2-1-1993

Gouge Detection Algorithms for Sculptured Surface NC Generation

James H. Oliver  
*Iowa State University*, oliver@iastate.edu

D. A. Wysocki  
*Chrysler Corporation*

E. D. Goodman  
*Michigan State University*

Follow this and additional works at: [http://lib.dr.iastate.edu/me_pubs](http://lib.dr.iastate.edu/me_pubs)

The complete bibliographic information for this item can be found at [http://lib.dr.iastate.edu/me_pubs/54](http://lib.dr.iastate.edu/me_pubs/54). For information on how to cite this item, please visit [http://lib.dr.iastate.edu/howtocite.html](http://lib.dr.iastate.edu/howtocite.html).
Gouge Detection Algorithms for Sculptured Surface NC Generation

Two algorithms are presented to detect and correct milling errors which can occur in commonly applied three-axis NC generation procedures for sculptured surface parts. The first method exploits surface normals and the geometry of the tool to accurately characterize the chordal deviation resulting from an actual tool motion. The second technique considers the global nature of an entire tool pass to detect gouging in areas of high curvature variation. This method is capable of detecting gouges in both locally concave and convex regions of curvature. The algorithms are designed in a modular fashion and can be incorporated as part of the general generative procedure. Results of such an implementation are presented which demonstrate the benefits of the gouge detection algorithms.

Introduction

Sculptured surface parts such as stamping dies and injection molds are often produced with three-axis NC end-mills using spherical (ball-nose) cutting tools. As shown in Fig. 1, this tool geometry allows relatively straightforward three-axis path generation since tool centers can be located along surface normal vectors. To reduce tool wear and milling time, NC generation algorithms for parametric sculptured surfaces are generally driven by the goal of minimizing the total number of tool motions while maintaining a specified overall milling tolerance.

Parametric sculptured surfaces can be defined in many different mathematical forms (see, for example, Faux and Pratt, 1979; Böhm et al., 1984; and Farin, 1988). Probably the most popular form is the nonuniform rational B-spline (NURB) (Piegel and Tiller, 1987); although systems based on higher order Bezier polynomials are also quite prevalent. Regardless of the underlying mathematical basis, a sculptured surface can be generally characterized by a bivariate parametric vector function \( p(u, v) \) which represents the spatial coordinates of surface point,

\[
p(u, v) = \left[ u^n u^{n-1} \ldots u \right] MG^T[u^m v^{m-1} \ldots v] \]

where,

\[
u, v \text{ are real valued parameters, typically } u \in [0, 1], v \in [0, J] \quad (I \text{ and } J \text{ are positive integers}),
\]

\[
n, m \text{ denote the degree of the polynomial interpolating functions},
\]

\[
M \text{ is a matrix of constants characterizing the interpolating basis (blending) functions, and}
\]

\[
G \text{ is a matrix of geometric coefficients to be interpolated,}
\]

typically control point coordinates or boundary curve characteristics.

By holding one parameter, say for example, constant, the surface definition \( p(u, v) \) reduces to a three-dimensional space curve dependent only on \( u \). One straightforward approach to generating NC milling programs for sculptured surface parts is to generate points along constant parameter curves, at uniform parametric steps, offset from the surface by the spherical tool radius, \( R \). The direction of the offset is based on the surface normal,

\[
n(u, v) = \frac{p^*(u, v) \times p^*(u, v)}{|p^*(u, v) \times p^*(u, v)|}
\]

where, \( p^*(u, v) \) and \( p^*(u, v) \) are parametric derivatives (surface tangents), e.g.,

\[
p^*(u, v) = \frac{dp(u, v)}{du} = \left[ nu^{n-1} \ldots u^{n-2} \ldots u \right] MG^T[u^{m-1} \ldots v]\]

and similarly for \( p^*(u, v) \). Thus, a tool center is specified by,

\[
t(u, v) = p(u, v) + Rn(u, v)
\]

Fig. 1 Spherical end tool position for milling a sculptured surface
An alternative approach is to first generate a surface offset from the design surface, then calculate tool locations on constant parameter lines on the offset surface (Chen and Ravani, 1987). However, exact offset surfaces often exhibit degeneracies which make them unsuitable for practical application. For example, if the principal radius of curvature of the generating surface is locally smaller than the offset distance, the resulting offset surface will be self-intersecting. In addition, if the surface to be milled has discontinuous tangents, the offset surface may have “holes.” Approximate offset techniques have been developed (Farouki, 1986) to address these problems, but they introduce a computationally expensive preprocessing step into the generation procedure.

