International Journal of Machine Tools & Manufacture 50 (2010) 394–403

Contents lists available at ScienceDirect

International Journal of Machine Tools & Manufacture

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

Trajectory generation in high-speed, high-precision micromilling using subdivision curves

Athulan Vijayaraghavan [a,*], Angela Sodemann [b], Aaron Hoover [a], J. Rhett Mayor [b], David Dornfeld [a]

[a] University of California, Berkeley, USA
[b] Georgia Institute of Technology, USA

ARTICLE INFO

Article history:
Received 16 March 2009
Received in revised form
23 September 2009
Accepted 20 October 2009
Available online 5 November 2009

Keywords:
Micromilling
Trajectory generation
Subdivision surfaces

ABSTRACT

Motion control in high-speed micromilling processes requires fast, accurate following of a specified curvilinear path. The accuracy with which the path can be followed is determined by the speed at which individual trajectories can be generated and sent to the control system. The time required to generate the trajectory is dependent on the representations used for the curvilinear trajectory path. In this study, we introduce the use of subdivision curves as a method for generating high-speed micromilling trajectories. Subdivision curves are discretized curves which are specified as a series of recursive refinements of a coarse mesh. By applying these recursive properties, machining trajectories can be computed very efficiently. Using a set of representative test curves, we show that with subdivision curves, trajectories can be generated significantly faster than with NURBS curves, which is the most common method currently used in generating high-speed machining trajectories. Trajectories are computed efficiently with subdivision curves as they are natively discretized, and do not require additional evaluation steps, unlike in the case of NURBS curves. The reduced trajectory generation time allows for improved performance in high-speed, high-precision micromilling. We discuss the use of several metrics to quantify the quality of the subdivision interpolation, and apply them in calculating the error during trajectory generation for the test curves.

1. Introduction

Motion control in high-speed micromilling processes requires fast, accurate following of a specified curvilinear path. In high-speed micromachining of parts which typically contain small features of very high curvature the requirements on accuracy can become extreme. In these applications, the accuracy with which the path can be followed is determined by the speed at which individual trajectories can be generated and sent to the control system. The time required to generate the trajectory is dependent on the representations used for the curvilinear trajectory path. Common representations include point sets with linear or circular interpolation, polynomial splines, and more recently, non-uniform, rational, b-spline (NURBS) curves. It is critical for the trajectory computation time to be minimal in order to limit chord error. As feature scales shrink and machining feedrates increase, the requirements on the trajectory generation system increase, and numerical methods are required to very rapidly compute trajectories.

We discuss in this paper the use of subdivision curves as a method for generating high-speed micromilling trajectories. Subdivision curves are discretized curves which are specified as a series of recursive refinements of a coarse mesh. Using the recursive properties of subdivision curves, the machining trajectories can be computed very efficiently. Catmull–Clark subdivision curves are used in representing the trajectories; these curves converge at the limit (infinite recursion) to cubic b-spline curves, and have properties of continuity comparable to that of cubic NURBS curves. The trajectory generation methodology discussed in this paper is to vary the discretization (or subdivision level) of the Catmull–Clark curves to match the accuracy required in the trajectory vectors. Since these curves are natively discretized, additional discretization steps are not required—unlike in the case of NURBS curves—and the trajectories are computed very efficiently. The discrete representation can also be exploited to develop error measures for the trajectory curve which can be used to estimate the quality of the machined features and predict phenomena such as gouging. We also compare this trajectory generation method to NURBS-based methods, and we show that a significant improvement in performance can be achieved.

* Corresponding author.
E-mail addresses: athulan@berkeley.edu (A. Vijayaraghavan), asodemann3@gatech.edu (A. Sodemann), ahoover@me.berkeley.edu (A. Hoover), rhett.mayor@me.gatech.edu (J. Rhett Mayor), dornfeld@berkeley.edu (D. Dornfeld).

0890-6955/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijmachtools.2009.10.010

2. Background and related work

2.1. Interpolation methods

In high-speed, high-precision micromilling, the spindle path is often initially specified as an ordered set of locating points through which the spindle is required to travel. If the distance between specified points is sufficiently large, a point-to-point travel method may allow for accurate path following. If the distance between points is small, however, this method results in frequent velocity variation [1,2], slow implementation [3], and discontinuities in the travel path. Data transmission errors [4] or delay are also possible as the data point density becomes large. In order to avoid these error sources in cases which require high point densities, the set of data points is first interpolated to obtain a parametric description of the spindle location data points.

Common interpolation methods include linear, circular, polynomial spline, and the NURBS method. Each of these methods has certain advantages and disadvantages for path following. The linear and circular methods consist of determining one or more lines and circles which most closely approximate the set of data points using a least-squared-error fit. This method—sometimes referred to as "reference word interpolation" [5]—is easy to implement [6]. The resulting interpolation is simple to apply to a trajectory-following algorithm. However, this method may result in large amounts of interpolation error for data sets which do not approximately describe lines and circles [7], or which require a large number of linear and circular segments to approximate the data set to within specified error limits. The large number of segments results in the same drawbacks as point-to-point following: feedrate fluctuation and discontinuity.

The polynomial spline method of interpolation may be used in cases of data sets which do not closely resemble lines and circles. In this method, splines of a specified degree (often cubic or quintic splines) are fit to the specified spindle location data points. This method can be used to fit a non-linear set of data points with fewer individual segments, while still maintaining ease of implementation [8]. However, this method still suffers from a lack of continuity between individual segments. The NURBS method is an improvement to the polynomial spline method, as continuity between the spline segments is enforced. In addition, the NURBS method requires less memory than other methods [1,8,9] and is computationally stable [1].

2.2. Real-time trajectory generation

Once a parametric description has been found for a set of location data points, the description can be utilized in trajectory generation. Trajectory generation refers to the process of determining time-dependent individual velocities for each axis of the machine tool. At each sampling instant, trajectories are chosen which will result in the most accurate following of the specified spindle path. This process should be distinguished from the motion control loop, which seeks to accurately achieve the specified velocities.

The trajectory generation process samples the interpolated curve at discrete intervals such that a fixed chord length is achieved between the sampled points. The chord length is determined based on the specified feedrate and the minimum sampling time of the controller. Since the minimum sampling time is known a priori, the chord length is constant for a given feedrate. The trajectory generation algorithm is employed to compute individual stage velocities during each time step. In some cases, this can be done offline as the curve to be followed is completely specified. However, in cases where high following accuracy is required, and in applications where the following errors may be large (such as high-speed high-precision micromilling), a suitable real-time error compensation method must be applied to the trajectory generation algorithm. Since errors are measured in real-time, the trajectory generation (and the compensation) must be computed in real-time as well.

