increment du can be derived as

$$ds = \|\mathbf{B}_u\|du \tag{11}$$

Furthermore, as the curve is traveled along, change in the unit tangent vector as a function of time can be expressed by applying the chain rule to Eq. 10:

$$\left\|\frac{d\mathbf{T}}{dt}\right\| = \left\|\frac{d\mathbf{T}}{ds}\frac{ds}{dt}\right\| = \kappa v \tag{12}$$

where $v = s = ds/dt$ is the tangential or the path velocity. Please note that the derivative of the unit tangent vector, $d\mathbf{T}/dt$, is orthogonal to itself. Therefore, the path normal can also be written as a function of time:

$$\mathbf{N} = \frac{1}{\|d\mathbf{T}/dt\|}\frac{d\mathbf{T}}{dt} \tag{13}$$

Axis velocity $\mathbf{v}=[v_x, v_y, v_z]^T$ and acceleration $\mathbf{a}=[a_x, a_y a_z]^T$ components are written from Eqs. 12 and 13 as

$$\left.\begin{aligned} &\mathbf{v} = v\mathbf{T}\ and \\ &\mathbf{a} = \frac{d}{dt}(v\mathbf{T}) = \frac{dv}{dt}\mathbf{T} + v\frac{d\mathbf{T}}{dt} \end{aligned}\right\} \tag{14}$$

and from Eq. 14 the effect of curvature on the axis accelerations is derived as

$$\mathbf{a} = \dot{v}\,\mathbf{T} + v^2\kappa\mathbf{N} \tag{15}$$

The $\dot{v} = \ddot{s} = dv/dt$ term represents tangential or the so-called path acceleration. Assuming that the corners are turned at constant speed, that term vanishes relating the axis accelerations directly to the path velocity, v and the curvature as

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = v^2\kappa\mathbf{N} \tag{16}$$

Thus, the objective hereby is to minimize curvature extremum, i.e., the maximum of the curvature along the Bezier blend, so that the cornering speed can be increased while still staying within the acceleration limits of drives.

In the proposed Bezier corner blend design, the shape of the Bezier, i.e., the control point locations, is controlled by the Euclidian distances c and d. Figure 3 shows the effect of c and d on the Bezier shape for a cornering angle of 60° when junction points ($\mathbf{P}_0$ and $\mathbf{P}_5$) are fixed. As illustrated, by keeping the junction points fixed and increasing d, control points are pushed away from corner transition point $\boldsymbol{P}_{\boldsymbol{trans}}$ leading to larger cornering errors. Increasing c leads to smaller fitting errors but in contrary may generate larger curvature radius depending on d. Thus, the optimal ratio of c/d is searched for minimizing the curvature extremum. Curvature along the Bezier can be written analytically from Eqs. 2, 10, and 11 as a function of the curve parameter, u:

$$\kappa = \frac{-12nu\sin\theta(2n+1)(u-1)(nu-2u-2nu^3+nu^4+4u^3-2u^4+1)}{L_T A^{\frac{3}{2}}} \tag{17}$$

where $n=c/d$, $L_T=2c+d$, and $A=A(u)$ is an 8th order polynomial. Please refer to the Appendix section for detailed derivation of Eq. 17. As noted, curvature is affected by the cornering angle, θ, n, and also the total transition length L_T, which simply scales the curvature in this design. The optimal n value for various cornering angles can be searched to minimize the curvature extremum from Eq. 17. Nevertheless, this operation may be computationally expensive considering the fact that it needs to be implemented in real-time for each and every corner. This bottleneck has also been stated previously in the design of corner blends [22, 23]. Instead of solving the optimization problem in real-time, a more practical approach is pursued here.

Since the cornering distance L_T simply functions as a scaling factor in the proposed design, it is set to unity

$$L_T = 2c + d = 1 \tag{18}$$

and only the effect of $n=c/d$ considered. The corner angle θ is fixed, and n is interpolated in the range of $n\in[0,2]$. Curvature is computed along the Bezier, at 100 discrete points proportional to the spline parameter, and its maxima is detected.

Figure 4 shows the change of maximum curvature, $\kappa_{\max}$. As shown, since Bezier blend is symmetric with respect to the angular bisector of the neighboring segments, maximum curvature occurs either in the middle of the Bezier, or two identical curvature extrema are observed. This vaguely indicates that the problem is convex, and there is a global optimal n value, which minimizes the maximum value of the curvature. The process is repeated for various cornering angles to record the optimal n. Figure 5 shows the n value for a wide range of cornering angles $\theta=10\ldots150°$, which minimizes the maximum curvature. A simple power regression is applied to obtain the following relation to estimate the curvature optimal n value for a wide range of cornering angles as

$$n(\theta) = \frac{1}{2.0769}\theta^{0.9927} \tag{19}$$

Equation 19 can be used in real-time to efficiently lookup the optimal control point ratio, n, to fit the curvature optimal Bezier blend.

2.3 Control of the cornering errors

As shown in Fig. 2c, proposed corner smoothing algorithm introduces geometric errors, ε, around the corner, which has to be limited by a given tolerance so that the dimensional integrity of the produced part is ensured. The Bezier blend is symmetric, and hence the maximum deviation from original path occurs in the middle at $u=0.5$,

$$\varepsilon = \left\|\mathbf{P}_{trans} - \mathbf{B}\big|_{u=0.5}\right\| \tag{20}$$

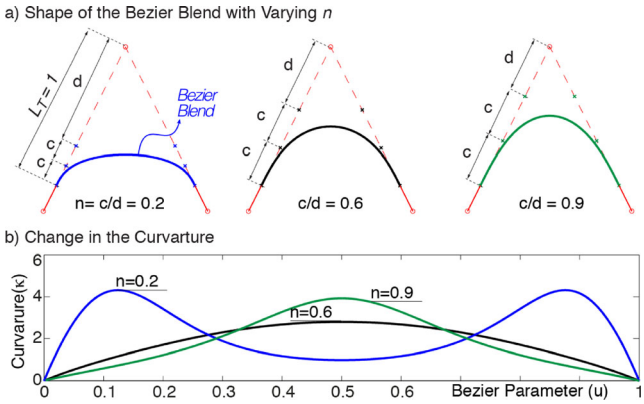

Fig. 3 Effect of n on the Bezier curvature (θ=60°)

The midpoint of the Bezier $\mathbf{B}|_{u=0.5}$ is evaluated from Eqs. 1 and 9 as

$$\mathbf{B}|_{u=0.5} = \mathbf{P}_{trans} + \frac{7c+16d}{32}(\mathbf{t}_s + \mathbf{t}_e) \tag{21}$$

The maximum cornering error is then computed from Eqs. 20 and 21 as

$$\varepsilon = \frac{7n+16}{32} d \|\mathbf{t}_s + \mathbf{t}_e\| \tag{22}$$

Equation 22 is further expanded, and actual values of c and d for a curvature optimum Bezier blend at a given cornering tolerance ε and corner angle θ are obtained as

$$d = \frac{32\varepsilon}{(7n+16)\sqrt{2+2\cos(\theta)}}$$
$$c = \frac{32\varepsilon}{(7n+16)\sqrt{2+2\cos(\theta)}} n \tag{23}$$

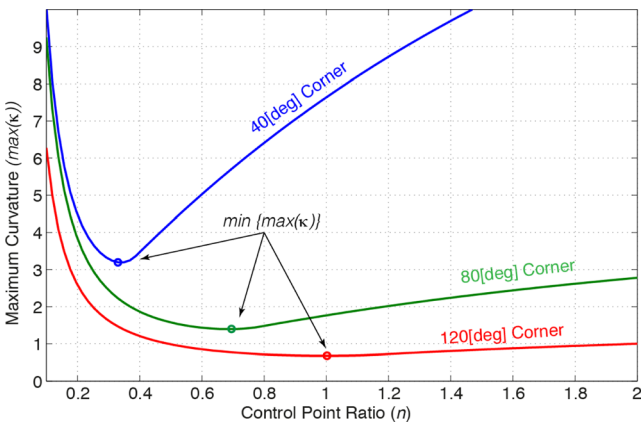

Fig. 4 Corner curvature for various cornering angles and control point ratios

The actual transition length can then be evaluated as $L_T = 2c+d$.