In actual practice, once tool center locations are generated, the coordinates of these points are written to an intermediate “cutter location” file (CL-file). The CL-file is then post-processed for a specific machine tool to incorporate the interpolation method to be used between CL-points, the tool rotational and translational speeds, and the cutting lubrication requirements (Chang and Melkanoff, 1989). Since CL-points are often interpolated linearly, the fewer the number of points, the rougher the final machined surface. However, large CL-files, while potentially more accurate dimensionally, result in unacceptably large CL-file size. To address this trade-off between milling accuracy and CL-file size, the CAM-I (1978) processor, CASPA (Computer Aided Sculptured Pre-APT) exploits algebraic properties of the surface to generate a non-uniformly spaced parametric distribution of points which reduces the number of CL-points while maintaining milling accuracy.

Loney and Ozsoy (1987) present a preprocessor for the CAM-I package which shifts the basis of the nonuniform point spacing from the abstract algebraic properties of the surface to the more intuitive geometric attributes of chordal deviation and cusp height. With this method, these geometric constraints are met sequentially. First, in a specified primary parametric direction, the number of linear tool motions for a given tool path is minimized based on a specified tolerance between the actual designed surface and a chordal approximation to it, as shown in Fig. 2. Second, based on a maximum allowable cusp height (a measure of material remaining between adjacent tool passes) a maximum “step-over” distance is calculated in the secondary parametric direction. This technique represents a significant improvement in interactivity and user friendliness over the standard CASPA system.

In this paper we represent an algorithm which increases the robustness of sculptured surface NC generation procedures in two ways. First, a more rigorous check of chordal deviation is developed to improve accuracy during generation of a single pass of the tool (i.e., in a primary direction constant parameter line). Secondly, given an entire tool pass, a method is presented to detect and correct gouges which can result when the general generation procedure, based on local surface properties, is applied to surfaces with high curvature variation. These methods can be employed as part of the generation procedure for increased accuracy and reduced verification and milling time.

Local Gouge Detection

Figure 3 depicts three points which lie on a constant parameter curve of a sculptured surface. If the curve is to be approximated by a line segment (chord) from \( p_1 \) to \( p_2 \), the chordal deviation is defined as the maximum distance from the actual curve to the chord. This deviation can be calculated by several methods.

Probably the simplest method is to approximate the deviation by assuming that the point on the curve which is furthest from the chord is defined parametrically by half of the total parametric variation (Wysocki, 1987). As depicted in Fig. 3, if points \( p_1 \) and \( p_2 \) are defined by parameters \( u_1 \) and \( u_2 \), respectively, then \( p_2 \) can be approximated by the parameter \( u_2' = (u_1 + u_2)/2 \). Calculation of the distance, \( H_1 \), from \( p_1 \) to the line segment formed by \( p_1 \) and \( p_2 \) is trivial. This technique is generally sufficient for surfaces with uniform parametric distribution, i.e., those in which parameter variations proportional to arc length variation. The method is simple to implement and computationally efficient, but only approximates the actual chordal deviation. If the underlying surface definition is characterized by a nonuniform parametric distribution, this approximate method could yield inaccurate results. Loney and Ozsoy (1987) address this problem with a numerical method to solve for the parameter value which yields the largest deviation from the chord. This method is more robust but incurs a significantly higher computational cost. (See also, Loney, 1983.)