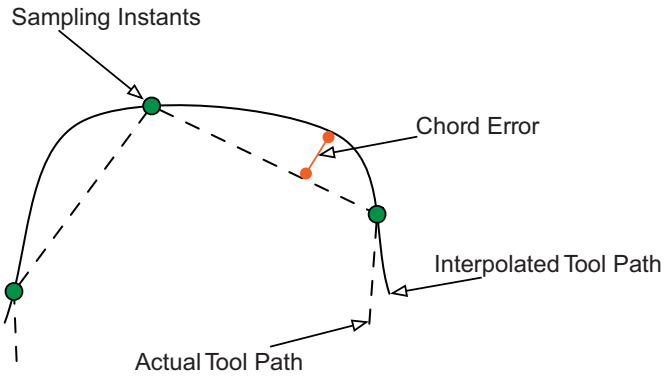

Fig. 1. Chord error in trajectory generation.

The process of trajectory generation gives rise to a source of geometrical error known as "chord error" [10]. This error is illustrated in Fig. 1.

Chord error increases with increasing feedrate, decreasing feature size, and decreasing sampling rate. In order to constrain the effects of chord error to a specified error limit, the feedrate is typically limited. Eq. (1) describes the method by which feedrate is limited in micromilling [11].

$$f \le \frac{2}{T_s}\sqrt{(\rho - r) - ((\rho - r) - \delta_{\max})^2} \tag{1}$$

Here, ρ is the radius of curvature of the feature, r is the tool radius, T_s is the sampling time in seconds, and $\delta_{\max}$ is the maximum chord error.

Eq. (1) reveals that high values of sampling time (low sampling rate) can severely limit the maximum feedrate achievable, particularly as the feature radius of curvature decreases and the required precision increases. In order to achieve the high feedrates required in high-speed, high-precision micromilling, a high sampling frequency (or small sampling period) is needed. This can only be accomplished in real-time by increasing the speed of the trajectory generation.

A review of the literature reveals that the most widely studied method of interpolation and trajectory generation is using NURBS curves.

2.3. Interpolation trajectory generation by NURBS

The NURBS method can be used either as a method of approximation or as a method of interpolation. The interpolation method is used to create a set of splines of specified order which pass through all data points specified, while the approximation method is used to create a set of splines which pass nearby all data points specified to within a set error limit. In this paper, we compare the NURBS method of interpolation against the subdivision curve method.

The NURBS method of interpolation consists of finding the control points for a set of approximating splines of a specified continuity to interpolate all the given data points. We refer the reader to [12] for a full derivation of NURBS interpolation.

The NURBS method suffers from a key disadvantage when applied to a real-time motion control system. The rate at which

trajectories can be generated from a NURBS path description is highly limited due to the computational complexity of this method [1,13]. This limitation can cause large amounts of chord error in regions having high curvatures in the path, and may require low feedrates to compute the trajectories in real-time.

In this paper we propose subdivision curves as an improvement over NURBS for real-time trajectory generation in high-speed, high-precision micromilling. Due to the reduced computational complexity of this method, real-time trajectory generation times are reduced. Thus, this method can be employed in cases of required high precision, high feedrates, and/or high curvatures.

3. Trajectory generation with subdivision curves

3.1. Subdivision surfaces

A subdivision surface, e.g. [14–16], is a generalization to three dimensions of a boundary representation expressed as the limit of a sequence of successive refinements on a given input control curve. They are specified using a control mesh of fixed topology and a subdivision scheme that is applied recursively to the mesh [17]. Each subdivision step increases the number of vertices in the mesh, and the subdivision scheme specifies the position and topology of new vertices in the mesh at each step. Since most subdivision schemes are based on splines, the schemes are generally local; that is, the position of new vertices are usually based on their 1- or 2-ring neighbors. Subdivision schemes are generally devised to ensure strict analytical properties at the limit (the surface generated when the scheme is applied infinite times). Analytically the limit surface behaves like piecewise polynomial patches.

In this work, however, we are concerned with the generation and evaluation of inherently two dimensional toolpaths. As such, we apply the subdivision method in only two dimensions to efficiently generate approximations of spline curves. There are several subdivision schemes that can be applied for trajectory representation and generation. Schemes can be classified based on the smoothness that can be achieved (C^n) and if they are approximating or interpolating [17]. Some popular subdivision schemes are classified in Table 1.

In this paper, we use the Catmull–Clark subdivision scheme in two dimensions to represent the machine tool trajectories. This scheme is based on tensor product bicubic splines, and generates a C^2 curve at the limit for regular meshes [14,21]. It also can be combined with so-called sharp schemes to pin specific vertices of the mesh [22]. The Catmull–Clark scheme is very versatile, and since it was derived from bicubic curves, it is closely analogous to NURBS curves. While other schemes can be just as effective, the close relationship with NURBS curves makes it very suitable for this application. Fig. 2 shows an example of Catmull–Clark subdivision applied to a 2D curve.

Table 1
Classification of subdivision schemes.

Name	Type	Smoothness
Loop [15]	Approximating	C^2
Modified Butterfly [16]	Interpolating	C^1
Catmull–Clark [14]	Approximating	C^2
Kobbelt [18]	Interpolating	C^1
Doo–Sabin [19]	Approximating	C^1
Root-three ($\sqrt{3}$) [20]	Approximating	C^2

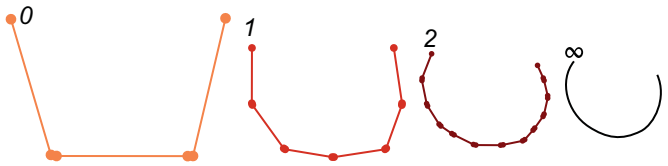

Fig. 2. Catmull–Clark subdivision applied to 2D curve. Numbers denote levels of subdivision.

3.2. Basic trajectory generation

To generate machine tool trajectories using subdivision curves, we begin with the set of spindle locating points that need to be interpolated. A cubic spline curve is fit through these points, and the control points of the spline are used as the control points of the subdivision curves. This ensures that the subdivision curve interpolates the same points as the NURBS curve.

The trajectory generation samples the subdivision curve at discrete intervals such that a fixed chord length is achieved between the sampled points. The chord length is determined based on the feedrate of the machine tool and the minimum sampling time of the controller. For the purposes of this analysis, and to ensure a constant maximum feed per tooth, we assume that the feedrate during cutting is fixed. The minimum sampling time is known a priori, thus the chord length the machine tool has to travel in each time step is constant. The trajectory generation algorithm computes individual velocities for the machine tool axes during each time step. While trajectory generation can be pre-computed as the curve is completely specified, error compensation needs to be applied in real-time, and hence the trajectory generation is also done in real-time. We first look at the basic trajectory generation algorithm for a given subdivision curve.

From the control points of the fitted spline, we have a subdivision curve of known topology with control points c. From the subdivision formulation, we know that:

$$Mc = x \tag{2}$$

where x are the subdivided points (the *actual* curve), and M is the subdivision tensor at some required level of subdivision. The subdivision tensor is a mathematical way of applying the multiple recursion steps to the subdivision control mesh, and is a function of the subdivision level. The subdivision level is chosen such that the maximum spacing of points in the subdivided curve is less than the chord length the machine tool travels during each time step. This ensures that the interpolation is accurate and that no regions of the curve are "missed" during machining.