3 Jerk-limited trajectory generation

Based on the developed corner smoothing algorithm in Section 2, a jerk-limited trajectory generation algorithm is proposed in this section. The objective is to modulate the path speed in such a way that the velocity is reduced at corners i.e. along Bezier segments and again raised at linear segments to reach the programmed speed. The proposed trajectory generation algorithm can be implemented in real-time within a look-ahead scheme, or it can be used off-line for simulating the motion of computer numerical-controlled (CNC) systems. The overall algorithm has three sections. At first, corners are rounded with Bezier blends, and the maximum feedrate for each corner is determined. Next, the jerk-limited feedrate profile with S-curve-type acceleration transients is planned, and it is optimized so that the total cycle time is minimized while respecting the axis velocity and acceleration limits. At last, the geometric tool path is interpolated in real-time without any feed fluctuation.

3.1 Determination of the maximum cornering speed

Conventional tool paths contain series of linear segments where each segment has its Euclidian length and a feedrate assigned to it. Once the proposed corner smoothing approach is implemented, corners are blended and segment lengths are altered. Figure 6 illustrates a planar tool path consisting of linear segments. The total length S_L of a tool path with N_s linear segments is computed considering length of each block:

$$S_L = \sum_{1}^{N_s} \|P_{k+1} - P_k\| \tag{24}$$

Due to corner blending, tool-path length is updated by subtracting corner transition length, L_T, and adding the length of the Bezier blend, L_B, as

$$S_\Sigma = S_L + \sum_{1}^{N_s-1} \left(L_{B,k} - 2L_{T,k}\right) \tag{25}$$

where L_B is computed through numerical integration as follows. The Bezier blend is divided into M (>100) subdivisions proportional to its parameter u, and the corresponding axis positions are computed as

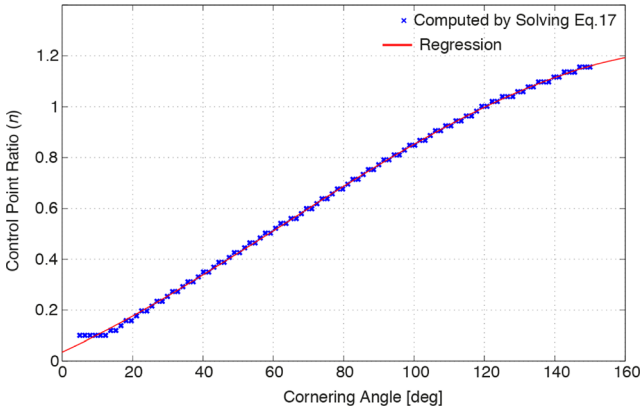

Fig. 5 Optimal n w.r.t. cornering angle

$$\left.\begin{aligned} x_j &= (1-j\Delta u)^5\mathbf{P}_{0,x} + 5j\Delta u(1-j\Delta u)^4\mathbf{P}_{1,x} + \ldots + j\Delta u^5\mathbf{P}_{5,x} \\ y_j &= (1-j\Delta u)^5\mathbf{P}_{0,y} + 5j\Delta u(1-j\Delta u)^4\mathbf{P}_{1,y} + \ldots + j\Delta u^5\mathbf{P}_{5,x} \\ z_j &= (1-j\Delta u)^5\mathbf{P}_{0,z} + 5j\Delta u(1-j\Delta u)^4\mathbf{P}_{1,z} + \ldots + j\Delta u^5\mathbf{P}_{5,z} \end{aligned}\right\},$$

$$\Delta u = \frac{1}{M}, j = 0, 1\ldots M \tag{26}$$

The resulting arc increments are then summed up:

$$\begin{aligned} L_B &= \sum_{j=1}^{M} \Delta s_{k,j} \\ &= \sum_{j=1}^{M} \sqrt{\left(x_{k,j}-x_{k,j-1}\right)^2 + \left(y_{k,j}-y_{k,j-1}\right)^2 + \left(z_{k,j}-z_{k,j-1}\right)^2} \end{aligned} \tag{27}$$

Next, due to curvature of the Bezier blend, axis acceleration limits may be violated if the curved sections were also to be traveled at the same speed of neighboring linear segments. The highest cruise speed along a Bezier blend is bounded by the acceleration limits $A_{x,\max}$, $A_{y,\max}$, *and* $A_{z,\max}$ of the drives. Equation 16 is used to determine the highest cornering speed as

$$F_{corner} = \min\left\{\frac{A_{x,\max}}{\kappa_{\max}}, \frac{A_{y,\max}}{\kappa_{\max}}, \frac{A_{z,\max}}{\kappa_{\max}}\right\} \tag{28}$$

A case study is shown in Fig. 6. The desired feedrate for the linear tool path is given as F=35 mm/s. The tool path is smoothened at two different cornering tolerances namely, 100 and 150 μm. As shown, depending on the corner tolerance, the transition length L_T changes. For tighter cornering tolerance, L_T becomes smaller and thus the maximum curvature is larger. Figure 6b shows the highest cornering speed for different cornering tolerances. It can be noted that the maximum cornering speed reduces as the cornering tolerance becomes tighter.

3.2 Feedrate scheduling

As presented in the previous section, when linear segments are joined with Bezier blends, corners can only be traveled at a much lower speed then their neighboring segments. The motion system has to slow down when approaching to a corner, cruise the Bezier blend and accelerate to the programmed feed of the successive linear segment. In order to achieve smooth motion, feed commands of consecutive segments must be stitched together smoothly. Speed transitions across blocks are achieved through a jerk-limited acceleration profile. Compatibility of the feedrates with respect to acceleration and jerk constraints given by the process planner are considered, and a practical feed planning scheme is developed as follows.

A jerk-limited motion profile or the so-called S-curve-type acceleration profile is utilized to generate smooth speed transition between the two adjacent feed blocks. The feedrate profile for velocity transition is shown in Fig. 7, and the corresponding acceleration function is defined as

$$a(t) = 30\Delta V\frac{t^4}{t_e^5} - 60\Delta V\frac{t^3}{t_e^4} + 30\Delta V\frac{t^2}{t_e^3} \tag{29}$$

where $\Delta V = F_{k+1} - F_k$ is the velocity difference to be covered from current k^{th} segment to the consecutive $(k+1)^{th}$ and t_e is total time spend in speed transition. Position (s), velocity (v), acceleration (a), and jerk (j) profiles can be obtained from Eq. 29 as

$$\left.\begin{aligned} s(t) &= F_k t + \frac{5}{2}\Delta V\frac{t^4}{t_e^3} - 3\Delta V\frac{t^5}{t_e^4} + \Delta V\frac{t^6}{t_e^5} \\ v(t) &= F_k - 10\Delta V\frac{t^3}{t_e^3} + 15\Delta V\frac{t^4}{t_e^4} + 6\Delta V\frac{t^5}{t_e^5} \\ a(t) &= 30\Delta V\frac{t^2}{t_e^3} - 60\Delta V\frac{t^3}{t_e^4} - 30\Delta V\frac{t^4}{t_e^5} \\ j(t) &= 60\Delta V\frac{t}{t_e^3} - 180\Delta V\frac{t^2}{t_e^4} + 120\Delta V\frac{t^3}{t_e^5} \end{aligned}\right\} \tag{30}$$