Either of these chordal deviation methods can be employed with heuristic search methods to find the minimum number of sequential tool path points (i.e., those with maximum parametric variation) while satisfying the chordal tolerance condition. For example, Loney and Ozsoy use a curve subdivision technique, while Wysocki (1987) employs a cast-and-correct method based on a binary search. As acceptable surface points are found, normal vectors are computed and tool centers are located at a distance equal to the tool radius along the normals.

However, both of these simple chordal deviation methods suffer limited accuracy and can produce unacceptable gouging because they are based solely on surface points and do not consider the normal and the geometry of the milling tool. Referring again to Fig. 3, although the tool sphere is tangent to the surface at both \( p_1 \) and \( p_2 \), the material removed is not, in general, determined by the line segment \( p_1 p_2 \). This is the case only if the surface normals at \( p_1 \) and \( p_2 \) are perpendicular to the line segment \( p_1 p_2 \). The following section describes a more accurate method for calculating chordal deviation.

**True Chordal Deviation** Since NC mills are often driven linearly between tool centers, it is necessary to consider the surface normal when calculating the true chordal deviation. Assuming that the nominal surface deviation \( (H_1 \) in Fig. 3) is calculated accurately as described above, the total true chordal deviation can be determined as follows. 

Figure 4 shows a segment of a constant parameter curve between two tool positions. As in Fig. 3, points \( p_1 \) and \( p_2 \)
If the surface is flat, so the chordal deviation is equal to the sum of the two values $H_1$ and $H_2$. For a locally concave surface, the total chordal deviation is equal to the greater of the two values $H_1$ and $H_2$. If $|L_1| = |L_2|$ the surface is flat, so the chordal approximation is exact. This procedure for calculation of true chordal deviation has been incorporated successfully in a proprietary CAD/CAM system. It offers improved overall accuracy at a relatively small additional computational cost as compared with a nominal chordal deviation method described above (Wysocki, 1987). A variation of the technique can also be employed to accurately compute an optimal step-over distance in the secondary parametric direction. This represents the parametric variation between adjacent passes of the tool and is constrained by a user specified tolerance on cusp height.

**Global Gouge Detection**

Calculation of true chordal deviation can improve the robustness of NC generation procedures, but since the general technique relies only on local properties of the surface, other interference problems can occur, especially when applying these techniques to surfaces with high curvature variation. Such problems often arise because points which are relatively distant parametrically can be very close (or even coincident) upon mapping to cartesian space. Since calculation of a tool pass along a constant parameter curve proceeds in monotonically increasing parametric steps, a global check of the entire tool pass must be made after it is calculated completely.

Figure 5 illustrates a typical problem in a locally concave region of the surface. If the local radius of curvature is any less than the tool radius, then the locus of CL-points generated via these procedures can produce a so-called “butterfly” pattern and the tool will gouge the design surface. One solution to this problem involves detecting and terminating points of the butterfly pattern and removal of the CL-points which fall between them. The tool path corrected via this method is shown in Fig. 6. Although gouging is corrected, the surface will not be completely machined with this tool path and geometry so the part must be milled further with a smaller diameter tool.

A related problem occurs in locally convex regions, as indicated in Fig. 7. In this case, the typical generation algorithm can fail to produce a sufficient number of CL-points to avoid gouging a convex or outside corner of the designed part. The solution requires insertion of a suitable interior point between the two offending tool locations.

Gouge detection is accomplished by comparing each individual tool motion to all others along the tool pass. Each comparison involves two distinct tool motions and thus four CL-points. Each pair is checked for gouging in both the concave and convex sense. If either condition holds, the appropriate corrective action is taken, otherwise the next pair is checked. The algorithm proceeds by comparing the first tool motion to the last, second to the last, and so on until the first motion is compared to the second. Then the second tool motion is compared to the remaining motions in a similar fashion. The method continues until all possible combinations have been considered. Denoting the number of CL-points in the tool pass as $N$, and the points themselves as an array, $p(1)$ through $p(N)$, the algorithm for gouge detection is given below in pseudo-code.
The points \( p''_1 \) and \( p''_2 \) are the CL-points which define one tool motion, while \( p''_3 \) and \( p''_4 \) described another one. The corresponding surface points are denoted \( p_1, p_2, p_3 \) and \( p_4 \). To determine if a butterfly occurs between the pair, the distance between \( p_2 \) and \( p_3 \), and between \( p_1 \) and \( p_4 \) is calculated. If either of these dimensions is less than the tool radius, then a butterfly exists and gouging can occur. (Of course, to reduce the computational burden, the square of the distance is typically calculated and compared.)