M is a $(p \times q)$ matrix; there are p points in the subdivided curve and q points in the input control mesh. Hence, we can represent x as

$$x = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_p \end{bmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_q \end{pmatrix} \tag{3}$$

where m_i is the i th row in M, and c_i is the i th control point.

The goal of the method is to find a set of points, P, separated by the required chord length, $L_{required}$, that lie on the curve. The vector difference between adjacent points in P is the trajectory between the points. The points in P are found sequentially as the position of p_{i+1} is dependent on p_i and the trajectory error the machine tool has encountered in traveling to p_i.

The crux of the trajectory generation algorithm lies in generating p_{i+1} for a given p_i using the arc length parametrization of the curve. From the arc length parametrization, the first point

along the curve that is arc length $L_{required}$ away from p_i is calculated. p_{i+1} necessarily cannot be before this point, as the chordal distance between two points on a curve cannot be larger than the arc length between the two points. From this point the pairwise chordal distance between the p_i and each subsequent point is computed, until the point located at a distance greater than the chordal length is reached, based on which p_{i+1} is computed.

Mathematically, the algorithm is developed as follows. The arc length parametrization is calculated based on the set of distance between adjacent control points, D. D is pre-computed very efficiently using the subdivision tensor of the curve; from Eq. (3), D is calculated as follows:

$$D = \begin{bmatrix} m_1 - m_2 \\ m_2 - m_2 \\ \vdots \\ m_{p-1} - m_p \end{bmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_q \end{pmatrix} \tag{4}$$

Using the arc length parametrization, the start point for finding p_{i+1} is identified. From this point, the chordal length between p_i and successive points on the subdivided curve is evaluated (as the euclidean distance between the two points). If x denotes the points on the curve, then point x_j can be found which is the closest point to p_i located further than the distance $L_{required}$. The point p_{i+1} clearly lies on the curve between x_{j-1} and x_j. p_{i+1} is calculated by interpolating between x_{j-1} and x_j. This is done as:

$$p_{i+1} = \alpha x_j + (1-\alpha)x_{j-1} \tag{5}$$

where,

$$\alpha = \frac{L_{required} - \|p_i - x_{j-1}\|}{\|x_j - x_{j-1}\|} \tag{6}$$

The trajectory T_i is then

$$T_i = p_{i+1} - p_i \tag{7}$$

The distance the machine travels in this step is: $d_i = \|T_i\|$

However, this trajectory term does not account for the trajectory-following error in the machine. Discrepancies between the spindle location at each step (s_i) and the evaluated position (p_i) must be compensated for during trajectory generation. A schematic of the trajectory generation is shown in Fig. 3.

4. Error metrics

As subdivision curves are naturally discretized, it is straighforward to compute positional errors during trajectory generation. The actual positions of the spindle can be compared to the commanded positions, and the actual distance traveled by the spindle can also be compared to the length that was required to be traveled. Based on this, we define the following error terms to measure the quality of the subdivision surface itself, as well as the quality of the trajectory generation algorithm. The quality of the subdivision surface is dependent upon the level of subdivision used in the surface relative to the chordal length ($L_{required}$).

The error metrics are as follows (where $\langle a-b \rangle = \sqrt{\sum_{i=0}^{N}(\|a_i - b_i\|)^2}$):

1. $\varepsilon_1 = \langle L_{actual} - L_{commanded} \rangle$—Is a comparison of the actual distance traveled in each step and the commanded distance. This metric is an indicator of the error in the movement of the spindle, as it is related to the mismatch between the actual and commanded positions during interpolation.
2. $\varepsilon_2 = \langle L_{actual} - l_{step} \rangle$—Is a comparison of the actual distance traveled in each step and the step length at the given feedrate and controller sampling time. This metric can be used to confirm if errors in trajectory generation are due to poor algorithm performance or poor machine tool performance. Along with this error, if ε_1 is large, then it indicates that the machine tool is exhibiting a large amount of following error. If, along with this error, ε_3 is also large, it indicates that the subdivision algorithm is performing poorly.
3. $\varepsilon_3 = \langle L_{commanded} - l_{step} \rangle$—Is a comparison of the commanded distance traveled in each step and the step length at the given feedrate and controller sampling time. This metric is an indicator of the effectiveness of the trajectory generation algorithm. The algorithm is required to generate at each step a trajectory such that the distance traveled is equal to the theoretical distance the spindle can travel in one time-step at the given feedrate.
4. $\varepsilon_4 = \langle L_{actual} - L_{arclength} \rangle$—Is a comparison of the actual distance (or the chord-length) and the arc-length along the set of data points being interpolated. This metric is an indicator of the discretization of the subdivided curve. If ε_4 is large and ε_1 is small, it indicates that the chord-length is much larger than the arc-length, and directly implies that the subdivision is coarse and must be refined. In addition to this, if ε_2 is small as well, this can be used to predict gouging during machining. For gouging to occur, the chord-length distance traveled along the curve must be larger than the arc-length for that region of the interpolant. A check is performed to determine if $\varepsilon_1, \varepsilon_2$ are small to ensure that this discrepancy is due to the curve characteristics, rather than to poor machine performance or algorithm performance.
5. $\varepsilon_5 = \langle s_i - p_i \rangle$—Compares the actual positions s_i and the required positions p_i over all the trajectory steps. This is a measure of the quality of the overall trajectory generation, and is related to ε_1.

Table 2 summarizes what the different error metrics indicate.

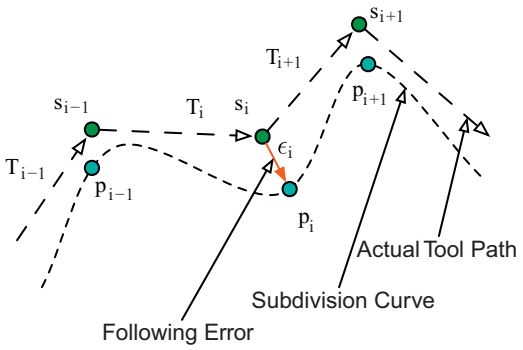

Fig. 3. Schematic of trajectory generation.

5. Performance tests

The subdivision trajectory generation and the NURBS trajectory generation were implemented using MATLAB, Version 2007a.

Table 2
Error metric indicators.

Error metrics	Types of errors
ε_1	Following error
ε_2	Trajectory algorithm
ε_3	Trajectory algorithm
ε_4	Curve discretization, gouging
ε_5	Following error

MATLAB is used in the implementation as it provides similar computational libraries to compare the two approaches. Whilst OpenGL libraries are available for NURBS interpolation, comparable subdivision code libraries are not available. Therefore, by consistently applying MATLAB for the comparative study based on the theoretical formation of both approaches, the unintentional bias of the results in favor of optimized NURBS code over non-optimized sub-division code was necessarily avoided. All tests were performed on a 2.8 GHz Intel Xeon Quad-core PC with 4GB memory.

