Chapter 7. Three Axis Tool Path Generation Methods

Optimal Tool Selection for Interference-free Sculptured Surface Machining

Zhonglin Han and Daniel Yang
CAD/CAM Lab., Dept. of Mechanical and Aerospace Engineering
University of California, Los Angeles, Los Angeles, CA 90095

Abstract: This paper presents a systematic tool-path generation methodology which incorporates interference detection and optimal tool selection for machining free-form surfaces on 3-axis CNC machines using ball-end cutters. In this method, the global and local interference is first detected and prevented, and then the optimal tools in terms of machining time are selected and tool paths are generated. A system of 5 algorithms is developed to determine the interference area. On the basis of these algorithms, the machining time of each available tool is estimated by considering tool size, scallop height, and accessible surface area. Finally, the combination of tools which possesses the minimum overall machining time is selected as the optimal tool sizes. Our case study has demonstrated the validity of the proposed methodology and algorithms.

1. INTRODUCTION

The tool-path generation of free-form surfaces is a difficult task and has been addressed by a number of researchers since the mid-80s [1-12, 15-20, 22]. Currently, a three-stage approach is adopted for the tool-path generation procedure (Figure 1). The first stage is to construct tool paths based on the given surface geometry and machining requirements. The second stage is to detect and correct potential tool interference problems in the paths from the first stage. The third stage is to perform the NC checking and generate machine-specific G-code commands [3].

The tool interference can be classified into two types: 1. *global interference* - the tool collides with the surface when moving on it as shown in Figure 2a; and 2. *local interference* - the local surface radius of curvature is smaller than tool radius as shown in Figure 2b and 2c. Here, we consider that the point P in Figure 2c has zero radius of curvature. The distinctive difference between the two types of interference is that local interference may be avoided by using a smaller tool, yet the global interference persists in many cases.

The detection and correction of tool interference is a tough problem. Several researchers attempt to solve the global interference problem by checking accessibility of surface points. Gaal et al. [5] use the concept of hidden-surface removal to check whether interference will happen. They impose a mesh on the design surface and test the surface normals on the grid points for interference. Tseng [22] presents a method for avoiding the tool interference problem of Bezier surfaces. His method is based on subdivision of the control polygon. Since the subdivided control polygon approximates the surface, an interference-free control polygon will correspond to an interference-free part surface. Lee [10] develops a two-stage strategy for determining the interference-free tool access direction. In the first stage, he takes advantage of the convex hull property of NURBS to find a quick but conservative range of feasible tool orientations. If the first stage fails, the second stage - a detailed feasibility check which uses exact surface normals, is employed.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35392-0_40

G. J. Olling et al. (eds.), *Machining Impossible Shapes*

For the local interference problem, Bobrow [1] presents an iterative algorithm to eliminate the interference at the transitions between surfaces. Choi et al. [4] present a robust method for local interference problem in 3-axis machining. They convert CC data points into triangular mesh. A two-stage interference check on the mesh points is performed by testing the distance between the cutter center and the mesh facet. In the first stage, interference detection and correction are performed on concave regions of the surface. In the second stage, new CL data are inserted to prevent interference on convex regions of the surface. Oliver et al. [15] notice that the locus of CL points forms a so-called "butterfly" pattern in the interference area. They present an algorithm for detecting the pattern and thereafter eliminating interference. Tang et al. [19] adopt an offset-surface approach. They offset the part surface to form an offset surface and then sweep the offset surface parallelly to form tool paths.

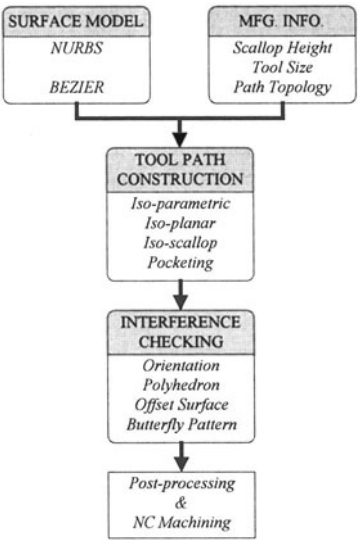


Figure 1. The schematic procedures in current tool-path generation method

In our opinion, the important role of tool size has been largely overlooked in the current tool-path generation scheme. In fact, the tool can greatly affect the total machining time and hence the cost. The larger the tool, the shorter the paths for the same scallop height allowance and the higher feed-rate we can apply to cutting. In machining practice, some surfaces are required to be machined in one cutting pass, therefore the largest tool without interference is the optimal tool. For surfaces which can be cut with multiple tools, a combination of tools which can result in the shortest machining time is the best.