The total transition time, t_e, needs to be determined in such a way that the tangential acceleration and jerk limits, $A_{\max}$ and $J_{\max}$, are not exceeded. As observed from Fig. 7,

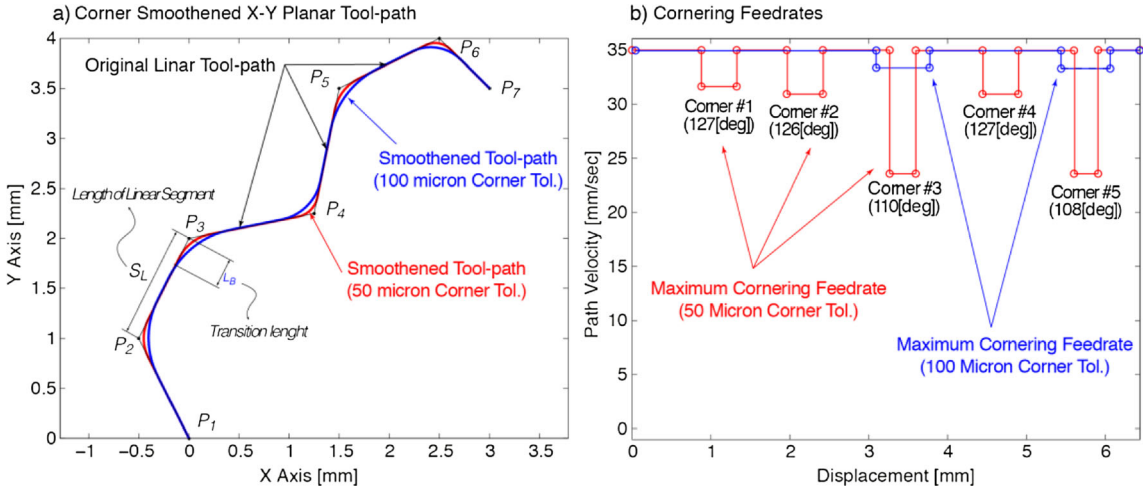

Fig. 6 Corner smoothened tool path

tangential acceleration peaks at $t=½t_e$. Thus, the speed transition interval $t_e^{A\max}$ with respect to acceleration limit can be computed from Eq. 30 as

$$A_{\max} = \frac{15}{8}\frac{|\Delta V|}{t_e} \rightarrow t_e^{A\max} = \frac{15}{8}\frac{|\Delta V|}{A_{\max}} \tag{31}$$

Similarly, tangential jerk prolife peaks at $t=¼t_e$, and the speed transition interval w.r.t. jerk limit is calculated as

$$J_{\max} = \frac{45}{8}\frac{|\Delta V|}{t_e^2} \rightarrow t_e^{J\max} = \sqrt{\frac{45}{8}\frac{|\Delta V|}{J_{\max}}} \tag{32}$$

In order to respect both limits, the transition interval is determined as

$$t_e = \max\left(t_e^{A\max}, t_e^{J\max}\right) \tag{33}$$

Finally, the total displacement spent during a speed transition is computed from Eqs. 30 and 33 as

$$\Delta S = \frac{F_{k+1} + F_k}{2} t_e \tag{34}$$

Feed transitions are dictated by desired speeds of the neighboring segments and the total displacement traveled within the speed transition. The "kinematic compatibility" is defined here as having sufficient travel length to smoothly change the feedrate, i.e., accelerate or decelerate, between segments [31]. If kinematic compatibility is achieved, then displacement, feedrate, and acceleration profiles will be continuous, and along with the jerk profile, they will be limited. Figure 8 illustrates the importance of the kinematic compatibility. In Fig. 8a, speed increase from low- to high-speed segment is shown where $F_k < F_{k+1}$. Please note that speed cannot be increased from within the current, k^{th} segment, since the current segment feed is lower. In other words, acceleration can only be planned after slow feed segment is traveled. Thus, speed transition is planned in the successive $(k+1)^{th}$ segment, and the compatibility condition is checked as

$$L_{k+1} - \frac{F_k + F_{k+1}}{2} t_{e,k} \geq 0,$$

$$\text{where } t_{e,k} = \max\left(\frac{15}{8}\frac{|\Delta V_k|}{A_{\max}}, \sqrt{\frac{45}{8}\frac{|\Delta V_k|}{J_{\max}}}\right) \tag{35}$$

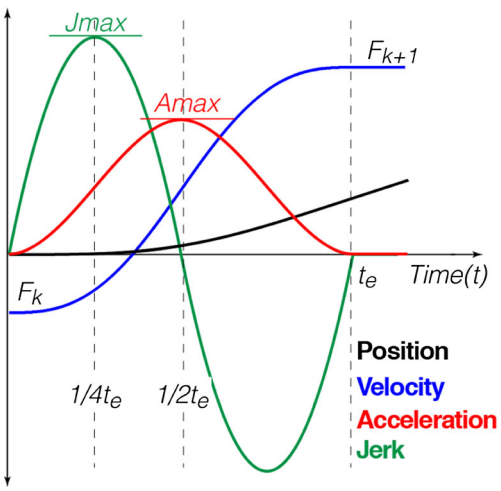

Fig. 7 S-curve-type acceleration profile for velocity transition

and $\Delta V_k = F_{k+1} - F_k$ and L_{k+1} is the length of the $(k+1)^{th}$ segment. Figure 8b shows a velocity transition from high to low speed. In this case, all the planning is performed within the current k^{th} segment, and the compatibility conditions are checked as

$$L_k - \frac{F_k + F_{k+1}}{2} t_{e,k} \geq 0 \tag{36}$$

In general cases, feed transitions spread across segment boundaries, and two adjacent segments may affect kinematic compatibility in the k^{th} segment. In order to ascertain whether kinematic compatibility is satisfied across the segment boundaries, previous and next segments also need to be tested against the feed compatibility conditions. Figure 8c simply illustrates this where a high-speed block is neighbored by two low-speed segments, which could be either linear or Bezier segment. In this case, all the acceleration transition must be planned within the k^{th} segment, and the compatibility conditions can be written as

$$L_k - \left(\frac{F_{k-1} + F_k}{2} t_{e,k-1} + \frac{F_k + F_{k+1}}{2} t_{e,k} \right) \geq 0 \tag{37}$$

If Eq. 37 cannot be satisfied, achievable speed of the segment k must be lowered so that the speed transition from and to the segments $k-1$ and $k+1$ is achieved. In simple cases, maximum speed along the Bezier blends can be computed in a forward traversal search algorithm, and the compatibility conditions from the start to the end of the tool path are checked. If a violation occurs, feedrate of the current block is reduced until it satisfies the compatibility conditions with its neighboring blocks. However, in a large-scale tool path compromised of many blocks, a compatible solution may not be acquired given the commanded feed, acceleration, and jerk values. In such circumstances, violations can only be corrected by tracing back the feedrate through the planned feed values and perform adjustment to earlier segments. A practical search algorithm is implemented here to efficiently generate a kinematically compatible feed profile. The proposed algorithm can be used offline for the entire tool path all at once, or it can be implemented in a windowing fashion for real-time execution. The feed planning algorithm is presented as follows:

Step 1 All the feeds in a scheduling window are initiated to a conservative feed value, $F_k = v_{start}$ where $k = 1 \ldots N$.

Step 2 Compatibility conditions are checked for each block counting in forward and backward directions simultaneously. If the there is no kinematic violation in any block, all the block feeds are updated gradually by the amount of dv, $F_k = F_k + dv$ and the velocity increment for the next round is doubled, $dv = 2dv$. However, if there is any kinematic violation for a block, the velocity increment is reduced by half, $dv = dv/2$ and the feedrates are updated.

Step 3 If the kinematic violation persists in a segment, its feedrate is frozen, and the rest of the blocks are updated with the current feed increment, dv. If the violation still persists at a certain block, in this case, the neighboring block feedrates are also frozen, and the remaining feedrates are updated.