To test the robustness of the algorithms against machine tool errors, a random error generator was used to add noise to the trajectory generation. The error generator randomly changed the goal position of the spindle as it began traversing a trajectory. The error was generated as a uniformly distributed random point within a "bubble" of fixed size. The size of the bubble was specified relative to the chord length during interpolation. Position error was compensated in all cases.

5.1. Example curves

The performance of the subdivision and NURBS trajectory generation algorithms was tested by computing the trajectory generation time for a variety of curves. Plots of the curves used in this study are shown in Fig. 4. These curves were selected for smoothly varying features, and are representative of the complex geometries that may be machined at the micro-scale. Each of these curves were represented by a set of spindle location data points which were subsequently interpolated by both NURBS interpolation and subdivision curves interpolation. The average spacing between adjacent spindle location data points is approximately 100 μm.

5.2. Computational time comparison

The runtime for a complete set of trajectory generation events was calculated using NURBS interpolation, and was repeated

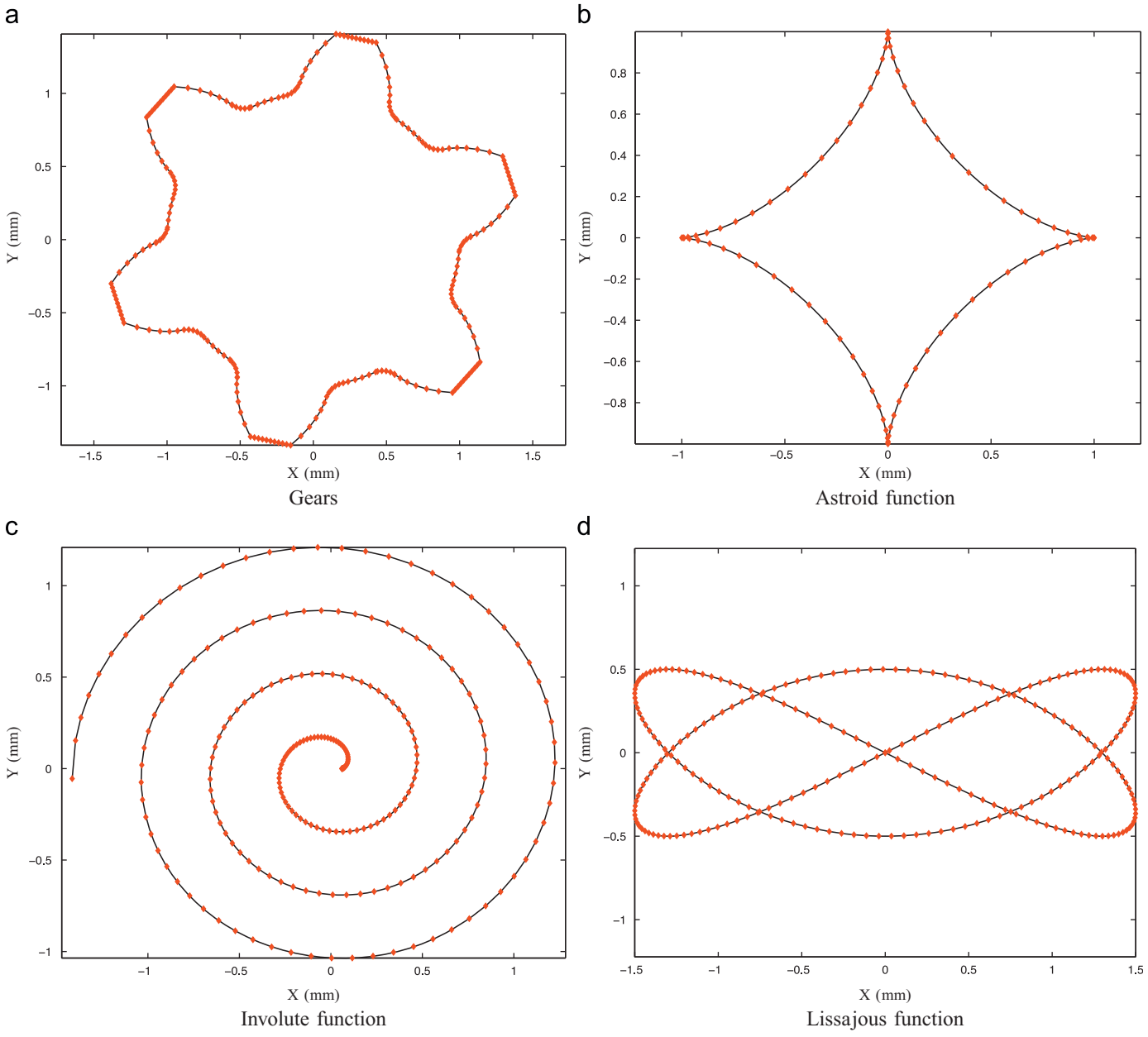

Fig. 4. Example curves.

using subdivision interpolation. The time required to generate a single trajectory was compared for the two interpolation methods. A range of controller sampling periods for a fixed feedrate of 1 mm/s, and a range of feedrates for a fixed sampling period of 2.5 ms was considered. The random bubble size was set to be 10% of the step length in both cases. The subdivision limit was determined such that the largest feature in the subdivided curve was smaller than the minimum step length. This was done to ensure that the trajectory generation would proceed forward in every step. If this condition is not met, it is possible that the algorithm would keep interpolating between the same two points in successive steps, leading to errors in the trajectory generation.

The computational time per step for the NURBS and the subdivision cases for the four test cases are shown in Fig. 5 for the sampling period case, and in Fig. 6 for the feedrate case. The subdivision method provides a 3-10X improvement in speed over NURBS in all the cases.

As the controller sampling period increased, the trajectory generation time for the NURBS method did not increase, as the complexity of the NURBS curve does not depend upon the distance traveled during each time step. In contrast, trajectory generation time for the subdivision method decreases as the sampling period increases. This is because the chord length decreases with decreasing sampling frequency, and the subdivision level required to accurately traverse the curve increases with decreasing chord length. The decrease is linear for the most part, but some non-linearity is introduced due to changes in the subdivision level that result from curve geometry and the spacing of the initial data points. Even given this increase, however, the subdivision method is faster than the NURBS method.

5.3. Applying the error metrics

The error metrics can be used in estimating the effectiveness of the trajectory generation algorithm. We calculate the error metrics for the gears curve, which contains typical micromilled features. The results for this curve provide a representative sample of errors found in the test curves.