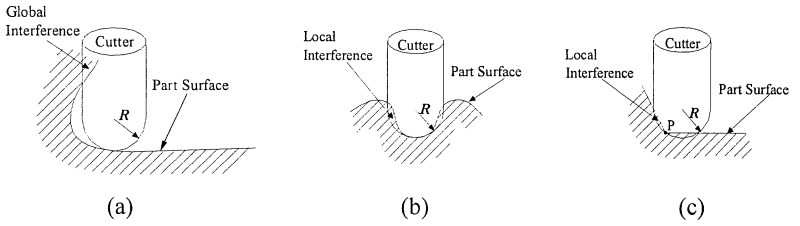


Figure 2. The global and local interference

This paper presents a systematic methodology which incorporates interference detection and optimal tool selection into the tool-path generation process. The resulting tool paths are for 3-axis CNC machines with ball-end cutters.

2. NEW TOOL-PATH GENERATION SCHEME

Aimed to improve machining efficiency by using the optimal tools and path topology, a new tool-path generation scheme for 3-axis machining is proposed and shown in Figure 3. Unlike the existing scheme, the first stage of the new scheme is to check the global and local interference. This rearrangement is valid because the interference problem is relevant only to surface and tool geometry in 3-axis machining [4]. The results from this stage are the accessible surface areas for each individual tool size.

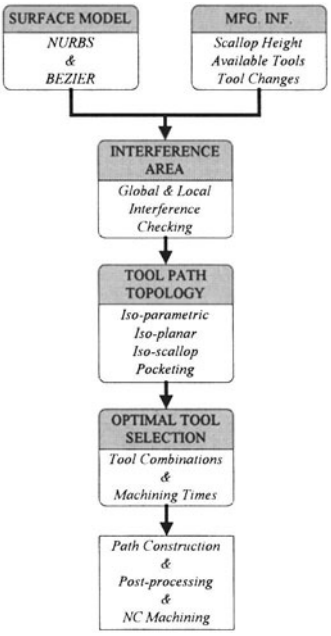


Figure 3. A new tool-path generation scheme with optimal tool-size selection

In the next stage, optimal tool-path topology is selected by considering the shape of interference area. Although automatic selection of optimal tool-path topology is possible [11], a user-interactive process is adopted for simplicity in this research.

The third stage is to select a tool or a combination of tools which can result in the shortest machining time. For a surface which has to be machined without tool change, the largest tool without causing interference problem is selected. Otherwise, the machining times of different combinations of tools are estimated and compared. The combination with the shortest time is selected as optimal. A model for estimating the machining time is here developed based on tool size, feed-rate, scallop height requirement, and area of targeted surface region in section 5.

In the last stage, the final interference-free tool paths are constructed based on the optimal tool sizes, the accessible surface areas, scallop height, and path topology.

The main advantage of this scheme is that it allows the selection of optimal tool size and topology, and hence, higher machining efficiency can be achieved. Methods for checking the global and local interference are the essential parts of this new tool-path generation scheme. In the next two sections, a systematic method and corresponding algorithms are presented to tackle the global and local interference.

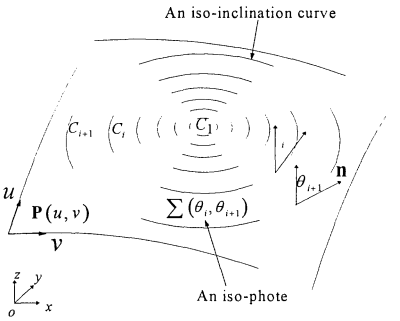


Figure 5. The iso-photes of a free-form surface

3. GLOBAL INTERFERENCE CHECKING

The global interference is caused by problematic surface geometry. It can happen in three scenarios: 1. there exists some part of surface where the angle between surface normal and tool axis exceeds 90^0 (see Figure 4a); 2. some part of surface is invisible if viewed from the direction of tool axis (see Figure 4b); and 3. the tool collides with surface boundary (see Figure 4c). We name these three scenarios Protrusion Interference (PI), Overlapping Interference (OI), and Boundary Collision Interference (BCI). Note that these three types of interference are not mutually exclusive. Some surfaces can have all of them. Furthermore, the OI is always accompanied by BCI, but not vice versa.

The global interference is solved in three phases according to the three scenarios. In the phase I, the PI is detected by applying the iso-photes and iso-inclination curves. In the phases II and III, the OI and BCI are identified by projecting, offsetting, and traversing the boundary curves. The final result is the global-interference-free areas on the surface.