Step 4 The feed update continues, and the final feed profile is generated when all the block feedrates are frozen and cannot be updated.

The proposed feedrate scheduling technique is applied on the tool path shown in Fig. 6a. The cornering tolerance is set to 25 μm and desired feedrate is set to $F = 35$ mm/s. Tangential acceleration and jerk limits are set to 1,500 mm/s^2 and 20×10^4 mm/s^3, respectively. Acceleration limits of the drives are also set to $A_{x,max} = A_{y,max} = 1{,}500$ mm/s^2, and the resultant feed profile is shown in Fig. 9a. As observed, segments are connected to each other with smooth acceleration transients, and corner segments are traveled at lower speeds considering acceleration limits of the drives. Furthermore, due to low acceleration and jerk limits, desired tangential feedrate could not be reached along the tool path. In Fig. 9b, the limits are increased to $A_{max} = A_{x,max} = A_{y,max} = 2{,}000$ mm/s^2 and $J_{max} = 40 \times 10^4$ mm/s^3. In this case, desired feed could be reached.

3.3 Real-time interpolation

After smoothing the given discrete path with Bezier blends, the new tool path is composed of linear and Bezier segments, and care must be taken while interpolating the mixed trajectory. Linear segments can simply be interpolated with linear interpolation:

$$\begin{bmatrix} x(kT_s) \\ y(kT_s) \\ z(kT_s) \end{bmatrix} = \mathbf{P}_k + \frac{\mathbf{P}_{k+1} - \mathbf{P}_k}{\|\mathbf{P}_{k+1} - \mathbf{P}_k\|} s(kT_s), \quad k = 0 \ldots N \tag{38}$$

where $s(kT_s)$ is the displacement profile sampled at sampling time of the interpolator, T_s, and N is the number of interpolation samples. However, since Bezier curves are not arc length parameterized [31], feedrate fluctuations will occur if it is

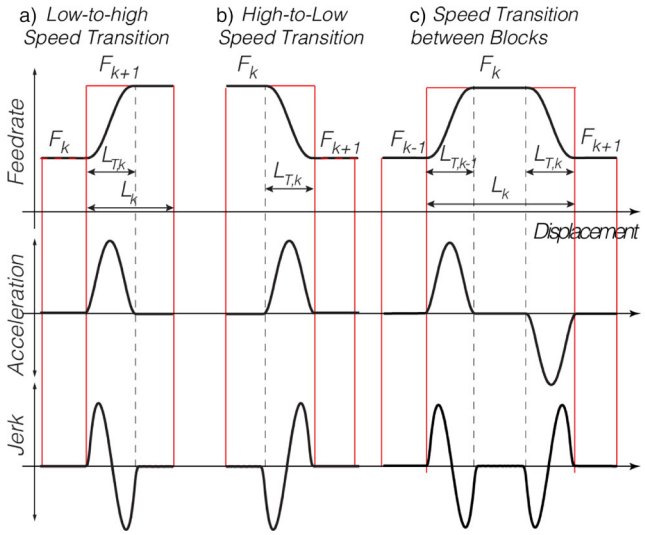

Fig. 8 Compatibility conditions along successive segments

interpolated by applying constant parameter increments Δu, which are proportional to the desired arc increment Δs in the form

$$\Delta u = \frac{1}{L_B}\Delta s \tag{39}$$

Furthermore, only G^2 continuity is pursued while fitting the Bezier curve to the sharp corners. As a result, while traveling at constant speed, severe feed fluctuations may occur at the transition points from linear to Bezier segments since the magnitude of tangent vector of the Bezier at the junction points is not matched. Such discontinuity will result in high frequency acceleration and jerk harmonics and will cause unwanted vibrations exciting structural modes of the drive system. Taylor approximation has been used in general practice to interpolate parametric curves, which may fail to work when crossing from the linear segment to the Bezier segments at high speed [31]. In order to avoid such problems, the iterative spline parameter computation technique from Erkorkmaz and Altintas [7] is applied to accurately interpolate the Bezier blends. The idea is to solve the value of Bezier parameter Δu to realize the desired arch displacement Δs. The arc increment Δs_{test} corresponding to the tested spline parameter u_{test} is estimated by computing the resulting axis increments:

$$\left.\begin{aligned}
&\Delta s_{test} = \sqrt{(\Delta x_{test})^2 + (\Delta y_{test})^2 + (\Delta y_{test})^2}\\
&where:\\
&\Delta x_{test} = x_{test} - x_{prev}\ ,\quad \Delta y_{test} = y_{test} - y_{prev},\quad \Delta z_{test} = z_{test} - z_{prev}\\
&x_{test} = (1-u)^5 P_{0,x} + 5u(1-u)^4 P_{1,x} + \ldots + u^5 P_{5,x}\\
&y_{test} = (1-u)^5 P_{0,y} + 5u(1-u)^4 P_{1,y} + \ldots + u^5 P_{5,x}\\
&z_{test} = (1-u)^5 P_{0,z} + 5u(1-u)^4 P_{1,z} + \ldots + u^5 P_{5,z}
\end{aligned}\right\} \tag{40}$$

where x_{prev}, y_{prev}, and z_{prev} are the axis commands applied in the previous control sample. The error between the desired and tested arc increments is

$$e = \Delta s_{des} - \Delta s_{test} \tag{41}$$

and its gradient with respect to the tested spline parameter is derived using Eqs. 39 and 41 as

$$\frac{de}{du_{test}} = \frac{\Delta x_{test}\left(\frac{\Delta x_{test}}{\Delta u}\right) + \Delta y_{test}\left(\frac{\Delta y_{test}}{\Delta u}\right) + \Delta z_{test}\left(\frac{\Delta z_{test}}{\Delta u}\right)}{\Delta s_{test}} \tag{42}$$

where (dx_{test}/du), (dy_{test}/du), and (dz_{test}/du) are simply obtained by substituting the calculated value of u_{test} into the derivative expressions given in Eq. 2. The correct value of the spline parameter is computed with regard to the convergence of the below iteration (i):

$$(u_{test})_{i+1} = (u_{test})_i - \frac{e_i}{(de/du_{test})_i} \tag{43}$$

The desired convergence is achieved when the arc increment error e is below 10^{-8} in Eq. 43. Since the Bezier blend is smooth and does not show sharp changes in the curvature, the initial guess for u_{test} can be obtained through linear approximation of the nonlinear relationship between the arc length s and spline parameter u from Eq. 38. Depending on the availability of a close initial guess and an analytical gradient enables Eq. 43 to converge reliably, within five iterations, at each trajectory interpolation cycle.

4 Simulation and experimental results

The proposed corner smoothing technique is tested through simulations and experimentally validated. At first, a simulation study is performed to illustrate the effect of curvature optimized corner smoothing. Experimental tracking experiments are then conducted on a conventional CNC machine tool controlled by in-house developed NC system.