Since the surface parametrization is not generally proportional to the linear tool path, the calculation of the surface parameter values at the butterfly initiation and termination points must be solved in an interactive fashion. First, define the vectors \( S = (p_{4} - p_{3}) \) and \( T = (p_{4} - p_{3}) \) as the tool path vectors under consideration, with corresponding unit vectors \( s = S/|S| \) and \( t = T/|T| \). As depicted in Fig. 10 (and described in the Appendix) the distances \( A \) along \( s \) and \( B \) along \( t \) define end points of the minimum length vector between \( S \) and \( T \). (Note that the Appendix presents only the case in which \( S \) and \( T \) are noncoplanar; calculation of scalars \( A \) and \( B \) in the coplanar case is trivial.)

Next, ratios between \( A \) and \( |S| \) and between \( B \) and \( |T| \) can be used to estimate the surface parameters corresponding to the butterfly initiation and termination points. For example, suppose we are on a constant \( v \) parameter curve of the surface (i.e., only \( u \) can vary). If \( u_1 \) and \( u_2 \) are the parameter values corresponding to \( p_1 \) and \( p_2 \), the estimated parametric value for the butterfly initiation point is given by:

\[
 u'_2 = u_1 + (u_2 - u_1)A/|S| 
\]

A similar formula is used to estimate the parameter value of the butterfly termination point, \( u'_1 \).

New surface points \( p'_2 \) and \( p'_3 \) are then evaluated with the updated parameter values, and corresponding CL-points are generated in the usual manner. The entire procedure is then repeated with these new points until \( u'_2 \) and \( u'_1 \) converge to steady values, typically in no more than five iterations. The final updated CL-points \( p''_2 \) and \( p''_3 \) are then written to the CL-file to replace \( p_2 \) and \( p_3 \), and all CL-points with corresponding surface point parameter values between \( u'_2 \) and \( u'_1 \) are removed from the file.

**Convex Regions.** Global gouging of a convex region of a surface is depicted in Fig. 11. As in the concave case, detection of this condition involves comparison of two distinct tool motions. The necessary conditions which indicate a global gouge in a convex region are:

1. \( |p_2 p_3| > |p_1 p_4| \)
2. \( |p_1 p_4| < R \) (tool radius)
3. \( H_t > \) chordal tolerance limit

\[
 i = 1; \\
 \text{for } (i < N - 1) \\
 \text{if (concave_gouge(seg1, seg2))} \\
 \text{fix_concave(seg1, seg2, p);} \\
 \text{increment } i; \\
 \} \\
 \text{The following sections describe the details of the gouge detection and correction functions for both concave and convex surface regions. These functions operate only on two distinct tool motions at a time.}

**Concave Regions.** Figure 8 depicts a gouge condition in a concave region of a surface. The first task is to determine if the butterfly pattern is initiated and terminated within the input pair of tool motions. If so, the parameter values of the surface points whose normals represent the butterfly initiation and termination must be determined. The final position \( p_3 \) in Fig. 8) of the first tool motion of the pair is replaced by the point corresponding to butterfly initiation and the initial position of the second tool motion \( p_4 \) in Fig. 8) is replaced with the butterfly termination point. Then the CL-points whose corresponding surface points lie parametrically between the butterfly initiation and termination points can be identified and deleted.