3.1 Phase I: Protrusion Interference (PI) Checking

Algorithm 1 describes the procedure to detect the protrusion interference. It has three steps:

Step 1: Sorting critical iso-inclination curves Consider a part surface $\mathbf{P}(u,v)$ defined in parametric form as shown in Figure 5. In mathematics, the unit surface normal is

$$\mathbf{n}(u,v) = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T = \frac{\mathbf{P}^u \times \mathbf{P}^v}{\left|\mathbf{P}^u \times \mathbf{P}^v\right|} \tag{1}$$

where $\begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T$ are the three components of the surface normal $\mathbf{n}$. An **iso-phote** is defined as a region on the surface where the normal vector $\mathbf{n}(u,v)$ does not differ by more than a prescribed angle from a fixed reference vector $\mathbf{V}$. The boundary curve of the iso-phote is called **iso-inclination curve**. The angle between the normal vector of a surface point and the fixed reference vector is named inclination angle. Referring to Figure 5, $\sum(\theta_i,\theta_{i+1})$ is an

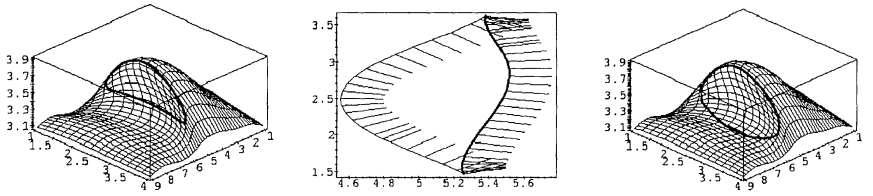
iso-phote on the free-form surface $\mathbf{P}(u,v)$ where the inclination angle is in the range of $[\theta_i, \theta_{i+1}]$. The curves C_i (i = 1, 2 ...) are iso-inclination curves. The iso-photes are originally used in computer graphics for their ability in shape feature detection.

Let the reference vector $\mathbf{V}$ = (0, 0, 1), then by simple vector analysis on the normal vector, the inclination angle at a point on surface is given as

$$\theta = \arccos\left(\frac{n_z}{\sqrt{n_x^2 + n_y^2 + n_z^2}}\right) \tag{2}$$

Also by vector analysis, the governing equation of the iso-inclination curves is obtained as

$$\begin{vmatrix} \frac{\partial y}{\partial u} & \frac{\partial z}{\partial u} \\ \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \end{vmatrix}^2 + \begin{vmatrix} \frac{\partial z}{\partial u} & \frac{\partial x}{\partial u} \\ \frac{\partial z}{\partial v} & \frac{\partial x}{\partial v} \end{vmatrix}^2 = \tan^2\theta \begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} \end{vmatrix}^2 \tag{3}$$



(a) (b) (c)

Figure 6. PI detection: (a) critical iso-inclination curve; (b) visible part of the critical iso-inclination curve (bold curve); (c) boundary of PI

Theoretically, the locus of the iso-inclination curve in the uv plane can be obtained by solving Eq. (3). However, numerically solving the equation is difficult when surface $\mathbf{P}(u,v)$ is modeled in complex parametric form. To overcome this problem, a grid-line based numerical algorithm is developed in Han and Yang [6]. In this research, we are interested in the 90^0 iso-inclination curves, called the "critical" iso-inclination curves here. In case that a critical iso-inclination curve is not a closed curve, the boundary curve of surface is traveled to close it. Figure 6a shows a free-form surface and its critical iso-inclination curve.

Step 2: Identifying the visible part of the critical iso-inclination curve This is done by projecting the closed curve onto XY plane and testing the direction of normal of each point on the curve. The normal vector is on XY plane since the inclination angle is 90^0. If the normal points to the outside of the closed curve, then the point on the curve is visible. The bold curve in Figure 6b is the visible part of the critical iso-inclination curve.

Step 3: Determining the PI boundary Obviously, the visible iso-inclination curve is one part of the PI boundary. The other part is the contact trace of tool sphere while the tool touches the visible iso-inclination curve. A method similar to Hwang [9] is used to calculate the contact points. Figures 6c shows the boundary of PI area for the example surface.

Algorithm 1: PI checking	
Input:	Surface $\mathbf{P}(u,v)$
	Cutter radius R
	Number of grid lines N_g

Output: The boundary curve of PI-free area
Algorithm:
Interference_PI ($\mathbf{P}(u,v)$, N_g)
Begin
A $N_g \times N_g$ grid is imposed on $\mathbf{P}(u,v)$.
IsoCurve $\Leftarrow$ calculated the critical iso-inclination curve.
if (IsoCurve is empty) exit;
else if (IsoCurve is not a closed curve)
IsoCurve $\Leftarrow$ IsoCurve + boundary curve;
VisibleCurve $\Leftarrow$ the visible part of IntCurve is determined by projecting IntCurve to XY plane and testing the surface normal.
ContactTrace $\Leftarrow$ the contact points between tool sphere and surface are calculated.
BoundCurve3D $\Leftarrow$ VisibleCurve + ContactTrace;
End