4.1 Simulation results

A 2-D rectangular tool path shown in Fig. 10 is smoothened with quintic Bezier blends. The cornering tolerance is selected as 150 μm, and the corner points are indicated as $\mathbf{P}_k$, k=0,

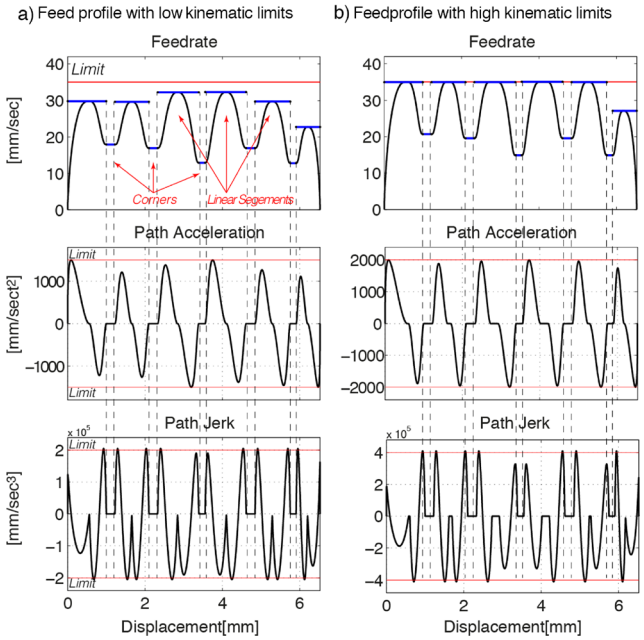

Fig. 9 Scheduled feedrate along corner smoothened tool path

1,…, 4. All the corner angles are 90° for this tool path, and they are smoothed with both using curvature optimal and nonoptimal Bezier blends. In the proposed curvature optimal Bezier blending method, the optimal control point ratio, n=1.3265, is looked up efficiently from Eq. 19. In the remaining two cases, n is selected greater and smaller than the optimal value. Figure 10b shows the change in the curvature. As noted, proposed blending technique ensures G^2 continuity where curvature at the junction points is zero for any n. In addition, when the optimal corner ratio is utilized, it delivers the smallest curvature extremum. The feedrate profile along the tool path is then generated as presented in Section 3, and presented in Fig. 11. As observed, when the curvature is not optimized along the Bezier, the cornering feedrates are conservative. Total cycle time for the proposed curvature optimized method is the smallest T_Σ=0.3655 s as compared to the nonoptimal cases where the cycle times are T_Σ=0.3874 s at n=0.6878 and T_Σ=0.3691 s at n=1.9652. This proves that the proposed Bezier corner smoothing method can deliver curvature continuous corner smoothing with minimum curvature extremum for the fastest cornering.

4.2 Experimental results

A commercial CNC machine tool is retrofitted to implement the proposed corner smoothing algorithms shown in Fig. 12. The conventional NC system is cancelled by switching the amplifiers of the machine into torque mode and modifying the PLC. Encoder feedback is taken from the motor side rotary encoders, and torque commands are directly send to the amplifiers. dSpace® real-time control system is utilized and P-PI cascade controllers are implemented for each axis for accurate position control. The trajectory is generated off-line in MATLAB® environment, and it is interpolated in real-time at the closed loop control interval of 10 kHz.

A realistic case study is presented to validate the effectiveness of the proposed sharp corner smoothing algorithm on a complex planar tool path shown in Fig. 13. The tool path consists of 126 linear segment lengths varying from 3 mm to 13 mm. The desired travel speed for each line is F=100 mm/s, maximum path acceleration is $A_{\max}$=2,500 mm/s^2 and jerk is set to be maximum $J_{\max}$=20×10^4 mm/s^3. The x- and y-axis acceleration limits are selected identical to the tangential acceleration $A_{\max}$=$A_{y,\max}$=$A_{\max}$=2,500 mm/s^2.

The tool path is traveled at three different trajectory generation strategies. Firstly, if the proposed corner smoothing technique is not applied, the motion must be commanded to perform an instantaneous stop at the end of each linear segment since the linear segments only ensure position continuity. In this case, the jerk-limited acceleration profile is utilized to plan the feed motion along each linear segment individually, and the point-to-point trajectory is generated. The resultant tangential feed profile and axis kinematics are shown in Fig. 14. As shown, servo system undergoes instantaneous stops at the junction points (see Fig. 14a). Please note that the path speed only reaches the commanded feed at longer linear segments due to the limited acceleration of the drives. Since linear tool path has zero curvature, axis accelerations and velocities never exceed given bounds as shown in Fig. 14b. The total cycle time is 13.39 s.

Next, the proposed corner smoothing technique is implemented. The cornering tolerance is set to ε=100 μm, and the tool path shown is smoothened. Since the tool path is blended with the curvature continuous Bezier blends, it can be traveled at a constant speed without stopping. In other words, a "nonstop" constant feed motion can be achieved. The reference tangential feed profile is shown in Fig. 15. The dashed line presents the "constant feed profile." As observed, the path speed is constant along the entire tool path, and total cycle time can be reduced down to 5.8 s. This cycle time is less than half of the conventional case. However, when the tool path is traveled at constant feedrate, axis acceleration limits are exceeded (see dashed lines in Fig. 15d, e). This is simply due to curvature of Bezier blends and requires the feedrate to be lowered in order to avoid axis acceleration or actuator saturation.

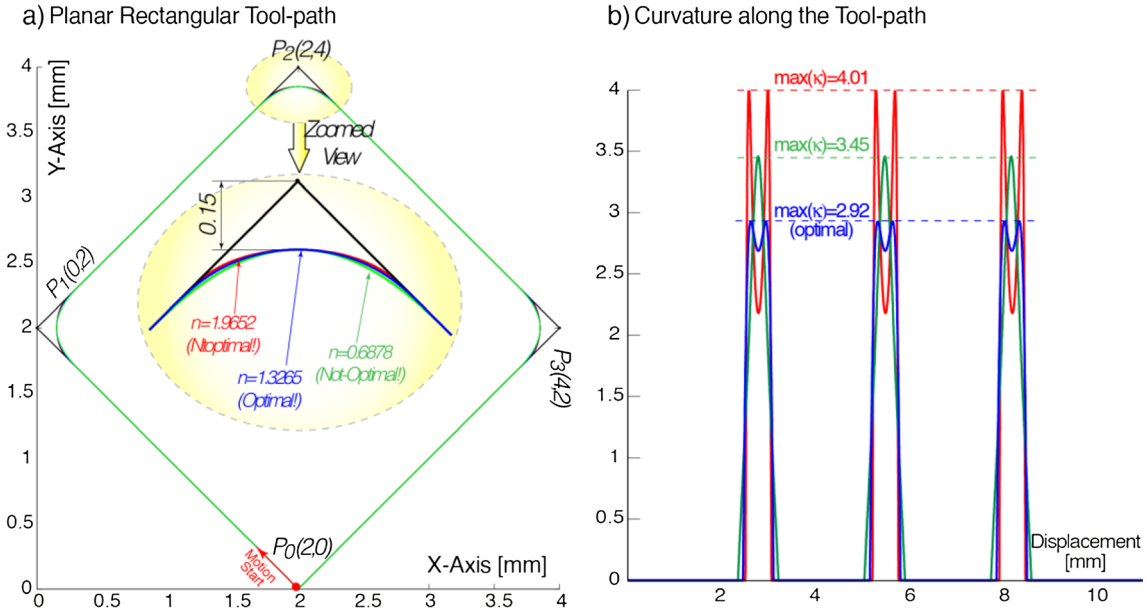

Fig. 10 Rectangular tool path

At last, feedrate along the smoothed tool path is scheduled where the tangential velocity is reduced down to respect the acceleration limits of the drives, $A_{x,max}$ and $A_{y,max}$. The "scheduled feed profile" is presented in Fig. 15a by the solid line. As shown, the feed is lowered before entering Bezier blend sections so that the axis acceleration limits are respected. Particularly, the feedrate is lowest at the four sharp corners, i.e., legs of the Starfish. It is increased along the linear sections to minimize the cycle time. Solid lines in Fig. 15d, e show that the axis acceleration limits are respected in the proposed method. The total cycle time is computed to be 6.64 s. This cycle time is slightly higher than the "constant feed" case since the feedrate is lowered at Bezier sections. Nevertheless, it is almost half of the point-to-point case. This proves that the developed system effectively schedules the feedrate to achieve highest productivity in high-performance motion stages while utilizing physical limits of the machines.