Consider metric ε_1, which can be used as an indicator of the machine performance. Fig. 7 is a plot of ε_1 calculated at different feedrates and different amounts of simulated machine trajectory-following errors, for a fixed controller sampling time of 5 ms. This figure indicates that this error metric does not change

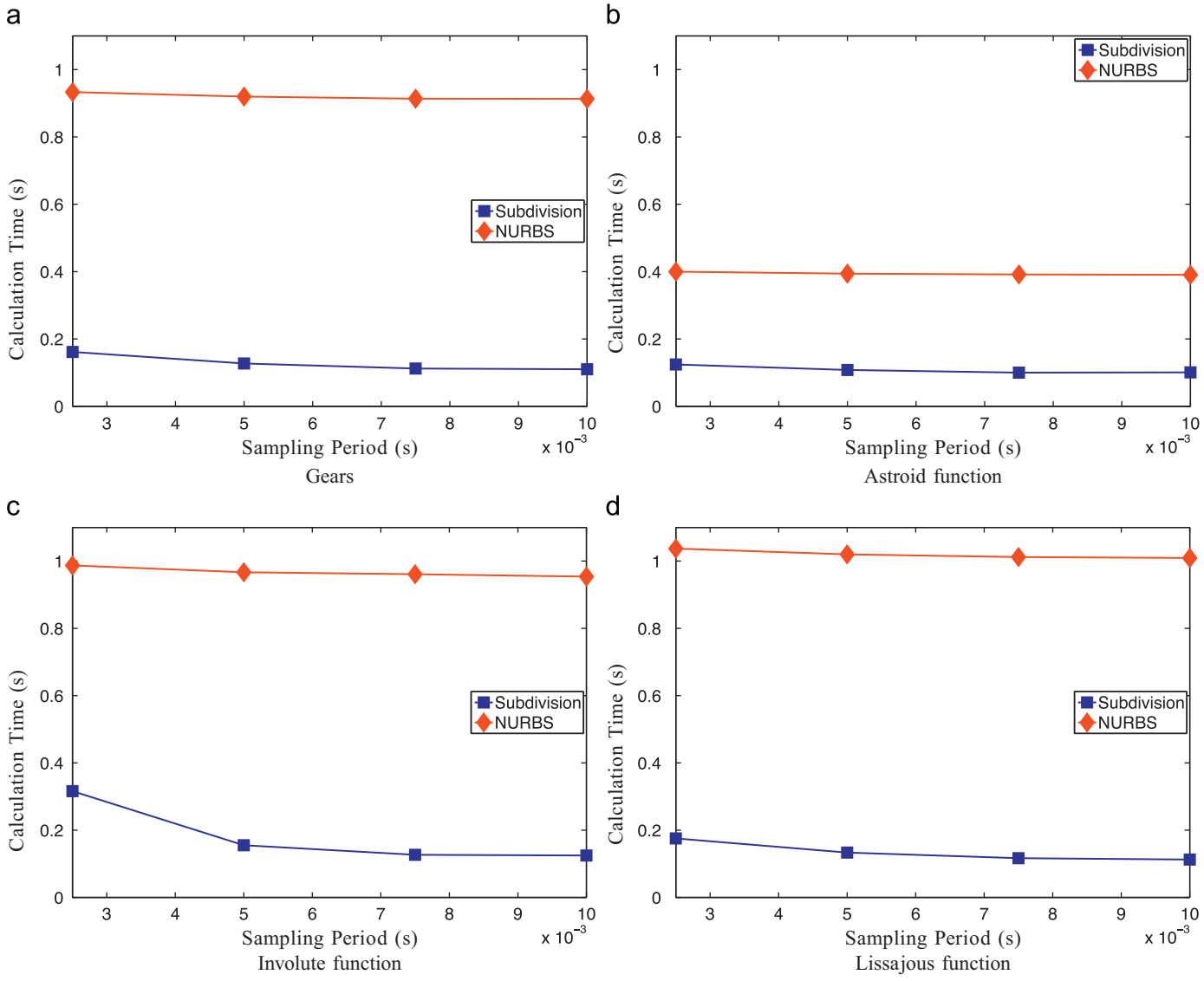

Fig. 5. Comparing NURBS and subdivision trajectory generation—sampling period variation.

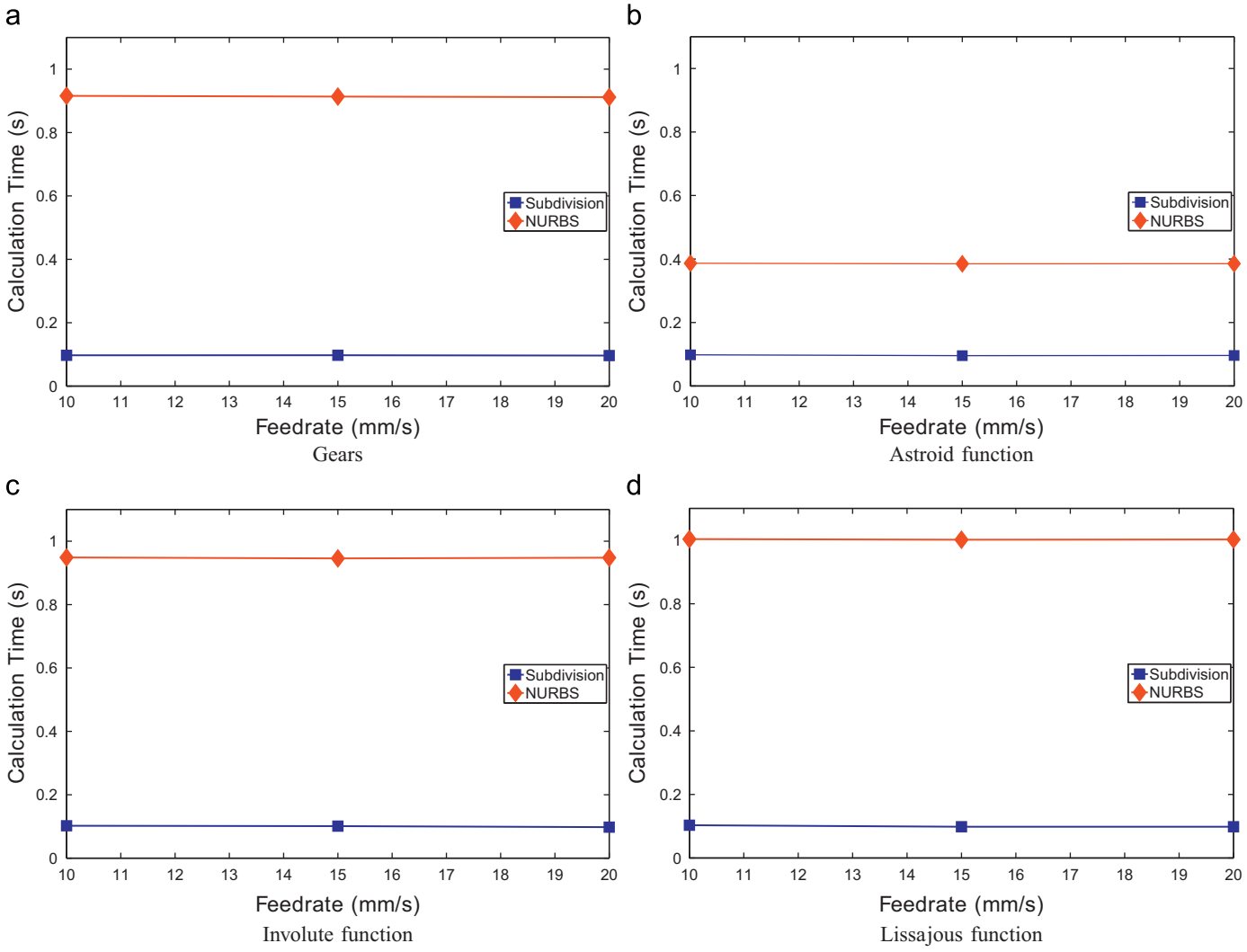

Fig. 6. Comparing NURBS and subdivision trajectory generation—feedrate variation.