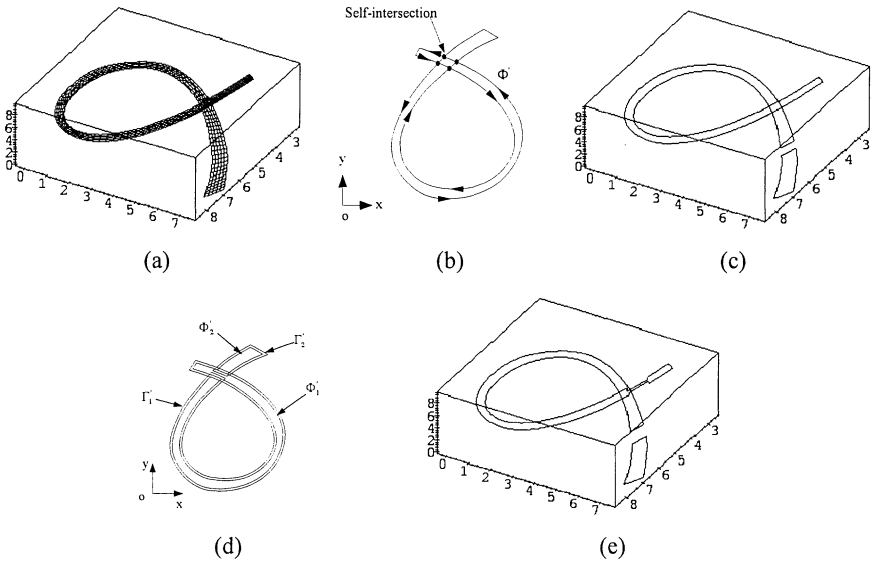


Figure 7. OI and BCI detection: (a) a 3D surface; (b) projection of the boundary curve; (c) OI-free areas; (d) offset curves; (e) the BCI-free areas.

3.2 Phase II: Overlapping Interference (OI) Checking

The OI is detected by traversing the projected surface boundary curve as stated in Algorithm 2. The algorithm is illustrated in following example.

Let the cutter radius be R and the surface be $\mathbf{P}(u,v)$ as shown in Figure 7a. The boundary curve Φ is extracted from surface model and stored in a bi-directional linked list. The curve Φ is projected onto XY plane and becomes a 2D closed curve Φ (Figure 7b). The next step is to find the self-intersection points on the curve Φ . These points are inserted into the linked list of boundary curve Φ with Z coordinate obtained by linear interpolation. If there is no such self-intersection point, then the whole surface is visible. Otherwise, the

visible area on surface is found by traversing the list Φ. The rule for traversing is described here. Refer to Figure 7b, the traversal starts from the point with the highest Z value. Points on list Φ are copied to a new list until an intersection point is met. In that case, the Z values of the current point and its pairing point are compared. If the current point has the higher Z value, the point are copied and the traversal continues with the next point on list Φ. Otherwise, the traversal jumps to the pairing point and moves on the reverse direction. The XY coordinates on list Φ are copied to the new list. The Z coordinate is calculated by mapping XY coordinates onto surface $\mathbf{P}(u,v)$. When the next intersection point is met, the traversal jumps again to the pairing point without condition. The procedure continues until the new list becomes a closed curve or no more untested point in list Φ. Based on the above procedure, the visible areas of surface $\mathbf{P}(u,v)$ are detected and shown in Figure 7c.

Algorithm 2: OI checking

Input: Surface $\mathbf{P}(u,v)$
Cutter radius R
Output: The boundary curve of OI-free area
Algorithm:
Interference_OI($\mathbf{P}(u,v)$, R)
Begin
$\Phi \Leftarrow$ boundary curve of $\mathbf{P}(u,v)$
$\Phi \Leftarrow \Phi$ is projected onto XY plane.
Calculate the self-intersection points of Φ.
$\Phi \Leftarrow \Phi$ is mapped back to 3D.
List Φ is re-ordered with the head point having the highest Z coordinate.
List Φ is traversed to form the boundary of OI-free area.
End

3.3 Phase III: Boundary Collision Interference (BCI) Checking

The BCI is avoided by checking the offset curve of surface boundary (Algorithm 3). Referring to Figure 7d, the boundary curves Φ_1 and Φ_2 are projected onto the XY plane and become 2D curves Φ_1 and Φ_2, respectively. The curves Γ_1 and Γ_2 are the offset curves of Φ_1 and Φ_2 with offset distance equals to the tool diameter 2R. The same Z values as their offset point are stored in Γ_1 and Γ_2, respectively. The 2R offset is to guarantee that we can have a safe estimation of the collision-free area by traversing these curves.

The lists Φ_1 and Γ_1 are traversed to construct the collision-free area. Referring to Figure 7d, the traversal starts with list Φ_1 from the point with the highest Z value. Points of list Φ_1 are copied to the new list until an intersection point between Φ_1 and Γ_1 is met. In that case, list Γ_1 is traversed in reverse direction and new calculated points are inserted to the new list. The XY coordinates of new point are the same as that of corresponding point on list Γ_1. The Z coordinate is calculated by projecting XY coordinates onto surface $\mathbf{P}(u,v)$. When the next intersection point is met, the traversal is switched back to list Φ_1. The

procedure continues until the new list becomes closed curve. The same procedure is applied to Φ_2 also. Figure 7e shows the final collision-free areas of the surface $\mathbf{P}(u,v)$.