The tracking responses of x- and y-axes are compared in Fig. 16. Largest tracking errors occur when high acceleration is demanded from the drives. This is mostly due to the fact that motion controllers cannot fully cancel the drive inertia in the feed-forward control loop. Since axis accelerations are bounded in the proposed scheme, it delivers tracking errors similar to the point-to-point motion. However, when the tool path is traveled at constant speed, the acceleration limits are exceeded at the corner sections and the tracking performance is degraded severely as shown in Fig. 16b, c.

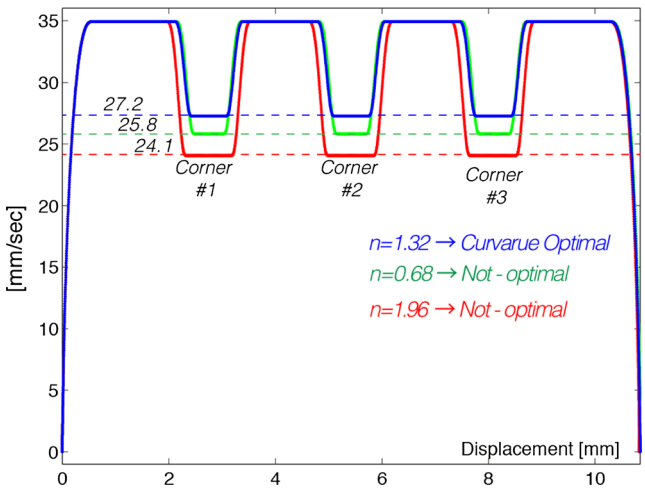

Fig. 11 Scheduled feedrate profile along rectangular tool path

5 Conclusions

A practical smooth motion generation algorithm for linear segmented tool paths is presented in this paper. Quintic Bezier blends with six control points are designed to blend the discrete linear segments with curvature continuous transitions. The feedrate along the multi-segmented tool path is scheduled efficiently to minimize the total cycle time and interpolated smoothly.

Compared to the previous works, the proposed algorithms in this paper have the following advantages: (1) The transition scheme can realize G^2 continuity, optimum curvature for high-speed cornering, and also control of the

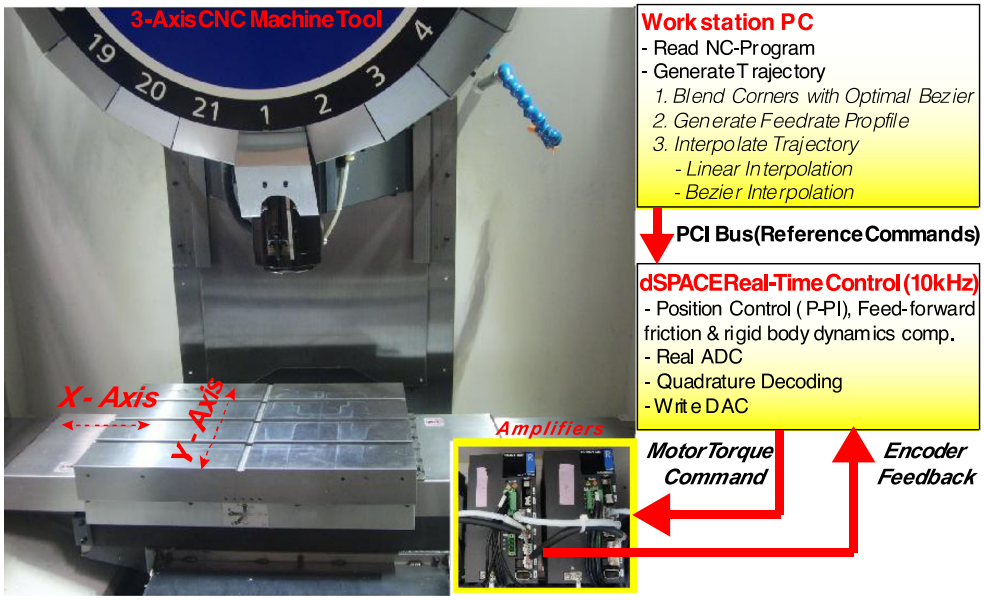

Fig. 12 Experimental setup

cornering errors. (2) The feed motion planning algorithm utilizes S-curve-type acceleration profile and schedules the feedrate efficiently along corner-blended tool path to respect the acceleration limits of the drives. (3) Feedrate transition from linear to Bezier segments and also the feed motion along the Bezier curve are interpolated accurately eliminating unwanted fluctuations. Finally, proposed algorithms are validated in simulations and also implemented for complex 2-D tool path contouring on the in-house controlled CNC machine tool.

Appendix

Curvature along the Bezier blend can be derived analytically from the well-known expression,

$$\kappa(u) = \frac{|\mathbf{B}_u(u) \times \mathbf{B}_{uu}(u)|}{|\mathbf{B}_u(u)|^3}$$