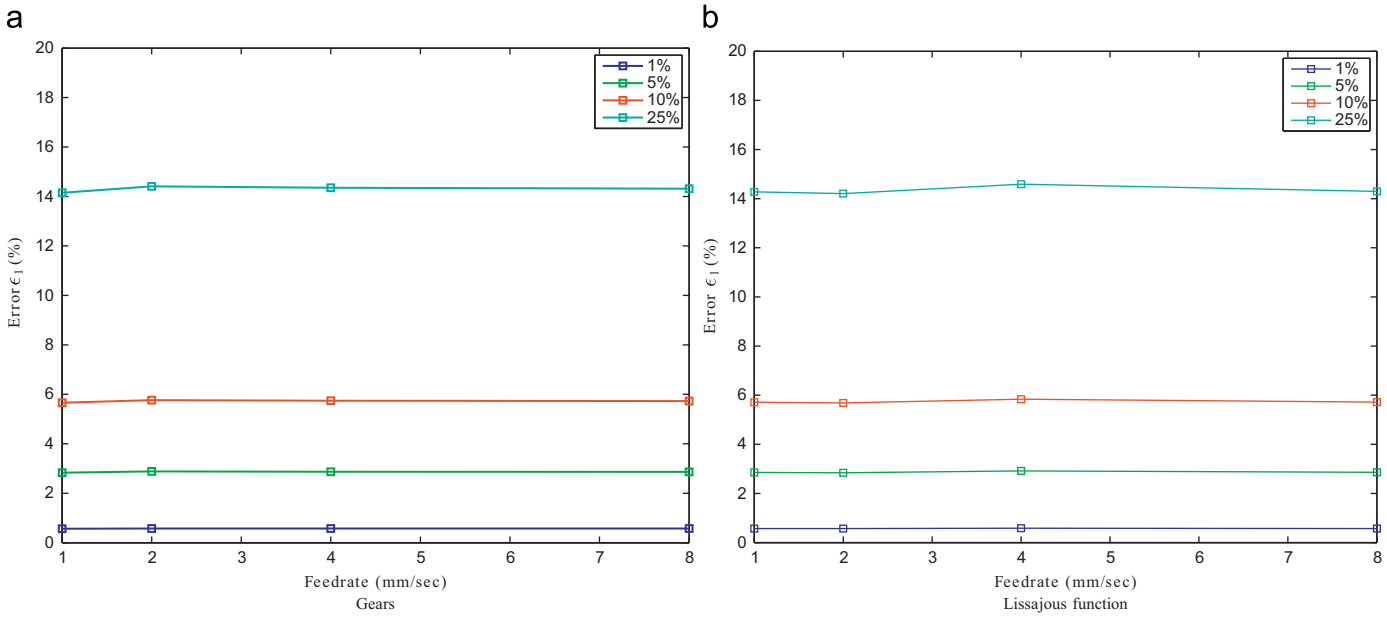

Fig. 7. Error 1 for different relative simulated machine trajectory-following errors.

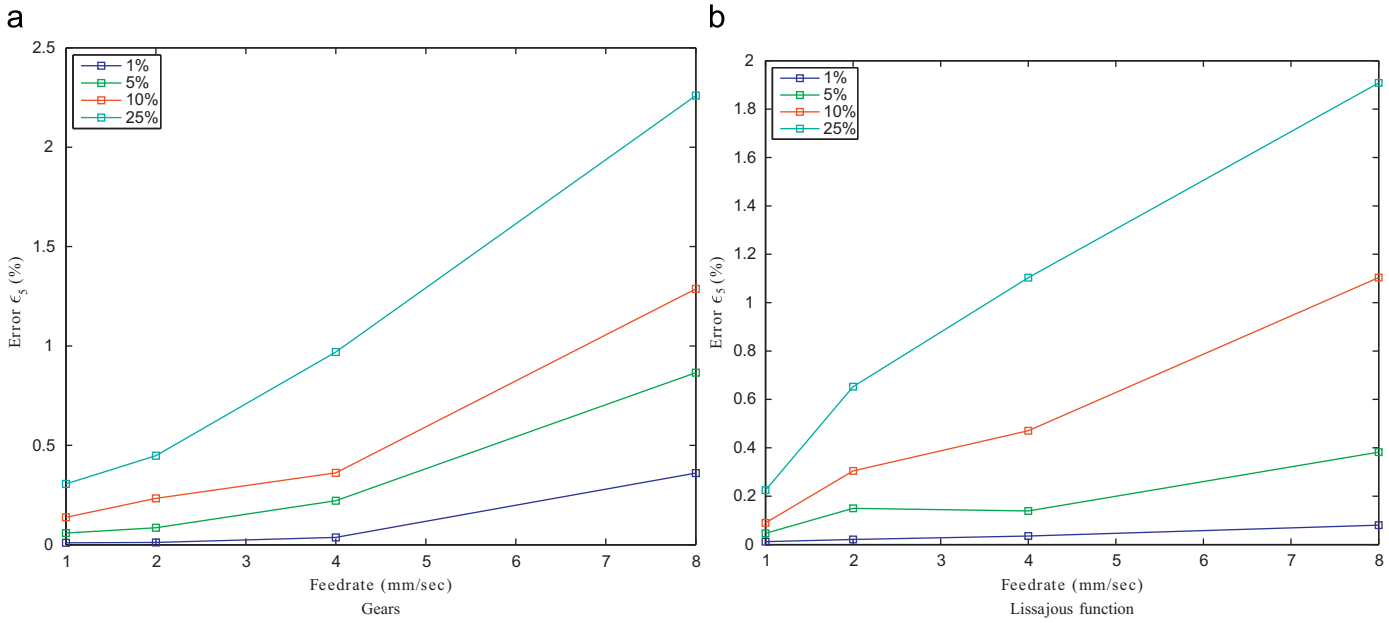

Fig. 8. Error 5 for different relative simulated machine trajectory-following errors.