Algorithm 3: BCI checking

Input: Surface $\mathbf{P}(u,v)$
Cutter radius R
Output: The boundary curve of BCI-free area
Algorithm:
Interference_BCI($\mathbf{P}(u,v)$, R)
Begin
$\Phi \Leftarrow$ boundary curve of $\mathbf{P}(u,v)$
$\Phi \Leftarrow \Phi$ is projected onto XY plane.
$\Gamma \Leftarrow \Phi$ is offset with value of 2R on XY plane.
Calculate the intersection points between Φ and Γ on XY plane. The points are inserted into the lists and marked.
$\Psi \Leftarrow$ Lists Φ and Γ are traversed, and points are copied to the new list Ψ according to the rule described in the paper.
$\Psi \Leftarrow \Psi$ is mapped back to 3D to form the final BCI-free boundary.
End

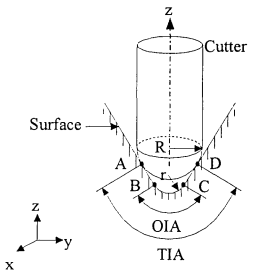


Figure 8. The TIA is normally larger than the OIA

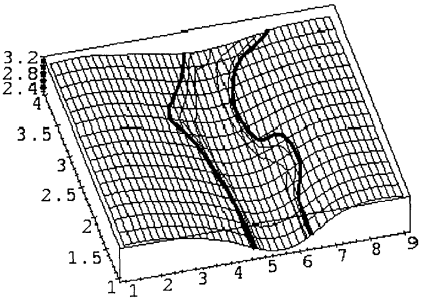


Figure 9. The OIAs of cutter radii 3mm, 4mm, and 5mm

4. LOCAL INTERFERENCE CHECKING

The local interference occurs when the normal curvature of the part surface is larger than that of the ball-end tool. As pointed by Choi and Jun [4], the above condition is both necessary and sufficient. In other words, the local interference can be completely identified by checking the surface normal curvature. In this research, the area where the local normal curvature is larger than that of tool is called original interference area (OIA). The total interference area (TIA) is usually larger than the OIA as illustrated in Figure 8. Therefore, a 2-phase approach is adopted in this research. In the phase I, the OIA is found by applying curvature theories from differential geometry. In the phase II, the TIA is found by locally searching the nearby area of OIA.

4.1 Phase I: Original Interference Area (OIA) Determination

Metrics of free-form surfaces For a parametric surface $\mathbf{P}(u,v)$, the normal curvature of a point on surface varies along the direction of measurement. However, there exist two principal directions along which the normal curvatures are of the maximum or minimum values [21]. In mathematics, given a point $\mathbf{P}(u,v)$ on surface, the maximum and minimum normal curvatures $\kappa_{1,2}$ are expressed as

$$\kappa_{1,2} = \mathrm{H} \pm \sqrt{\mathrm{H}^2 - \mathrm{K}} \tag{4}$$

$$\mathrm{H} = \frac{EN - 2FM + GL}{2\left(EG - F^2\right)} \tag{5}$$

$$\mathrm{K} = \frac{LN - M^2}{EG - F^2} \tag{6}$$

where H represents the mean curvature;
K, Gaussian curvature; and
$E = \mathbf{P}^u \bullet \mathbf{P}^u$; $F = \mathbf{P}^u \bullet \mathbf{P}^v$; $G = \mathbf{P}^v \bullet \mathbf{P}^v$;
$L = \mathbf{P}^{uu} \bullet \mathbf{n}$; $M = \mathbf{P}^{uv} \bullet \mathbf{n}$; $N = \mathbf{P}^{vv} \bullet \mathbf{n}$.

Detection of OIA A grid-line based algorithm is developed as depicted in Algorithm 4. It has two steps. Step 1 is to calculate the normal curvature at each grid point according to Eqs. (4)-(6). Recall that surface normal $\mathbf{n}$ and first order derivatives $\mathbf{P}^u$ and $\mathbf{P}^v$ at the grid point have already been calculated when PI is checked. Therefore, only the second derivatives $\mathbf{P}^{uu}$ and $\mathbf{P}^{uv}$ need to be calculated. Of the two principal curvatures, if both are positive or negative, the one having the larger absolute value is stored as the normal curvature. Otherwise, the negative one is stored. In step 2, the boundary points of OIA are identified and linked to form a closed 2D curve in uv space. The method here is similar to that used in generating the iso-photes. The boundary points are found by interpolating the proper grid interval; and these points are linked by sorting for neighborhood relation. Based on this algorithms, the OIAs of a B-Spline surface with 6×11 control points are calculated and shown in Figure 9.