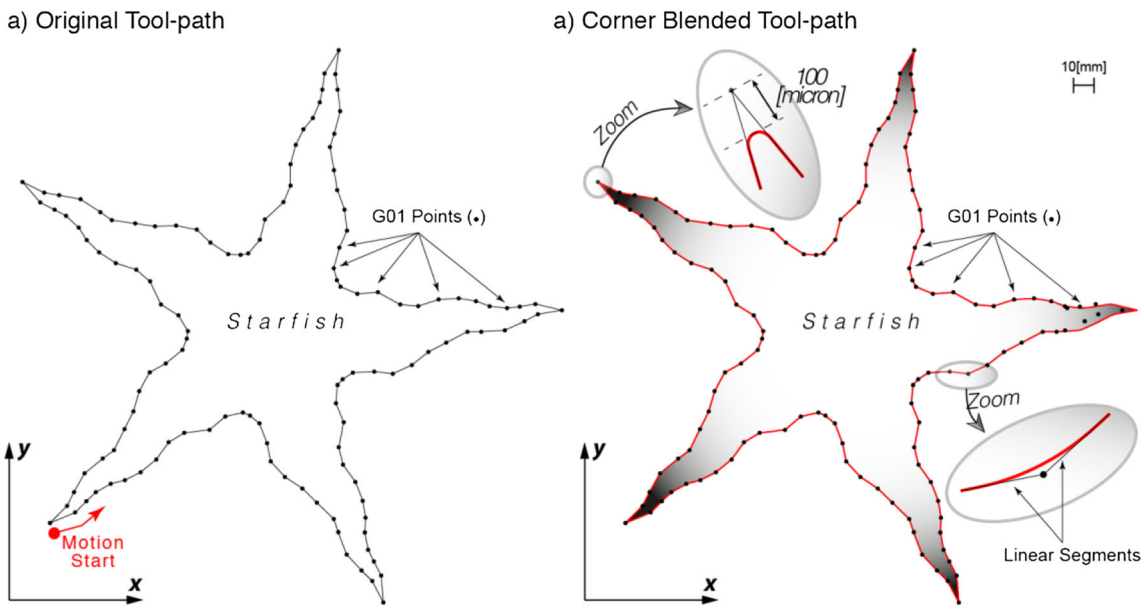

Fig. 13 Experimental planar tool path

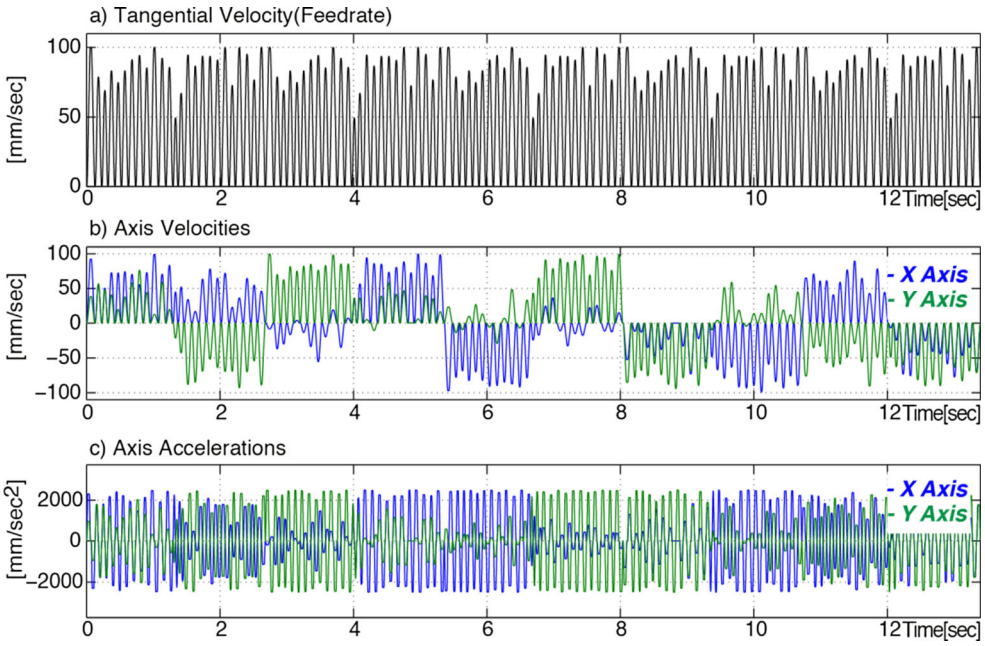

Fig. 14 Profiles during point-to-point motion

A single Bezier blend is illustrated in Fig. 17. The corner transition point is the origin of the Cartesian coordinate system, and control points can be written from Fig. 17 as

$$\left.\begin{aligned} \mathbf{P}_0 &= \begin{bmatrix} d(2n+1) \\ 0 \end{bmatrix}, \quad \mathbf{P}_1 = \begin{bmatrix} d(n+1) \\ 0 \end{bmatrix}, \quad \mathbf{P}_2 = \begin{bmatrix} d \\ 0 \end{bmatrix} \\ \mathbf{P}_3 &= \begin{bmatrix} d\cos(\theta) \\ d\sin(\theta) \end{bmatrix}, \quad \mathbf{P}_4 = \begin{bmatrix} d(n+1)\cos(\theta) \\ d(n+1)\sin(\theta) \end{bmatrix}, \quad \mathbf{P}_5 = \begin{bmatrix} d(2n+1)\cos(\theta) \\ d(2n+1)\sin(\theta) \end{bmatrix} \end{aligned}\right\} \tag{44}$$

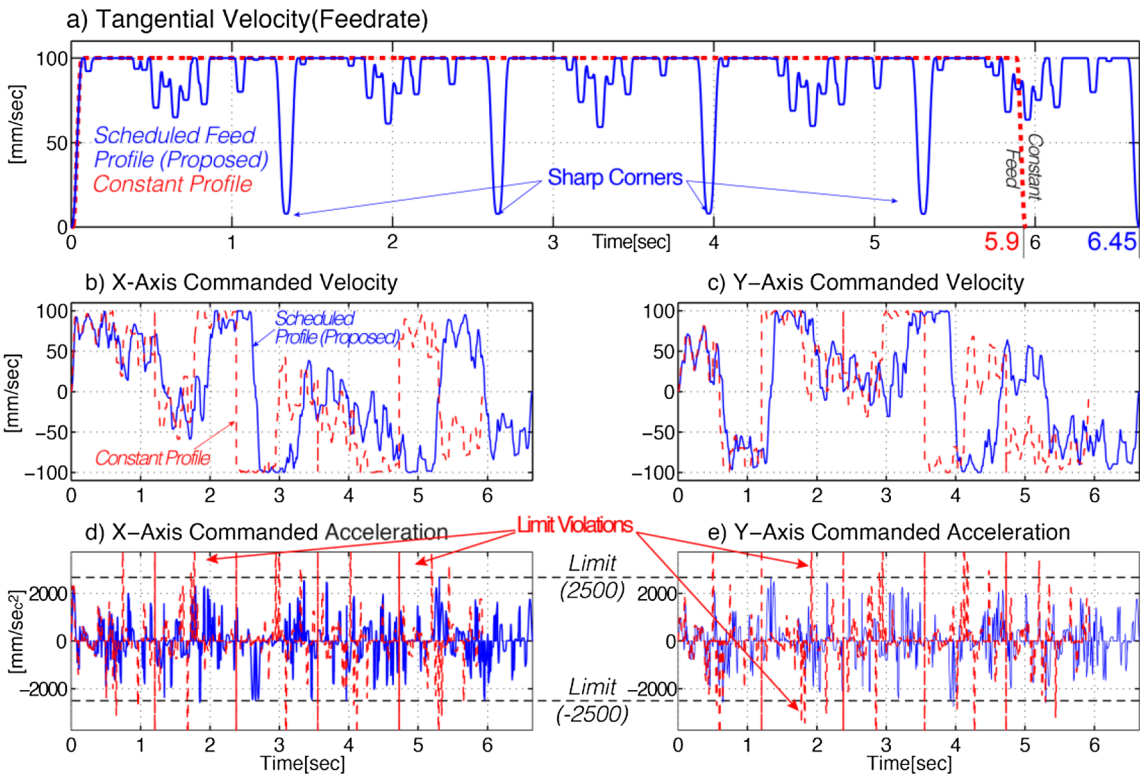

Fig. 15 Comparison of scheduled and constant feed profiles

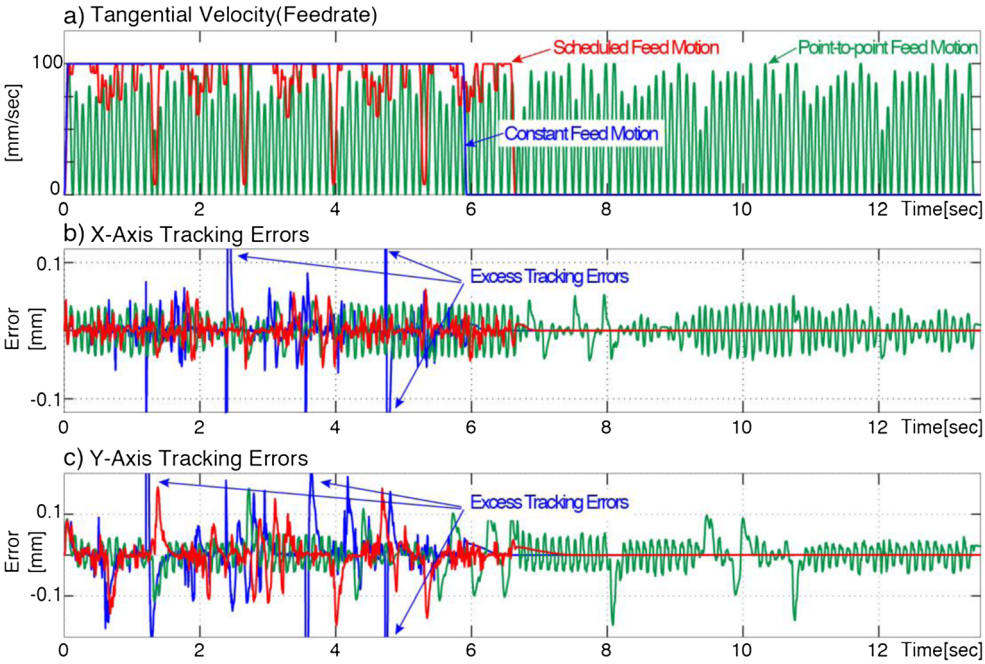

Fig. 16 Experimentally recorded tracking errors of the axes

where $n=c/d$, and the Quintic Bezier Blend is parameterized by plugging the control points from Eq. 44 into Eq. 1 as