The geometry necessary for this check is shown in Fig. 9. The points \( p_1 \) and \( p_2 \) are the CL-points which define one tool motion, while \( p_3 \) and \( p_4 \) described another one. The corresponding surface points are denoted \( p_1, p_2, p_3 \) and \( p_4 \). To determine if a butterfly occurs between the pair, the distance between \( p_2 \) and \( p_3 \), and between \( p_1 \) and \( p_4 \) is calculated. If either of these dimensions is less than the tool radius, then a butterfly exists and gouging can occur. (Of course, to reduce the computational burden, the square of the distance is typically calculated and compared.)

Since the surface parametrization is not generally proportional to the linear tool path, the calculation of the surface parameter values at the butterfly initiation and termination points must be solved in an interactive fashion. First, define the vectors \( S = (p_{4} - p_{3}) \) and \( T = (p_{4} - p_{3}) \) as the tool path vectors under consideration, with corresponding unit vectors \( s = S/|S| \) and \( t = T/|T| \). As depicted in Fig. 10 (and described in the Appendix) the distances \( A \) along \( s \) and \( B \) along \( t \) define end points of the minimum length vector between \( S \) and \( T \). (Note that the Appendix presents only the case in which \( S \) and \( T \) are noncoplanar; calculation of scalars \( A \) and \( B \) in the coplanar case is trivial.)

Next, ratios between \( A \) and \( |S| \) and between \( B \) and \( |T| \) can be used to estimate the surface parameters corresponding to the butterfly initiation and termination points. For example, suppose we are on a constant \( v \) parameter curve of the surface (i.e., only \( u \) can vary). If \( u_1 \) and \( u_2 \) are the parameter values corresponding to \( p_1 \) and \( p_2 \), the estimated parametric value for the butterfly initiation point is given by:

\[
 u'_2 = u_1 + (u_2 - u_1)A/|S| 
\]

A similar formula is used to estimate the parameter value of the butterfly termination point, \( u'_1 \).

New surface points \( p'_2 \) and \( p'_3 \) are then evaluated with the updated parameter values, and corresponding CL-points are generated in the usual manner. The entire procedure is then repeated with these new points until \( u'_2 \) and \( u'_1 \) converge to steady values, typically in no more than five iterations. The final updated CL-points \( p''_2 \) and \( p''_3 \) are then written to the CL-file to replace \( p_2 \) and \( p_3 \), and all CL-points with corresponding surface point parameter values between \( u'_2 \) and \( u'_1 \) are removed from the file.

**Convex Regions.** Global gouging of a convex region of a surface is depicted in Fig. 11. As in the concave case, detection of this condition involves comparison of two distinct tool motions. The necessary conditions which indicate a global gouge in a convex region are:

1. \( |p_2 p_3| > |p_1 p_4| \)
2. \( |p_1 p_4| < R \) (tool radius)
3. \( H_t > \) chordal tolerance limit
Condition 1 indicates a convex region, while conditions 2 and 3 indicate that the heuristic generation procedure based on chordal deviation has failed. The heuristic search fails because no matter how small the parametric step size, the deviation remains larger than the tolerance limit, thus indicating an abrupt convex region or "corner." In condition 3, $H_2$ refers to the auxiliary chordal deviation described above. Since $p_2$ and $p_3$ are very close, the contribution of $H_1$ to the total chordal deviation is neglected.

If these conditions hold for a given pair of tool motions, the tool path is corrected by inserting a new CL-point between $p_{o2}$ and $p_{o3}$, as shown in Fig. 12. The location of this point is determined as follows. As in the concave case, vectors $s$ and $t$ are constructed from CL-points $p_{nl}$, $p_{o2}$, $p_{o3}$ and $p_{o4}$. The vector of minimum distance $Cq$ between $S$ and $T$ is calculated as shown in Fig. 10 (and described in the Appendix). The new CL-point can be located at a point one half the distance along $Cq$, or, more precisely:

$$p_{o5} = p_{o1} + As + Cq/2$$

The gouge is corrected by inserting this point into the CL-file between $p_{o2}$ and $p_{o3}$.