Algorithm 4: The detection of OIA

Input: Surface $\mathbf{P}(u,v)$
Cutter radius R
Number of grid lines $\mathrm{N_g}$
Output: The boundary curve of OIA
Algorithm:
OIA($\mathbf{P}(u,v)$, R, $\mathrm{N_g}$)
Begin
A $\mathrm{N_g} \times \mathrm{N_g}$ grid is imposed on $\mathbf{P}(u,v)$.
Surface metrics, i.e. *E, F, G, L, M, N, K, H,* and $\kappa_{1,2}$, are calculated at the grid points according to Eqs. 4-6.
$R_{1,2} \Leftarrow 1/\kappa_{1,2}$;
if ($R_1 \cdot R_2 \geq 0$)
The one between R_1 and R_2 which has smaller absolute value is stored in an array $\mathrm{R[N_g][N_g]}$.

else

The one between R_1 and R_2 which has negative value is stored in the array $R[N_g][N_g]$.

OIAPoints $\Leftarrow$ the points with radius of curvature equals to R are calculated by interpolating array $R[N_g][N_g]$.

OIABoundary $\Leftarrow$ the OIAPoints are sorted and linked according to the neighborhood relation to form the OIA boundary.

End

4.2 Phase II: Total Interference Area (TIA) Determination

Referring to Figure 9, the boundary of TIA is larger than that of OIA by a distance up to the tool diameter 2R when projected onto XY plane. Therefore, the TIA can be found by locally checking the distance between tool center and surface around the OIA.

Algorithm 5 outlines the checking procedure. In the algorithm, a local grid is first imposed around the OIA (Figure 10). The boundary of this local grid is determined by ensuring its distance to the points of OIA list is larger than 2R. Then, the grid point is offset along the surface normal by an amount of tool radius. The distances between the offset point and all grid points are calculated. The smallest distance is stored. Figure 11a shows the result from this step. Next, the boundary points of TIA are identified according to the distance. At last, the boundary points are linked by sorting the neighborhood relation among the points. Figure 11b shows the TIAs for cutter radii 3mm, 4mm, and 5mm.

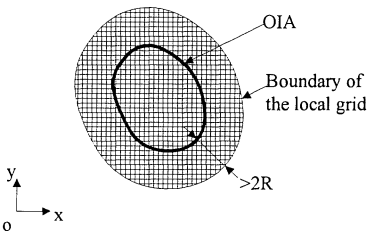


Figure 10. A local grid is imposed around the OIA

Algorithm 5: The TIA checking

Input: Surface $\mathbf{P}(u,v)$
The boundary curve of OIA - OIAList
Cutter radius R
Parametric increments Δu and Δv

Output: The boundary curve of TIA

Algorithm:

TIA($\mathbf{P}(u,v)$,OIALists, R, Δu , Δv)

Begin

A local grid is imposed around OIAList with Δu and Δv .

for (each grid point)

Grid point is offset by R along the normal direction.

The distances between the offset point and all other grid points are calculated, and the smallest distance is stored.

TIApoints ⇐ the stored center-surface distances is tested to find the TIA boundary points.
TIAlist ⇐ the TIApoints are sorted to form the TIA boundary.

End

5 OPTIMAL TOOL SELECTION

In the previous two sections, methods for detecting the global and local interference have been developed. On the basis of these methods, another procedure for selecting the optimal tool size(s) in machining part surface with/without tool change is presented in this section.

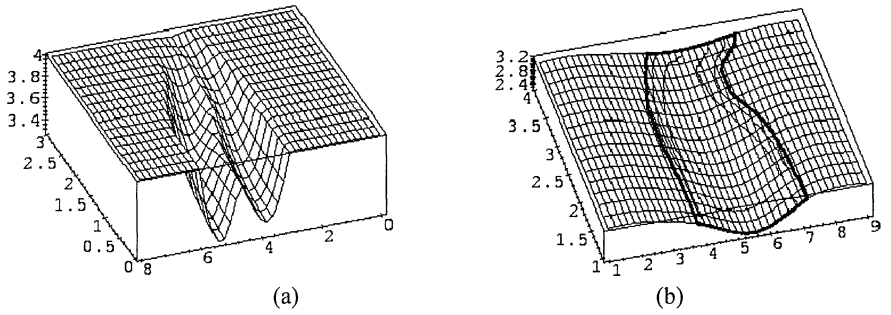
5.1 Goal of Optimization

For milling processes, the optimal combination of tool sizes is the one which results in the minimum machining time. Thus, if the number of tool change is N_{tc}, the time required for each tool change is t_{tc}, the radii of cutters are R_i (i=1, ..., N_{tc}), and the corresponding machining times are t_i, the goal of optimization G is

$$G = \min\left(\sum_{i=1}^{N_{tc}}(t_i + t_{tc})\right) \tag{7}$$

A model of machining time is established here. Suppose that the length of tool path is L_i and the feed-rate is f_i, the machining time t_i will be

$$t_i = \frac{L_i}{f_i} \tag{8}$$



(a) (b)

Figure 11 TIA detection: (a) center-surface distance; (b) TIAs of cutter radii 3mm, 4mm, and 5mm.