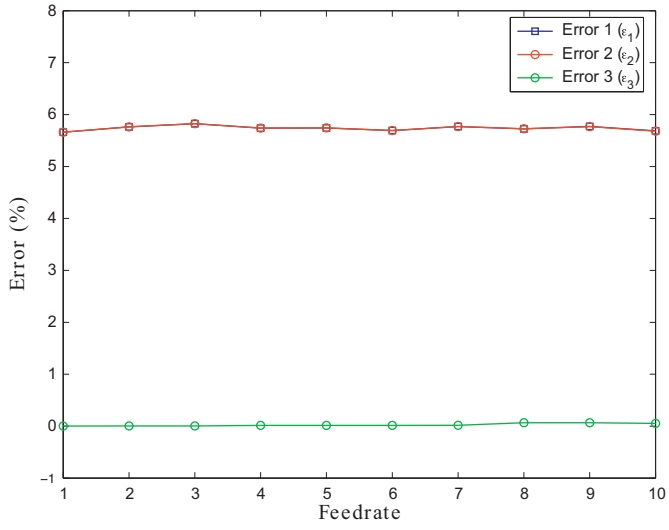

Fig. 9. Errors 1, 2, 3 for gears curve.

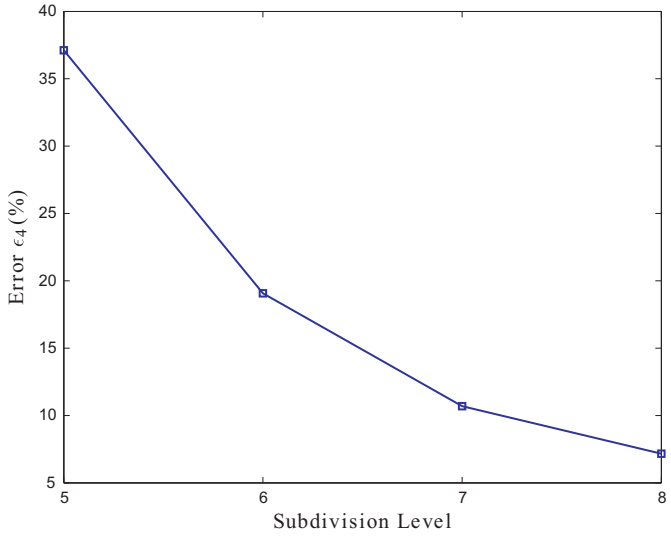

Fig. 10. Error 4 for gears.

significantly with increasing feedrate, but changes very significantly with the simulated machine trajectory-following errors. The error metric is approximately half the simulated machine trajectory-following error; this can be explained by examining the method by which the error metric is calculated. Since the metric involves taking the root-mean-square, the signed errors are lost, and as the error is evenly distributed about the goal position, the metric shows approximately half the random error value. For example, for the case of a 10% machine error, an ε_1 value of about 5% can be expected. Fig. 8 is a plot of ε_5 for the same cases. This plot indicates that this error metric increases with simulated machine trajectory-following error as well as with the feedrate. As the feedrate increases, the chord length increases, and this leads to increasing mismatch between the actual and commanded positions.

For a simulated machine trajectory-following error size of 10% for the gears curve, error metrics ε_1, ε_2, ε_3 are shown against feedrate in Fig. 9 (with a fixed sample time of 5 ms). We can see that ε_3 is close to negligible for all the cases, and that ε_1 and ε_2 are almost identical. This indicates that the trajectory generation was accurate, as the step length is equal to the trajectory chord length in each case (this was ensured by using the interpolating function during generation). This also implies that the primary source of error in this system was from a mismatch between the actual and commanded lengths.

Error metric ε_4 can be used to indicate the accuracy of the subdivision. Fig. 10 is a plot of this error metric against different levels of subdivision used for the analysis, beginning at the minimum level required for accurate trajectory generation (with all other variables fixed–feedrate: 1 mm/sec, sampling time: 5 ms). As the subdivision level increased, this error metric decreased, indicating that the curve was being interpolated more accurately, and that errors like gouging were also decreasing.

The difference between the arc length and the chord length can also be studied on a step-by-step basis. Fig. 11 shows this error for each trajectory step for the gears curve at a subdivision level of 5. We can clearly see spikes in the curve corresponding to

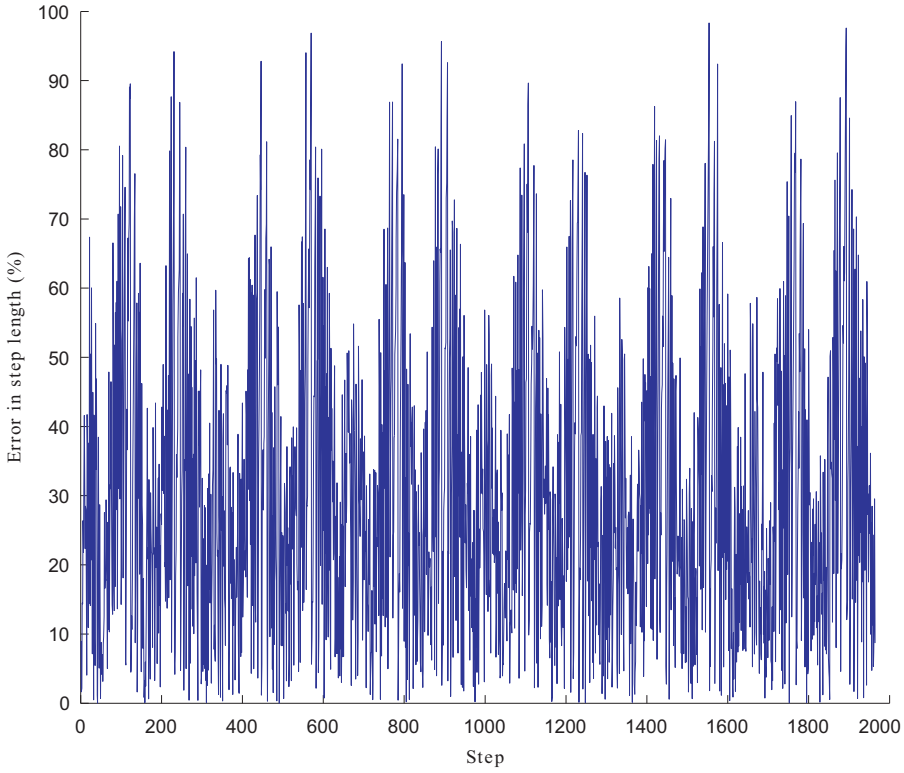

Fig. 11. Error 4 for gears at each step.

the six gear teeth; each tooth has two spikes, corresponding to the large error in the high curvature regions found at the gear corners.

6. Discussion

Subdivision curves are a very capable representation for generating machine tool trajectories. Since they are naturally discretized, the additional processing required to numerically sample them at high-speeds is minimal. Error metrics are also easy to calculate as we are operating on a discrete set of points. The potential for ambiguity is reduced when using subdivision curves as the curve is explicitly defined at the vertices, unlike in the case of NURBS curves where the curve must be evaluated. In addition to providing an off-line measure of interpolation suitability, the error metrics can also be applied in real-time error monitoring during machining. For example, the difference between arc length and chord error as shown in Fig. 11 could be monitored during machining, and employed as a way of predicting part quality before manufacturing is completed. These metrics can also be used as feedback to the motion control loop to improve the following error in the machine tool.