### Results

The gouge detection algorithms described above have also been successfully implemented in a proprietary CAD/CAM system. The following discussion presents an NC generation example which demonstrates the practical utility of the algorithms.

The ruled surface shown in Fig. 13 is intentionally designed to exercise the capabilities of the algorithms presented here. The surface is approximately ten inches long by seven inches wide. The two inch high "cusp" on one edge blends into a flat edge across the width of the surface. A spherical (ball-nosed) milling tool of radius 2.5 inches is to be used to mill the surface. To accentuate the effect of curvature changes, the primary parametric direction is designated along the length of the surface, across the cusp.

Figure 14 shows the results of the initial tool path generation based on true chordal deviation. Several butterflies are apparent indicating gouging in the concave regions, and the top of the cusp is apparently gouged due to abrupt convex change in curvature there. To improve the clarity of the illustration, only three tool passes are shown, the actual NC program contains many more tool passes. The results of the global gouge detection algorithm are shown in Fig. 15. Note that the butterflies have been eliminated and points have been inserted in the convex regions to avoid gouging there. The full corrected NC program is illustrated in Fig. 16. This was generated by specifying a cusp height of 0.002 inches, which resulted in nine tool passes.
Summary and Conclusions

Two techniques for gouge detection in three-axis sculptured surface NC generation have been presented. The first method exploits the local surface normal and the geometry of the tool to accurately characterize the chordal deviation resulting from an actual tool motion. The second technique considers the global nature of an entire tool pass to detect gouging in areas of high curvature variation, with distinct consideration of both concave and convex regions. These gouge detection and correction algorithms are computationally efficient and simple to implement as part of a general sculptured surface NC generation procedure for three-axis end mills. The initial implementation of these techniques has proven their feasibility in a production environment and demonstrated substantial error reduction when compared to generation techniques based on local surface geometry and traditional chordal deviation.

Though these procedures provide for improved NC generation capabilities, they deal only with the spherical end of the tool geometry. Since the sides of the tool may actually cut the part, or interfere with adjacent surfaces, NC programs created by these algorithms may still cause unexpected errors. To address this issue, the techniques presented here have been incorporated with a robust NC verification algorithm (Oliver and Goodman, 1990) to provide complete consideration of all milling geometry. This combined system will substantially improve manufacturing productivity by reducing the debugging and off-line verification time normally incurred in NC production processes.

Acknowledgments

The authors are grateful to Chrysler Corporation for support of this work.

References


Farin, I. D., and Pratt, M. J., 1979, Computational Geometry for Design and Manufacture, Ellis Horwood Ltd., West Sussex, U.K.


Appendix

Figure 10 illustrates the geometry necessary to describe the calculation of the minimum distance between two line segments. Vector S is located at point \( p_{n1} \) and vector T at \( p_{n3} \). Vector \( r \) represents the difference between points \( p_{n1} \) and \( p_{n3} \), and \( s \) and \( t \) are the unit vectors in the direction of S and T respectively, i.e.,

\[
r = (p_{n3} - p_{n1})
\]

\[
s = S/|S|
\]

\[
t = T/|T|
\]

The vector perpendicular to both \( s \) and \( t \) is given by their cross product, thus,

\[
q = s \times t / |s \times t|
\]

To calculate the minimum distance between S and T, the following vector loop equation can be written,

\[
As + Ct = r
\]

or, expanded in matrix form,

\[
\begin{bmatrix}
s_1 & t_1 & q_1 \\
s_2 & t_2 & q_2 \\
s_3 & t_3 & q_3
\end{bmatrix}
\begin{bmatrix}
A \\
B \\
C
\end{bmatrix}
\begin{bmatrix}
f_1 \\
f_2 \\
f_3
\end{bmatrix}
\]

Provided that S and T are not coplanar, this system of three equations in three unknowns can be solved for the constants \( A, B, \) and \( C \); where \( C \) represents the distance between S and T.