The length of tool path, in turn, depends on several other factors, i.e. the area of machining A_i, cutter radius R_i, scallop height h, size of path interval g_i, and path topology. The effect of path topology on the resulting length of tool path is complicated and beyond the scope of this research. Therefore, only the zigzag paths are considered here (see Figure 12). Let S_{io} and S_{ii} be the lengths of the outer and inner boundaries of A_i, then the length of tool path is estimated as

$$L_i = \frac{A_i}{g_i} + \frac{S_{io} + S_{ii}}{2} \tag{9}$$

The path interval can be calculated by approximating surface curve with straight line [18], thus

$$g_i = 2\sqrt{R_i^2 - (R_i - h)^2} = 2\sqrt{2R_i h - h^2} \tag{10}$$

Substituting Eqs. (9) and (10) into Eq. (8), we have the simplified model of machining time as

$$t_i = \frac{A_i}{2f_i\sqrt{2R_i h - h^2}} + \frac{S_{io} + S_{ii}}{2f_i} \tag{11}$$

Finally, substituting Eq. (11) into Eq. (7), the goal of optimization becomes

$$G = \min\left(\sum_{i=1}^{N_{tc}}\left(\frac{A_i}{2f_i\sqrt{2R_i h - h^2}} + \frac{S_{io} + S_{ii}}{2f_i} + t_{tc}\right)\right) \tag{12}$$

The number of tool changes is limited to within 3, i.e. $N_{tc} \le 3$. Too many tool changes is generally not desired in machining practice since it can affect surface finish and cause unnecessary tool wear.

5.2 Optimization Procedure

A free-form surface is studied to illustrate the optimal tool selection procedure. As shown in Figure 13a, the surface is a B-Spline surface with 6×11 control points. The middle part of surface is of concave shape.

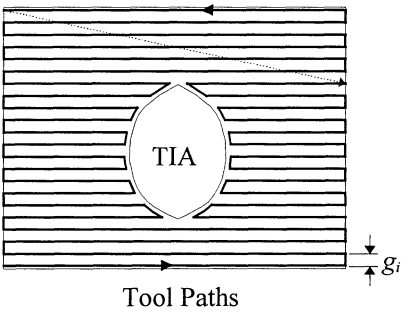


Figure 12. The tool-path topology for avoiding interference

Following machining condition is adopted. The required scallop height $h = 0.1$ mm. The radii of available tools $R_1 = 19.05$ mm ($\frac{3}{4}$"), $R_2 = 15.875$ mm ($\frac{5}{8}$"), $R_3 = 12.7$ mm ($\frac{1}{2}$"), $R_4 = 9.525$ mm ($\frac{3}{8}$"), $R_5 = 6.35$ mm ($\frac{1}{4}$"), $R_6 = 4.7625$ mm ($\frac{3}{16}$"), $R_7 = 3.96875$ mm ($\frac{5}{32}$"), $R_8 = 3.175$ mm ($\frac{1}{8}$"). The feedrates $f_1 = f_2 = f_3 = 152.4$ mm/min (6in/min), $f_4 = f_5 = 127$ mm/min (5in/min), $f_6 = f_7 = f_8 = 101.6$ mm/min (4in/min). The tool change time $t_{tc} = 1$ minute. The optimization is performed in following steps:

Step 1 Check the global interference by Algorithms 1 and 2. If the interference does present, user is warned and asked for revising his design. In case that the warning is ignored,

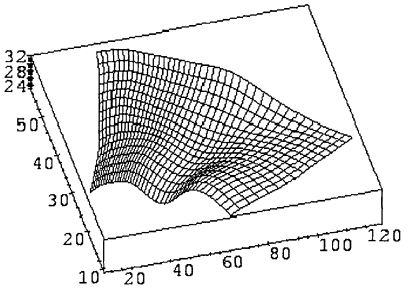
the accessible surface area is evaluated and stored. There is no global interference in this example surface.

Step 2 Calculate the TIAs for the available tools by using Algorithms 3 and 4. The calculation ends when either the TIA of a tool is zero or the smallest tool is reached. As a result, a set of tools (called the candidate tools) and their corresponding TIA boundaries are formed. For the example surface, the candidate tools are R_1, R_2, R_3, R_4, R_5, R_6, and R_7; and Figure 13b shows the TIAs.

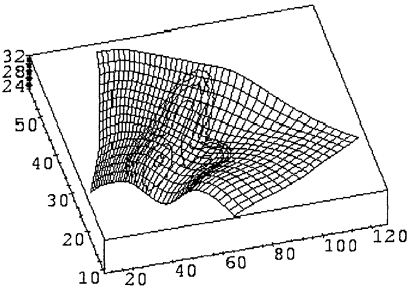
Step 3 Calculate the accessible area, gouge area, and lengths of outer and inner boundary for each candidate tools. The results are shown in Table 1.