Using subdivision curves for trajectory generation has the potential of increasing the productivity of micromilling through increased feedrates. The amount of chord error in a micromilling operation is dependent on the chord length traversed by the spindle during one sampling iteration. The chord length is equal to the product of feedrate and sampling time. Therefore, a decrease in sampling time will allow an increase in feedrate without an increase in chord error. Sampling time is limited by trajectory generation time. Since trajectory generation by subdivision curves is faster compared to the NURBS method, this allows for increased feedrates during micromilling.

It has to be noted that in the subdivision curves method, the time to generate a trajectory is dependent *only* on the chord length, which is a function of both the sampling time and the feedrate. The NURBS method does not exhibit the same dependency. This means that the algorithm behaves similarly at cases of high feedrate and high sampling rate as well in cases of low feedrate and low sampling rate. In terms of machining performance, however, these two cases are not comparable. The performance of the machine tool (especially with respect to following errors) is much better in the latter case, but the algorithm does not necessarily reveal this. Additionally, because of this effect, the amount of increased feedrate available by using the subdivision method depends upon the chord length: a longer chord length will result in greater benefit, while a shorter chord length will result in a smaller benefit.

It is important to note that subdivision curves have to be carefully applied in machining applications; if used improperly this method can lead to significant errors. The subdivision level used in the trajectory generation should be determined such that discretization errors are decreased. When using uniform subdivision, the level should ensure that the largest feature is smaller than the chord length. When using adaptive subdivision schemes, the level can be locally defined based on the size of the local feature relative to the chord length. Adaptive schemes can potentially further decrease the run-time of the algorithm, as the computational complexity of the curve is determined based on the local feature size and curvature.

7. Conclusions

In this study, we introduced the method of subdivision curves as an interpolation method with reduced trajectory generation time when compared to the NURBS method. The reduced trajectory generation time allows for increased feedrate or reduced chord error in high-speed, high-precision micromilling. Several error metrics were developed to quantify the quality of

the subdivision interpolation. A set of test curves were used to compare the trajectory generation times for the subdivision curves method compared to the NURBS method. Subdivision curves were shown to reduce trajectory generation calculation time by as much as 10 times, relative to NURBS.

This research underscores the requirements advanced manufacturing processes such as high-speed micromilling place on design representations. It is no longer adequate for the design tools used in manufacturing processes to accurately capture the shape. They also need to be robust enough to allow for fast computational analysis for application in real-time environments. Subdivision curves have the potential for application in this domain, as they offer bounded guarantees on the geometric fidelity of curves and surfaces, while making available new computational methods to improve the performance of applications that operate on the geometric property of the curves. Their utilization in traditional manufacturing domains have been limited, and this work has demonstrated that they offer rich potential in improving the accuracy and performance of manufacturing processes.

Acknowledgments

The work presented in this paper has been an outcome of a collaborative research effort of the Laboratory for Manufacturing and Sustainability (LMAS) at the University of California, Berkeley, and the Laboratory for Mesoscale Manufacturing & Energy Systems Optimization (MESOLab) at the Georgia Institute of Technology. The authors gratefully acknowledge the support of the respective sponsors and industrial affiliates.

References

[1] M.Y. Cheng, M.C. Tsai, J.C. Kuo, Real-time NURBS command generators for CNC servo controllers, International Journal of Machine Tools and Manufacture 42 (7) (2002) 801–813.
[2] C.C. Lo, Real-time generation and control of cutter path for 5-axis CNC machining, International Journal of Machine Tools and Manufacture 39 (3) (1999) 471–488.
[3] J. Park, S. Nam, M. Yang, Development of a real-time trajectory generator for NURBS interpolation based on the two-stage interpolation method, The International Journal of Advanced Manufacturing Technology 26 (4) (2005) 359–365.
[4] H.T. Yau, M.J. Kuo, NURBS machining and feed rate adjustment for high-speed cutting of complex sculptured surfaces, International Journal of Production Research 39 (1) (2001) 21–41.
[5] D.Y. Yang, D.G. Ahn, C.H. Lee, C.H. Park, T.J. Kim, Integration of CAD/CAM/CAE/RP for the development of metal forming process, Journal of Materials Processing Technology 125 (2002) 26–34.
[6] I. Zeid, Geometric modeling. Mastering CAD/CAM, 2005.
[7] S.H. Nam, M.Y. Yang, A study on a generalized parametric interpolator with real-time jerk-limited acceleration, Computer-Aided Design 36 (1) (2004) 27–36.
[8] K. Erkorkmaz, Y. Altintas, Quintic spline interpolation with minimal feed fluctuation, Journal of Manufacturing Science and Engineering 127 (2005) 338–349.
[9] C. Lartigue, F. Thiebaut, T. Maekawa, CNC tool path in terms of B-spline curves, Computer-Aided Design 33 (4) (2001) 307–319.
[10] Y. Sun, J. Wang, D. Guo, Guide curve based interpolation scheme of parametric curves for precision CNC machining, International Journal of Machine Tools and Manufacture 46 (3–4) (2006) 235–242.
[11] J.R. Mayor, A.A. Sodemann, Intelligent tool-path segmentation for improved stability and reduced machining time in micromilling, Journal of Manufacturing Science and Engineering 130 (2008) 031121.
[12] L.A. Piegl, W. Tiller, The NURBS Book, Springer, Berlin, 1997.
[13] C.W. Cheng, M.C. Tsai, Real-time variable feed rate NURBS curve interpolator for CNC machining, The International Journal of Advanced Manufacturing Technology 23 (11) (2004) 865–873.
[14] E. Catmull, J. Clark, Recursively generated b-spline surface on arbitrary topological meshes, Computer-Aided Design 10 (6) (1978) 350–355.
[15] C.T. Loop, Smooth subdivision surfaces based on triangles, Master's thesis, Department of Mathematics, University of Utah, 1987.
[16] D. Zorin, P. Schröder, W. Sweldens, Interpolating subdivision for meshes with arbitrary topology, in: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, ACM, New York, NY, USA, 1996, pp. 189–192.
[17] D. Zorin, P. Schroder, Subdivision for modeling and animation, SIGGRAPH 2000 Course Notes, December 2000.
[18] L.P. Kobbelt, A subdivision scheme for smooth interpolation of quad-mesh data, Eurographics 98, June 1998.
[19] D. Doo, M. Sabin, Behaviour of recursive division surfaces near extraordinary points, Computer-Aided Design 10 (6) (1978) 356–360.
[20] L.P. Kobbelt, Root-three subdivision, in: SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000.
[21] A.S. Cavaretta, C.A. Michelli, The design of curves and surfaces by subdivision algorithms in: Mathematical Methods in Computer Aided Geometric Design, Academic Press Professional, Inc., San Diego, CA, USA, 1989, pp. 115–153 ISBN:0-12-460515-X.
[22] T. DeRose, M. Kass, T. Truong, Subdivision surfaces in character animation, in: SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, 1998, pp. 85–94.