$$\mathbf{B}(u)=\left\{\begin{array}{l} B_x=\dfrac{L(\cos\theta u^4-5u-3\cos\theta+10u^3-10u^4+3u^5+2)}{2} \\ \quad -\dfrac{5Lu(u-1)(u+4u^2\cos\theta-3\cos\theta u^3-5u^2+3u^3+1)}{2n+1} \\ B_y=\dfrac{Lu\sin\theta(5nu-15u-3nu^2+6u^2+10)}{2n+1} \end{array}\right\} \tag{45}$$

where $L_T=2c+d$, θ is the inner cornering angle, and $u\in[0,1]$. The 1st and 2nd derivatives, $\mathbf{B}=\frac{d\mathbf{B}(u)}{du}$ and $\mathbf{B}_{uu}=\frac{d\mathbf{B}(u)}{du^2}$, can be computed from Eq. 45 to evaluate the curvature along the Bezier blend as

$$\kappa(u)=\frac{-12(2n+1)nu\sin\theta(u-1)(nu-2u-2nu^3+nu^4+4u^3-2u^4+1)}{L_T A^{\frac{3}{2}}} \tag{46}$$

where

$$\left.\begin{array}{l} A=a_1u^8+a_2u^7+a_3u^6+a_4u^5+a_5u^4+a_6u^3+a_7u^2+n^2 \\ a_1=(1-\cos)\left(72-72n+18n^2\right) \\ a_2=288\cos\theta+288n(1-\cos\theta)-72n^2(1-\cos\theta) \\ a_3=432(1-\cos)+432n\cos+n^2(116-100\cos\theta) \\ a_4=288(\cos\theta-1)+288n(1-\cos\theta)+48n^2(\cos\theta-2) \\ a_5=72(1-\cos\theta)-60n(1-\cos\theta)+6n^2(5+\cos\theta) \\ a_6=24n(\cos\theta-1)+8n^2(2-\cos\theta) \\ a_7=12n(1+\cos\theta)+12n^2 \end{array}\right\} \tag{47}$$

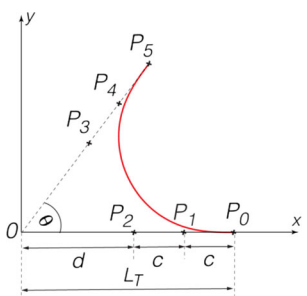

Fig. 17 Single Bezier blend

References

1. Piegl L, Tiller W (2003) The NURBS book, 2nd edn. Springer, Berlin Heidelberg
2. Koren Y, Lin R-S (1995) Five-axis surface interpolators. Ann CIRP 44(1):379–382
3. Zhang QG, Greenway RB (1998) Development and implementation of a NURBS curve motion interpolator. Robot Comput Integr Manuf 14(1):27–36

4. Langeron JM, Duc E, Lartigue C, Bourdet P (2004) A new format for 5-axis tool path computation, using Bspline curves. Comput Aided Des 36(12):1219–1229
5. Timar S, Farouki R, Smith T, Boyadjieff C (2005) Algorithms for time-optimal control of CNC machines along curved tool paths. Robot Comput Integr Manuf 21:37–53
6. Wang F-C, Wright PK, Barsky BA, Yang DCH (1999) Approximately arc-length parameterized C3 quintic interpolatory splines. J Mech Des-T ASME 121(3):430–439
7. Erkorkmaz K, Altintas Y (2005) Quintic spline interpolation with minimal feed fluctuation. J Manuf Sci E-T ASME 127(2):339–349
8. Barre PJ, Bearee R, Borne P, Dumetz E (2005) Influence of a jerk controlled movement law on the vibratory behaviour of high-dynamics systems. J Intell Robot Syst 42(3):275–293
9. FANUC (2006) 5-axis machining features FANUC series 30i-model a/31i-model A5. Int J Mach Tool Manu 46:235–242
10. Siemens (2009) Milling with SINUMERIK: 5-axis machining manual
11. Wang J-B, Yau H-T (2009) Real-time NURBS interpolator: application to short linear segments. Int J Adv Manuf Technol 41(11–12): 1169–1185
12. Tsai M-S, Nien H-W, Yau H-T (2010) Development of a real-time look-ahead interpolation methodology with spline-fitting technique for high-speed machining. Int J Adv Manuf Technol 47(5–8):621–638
13. Fleisig RV, Spence AD (2001) Constant feed and reduced angular acceleration interpolation algorithm for multi-axis machining. Comput Aided Design 33(1):1–15
14. Goldenthal R, Bercovier M (2004) Spline curve approximation and design by optimal control over the knots. Computing 72(1–2):53–64
15. Werner H, Thomas R (2004) Smoothing rational B-spline curves using the weights in an optimization procedure. Comput Aided Geom D 12(8):837–848
16. Macfarlane S, Croft EA (2001) Design of jerk bounded trajectories for on-line industrial robot applications. Proc IEEE Int Conf Robot Autom 1:979–984
17. Jouaneh MK, Wang Z, Dornfeld DA (1990) Trajectory planning for coordinated motion of a robot and a positioning table. Part 1. Path specification. IEEE Trans Robot Autom 6:735–745
18. Jouaneh MK, Dornfeld DA, Tomizuka M (1990) Trajectory planning for coordinated motion of a robot and a positioning table. Part 2. Optimal trajectory specification. IEEE Trans Robot Autom 6:746–759
19. Yutkowitz SJ, Chester W (2005) Apparatus and Method for Smooth Cornering in a Motion Control System. United States, Siemens Energy & Automation, Inc, Alpharetta, GA, (US Patent 6922606)
20. Pateloup V, Duc E, Ray P (2010) B-spline approximation of circle arc and straight line for pocket machining. Comput Aided Design 42: 817–827
21. Zhang L, You Y, He J, Yang X (2011) The transition algorithm based on parametric spline curve for high-speed machining of continuous short line segments. Int J Adv Manuf Technol 52:245–254
22. Zhao H, Zhu L-M, Ding H (2013) A real-time look-ahead interpolation methodology with curvature-continuous B-spline transition scheme for CNC machining of short line segments. Int J Mach Tool Manu 65:88–98
23. Bi Q, Wang Y, Zhu L, Ding H (2011) A practical continuous-curvature Bezier transition algorithm for high-speed machining of linear tool path. J Intell Robot Syst 7102:465–476
24. Ekorkmaz K, Yeung C-H, Altintas Y (2006) Virtual CNC system. Part II. High speed contouring application. Int J Mach Tool Manu 46(10):1124–1138
25. Beudaert X, Lavernhe S, Tournier C (2013) 5-axis local corner rounding of linear tool path discontinuities. Int J Mach Tool Manu 73:9–16
26. Sencer B, Susumu I, Shamoto E (2014) Curvature-continuous Sharp Corner Smoothing Scheme for Cartesian Motion Systems. 13th IEEE International Workshop on Advanced Motion Control, Yokohama, Japan
27. Erkorkmaz K, Altintas Y (2001) High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation. Int J Mach Tool Manu 41(9):1323–1345
28. Lin R-S (2000) Real-time surface interpolator for 3-D parametric surface machining on 3-axis machine tools. Int J Mach Tool Manu 40(10):1513–1526
29. Sencer B, Altintas Y, Croft E (2008) Feed optimization for five-axis CNC machine tools with drive constraints. Int J Mach Tool Manu 48(7–8):733–745
30. Altintas Y, Altintas Y, Erkorkmaz K (2003) Feedrate optimization for spline interpolation in high speed machine tools. CIRP Ann 52:297–302
31. Heng M, Erkorkmaz K (2010) Design of a NURBS interpolator with minimal feed fluctuation and continuous feed modulation capability. Int J Mach Tool Manu 50:281–293
32. Kreyszig E (1991) Differential geometry, 1st edition. Dover Publications