Table 1 The accessible area of the candidate tools

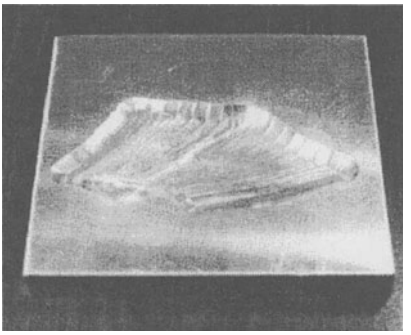
	R_1	R_2	R_3	R_4	R_5	R_6	R_7
Accessible Area (mm^2)	2012	2122	2263	2495	2779	2863	2869
Gouge Area (mm^2)	857	747	606	374.2	90	6.1	0
Outer Boundary (mm)	253.8	141.4	135.7	128.1	114.9	43.29	11.59



(a) a free-form surface



(b) TIAs for different cutters



(c) the machined surface with two cutters

Figure 13. Optimal tool selection

Step 4 Based on the general shape of TIA and surface geometry (Figure 13b), the iso-parametric machining on u direction is the preferable tool-path topology.

Step 5 According to user specified number of tool changes, combinations of the candidate tools are formed. The smallest tool of the candidate tools, R_7, is included in every combination so that the whole surface can be machined. For each combination, the machining time is calculated according to Eq. (12).

Tables 2, 3 and 4 show the machining time for different combinations when desired machining passes are 1, 2 and 3, respectively.

Table 2 The machining times for 1-pass machining

Combination	R_7
Mach. Time (min.)	18.20

Table 3 The machining times for 2-pass machining

Combination	R_1 & R_7	R_2 & R_7	R_3 & R_7	R_4 & R_7	R_5 & R_7	R_6 & R_7
Mach. Time (min.)	12.14	12.01	11.92	13.13	13.63	17.91

Step 6 Compare the machining times for the combinations and select the one with the shortest time as the optimal tool sizes. If only one machining pass is allowed, tool R_7 will be the optimal tool. Form Table 3, the combination of R_3 and R_7 is the best for machining the surface with 2 passes. For 3-pass machining, one should choose the combination of R_1, R_5, and R_7. Comparing to 1-pass machining, the 2-pass and 3-pass machining can result in 34.5% and 35.1% time savings. The 2-pass machining is then selected since it asks for less tool changes. Figure 13c shows a picture of the machined surface with cutters R_3 and R_7. The tool paths are iso-parametric. The real machining time is fairly closed to the predicted time.

Table 4 The machining times for 3-pass machining

Combination	R_1, R_2, R_7	R_1, R_3, R_7	R_1, R_4, R_7	R_1, R_5, R_7	R_1, R_6, R_7
Mach. Time (min.)	13.62	13.08	12.72	11.81	12.84
Combination	R_2, R_3, R_7	R_2, R_4, R_7	R_2, R_5, R_7	R_2, R_6, R_7	R_3, R_4, R_7
Mach. Time (min.)	13.35	12.89	11.91	12.76	12.18
Combination	R_3, R_5, R_7	R_3, R_6, R_7	R_4, R_5, R_7	R_4, R_6, R_7	R_5, R_6, R_7
Mach. Time (min.)	12.12	12.74	13.92	14.17	14.71

6. DISCUSSIONS AND CONCLUSIONS

The optimal tool selection method presented in this paper is designed for any type of parametric surfaces to be machined; and it is used as an inline process in producing interference-free high-efficiency tool paths for 3-axis NC machining using ball-end cutters.

All algorithms presented are implemented and tested. Algorithms 1, 4, and 5 are grid-line based. They are fairly robust and concise but demand high computational power if very fine resolution of grid is imposed. Since a linear interpolation is adapted in algorithms 1 and 4, the number of grid line does not have to be very high. From our experience, a 100×100 grids is sufficient for a 125mm by125mm surface. Algorithm 5 demands higher resolution to determine the TIAs accurately. Normally, one can get smooth TIA boundary curve if the density of grid is doubled. It should be pointed out that algorithm 5 uses only local grids while algorithms 1 and 4 need global grids. Furthermore, the error caused by the interpolation procedure in algorithm 4 can be remedied by algorithm 5 since it is a complete searching process. The resulting collision-free area from Algorithm 3 is conservative. The algorithm could be improved by adaptively offsetting the projected boundary curves. Finally, although algorithms 1, 2 and 3 are designed for detecting the global interference, they can also be incorporated in computer aided process planning (CAPP) systems as tools for testing surface manufacturablity.

A model of machining time for different tool sizes is proposed. In the model, machining time is determined by considering path interval and accessible surface area. Some key simplifications are: 1. the path interval is calculated by assuming flat surface; and 2. the tool-path topology is assumed to a zigzag pattern as shown in Figure 12. According to our case studies, the predicted machining time from the model agrees with the real machining time well. This subject deserves some more study.

Currently, a comparison of all possible combinations of tools is performed to determine the optimal tools. This could be a time consuming procedure when the number of available tools is large. We believe that sound searching strategies can be found either heuristically or analytically to simplify this process.

ACKNOWLEDGMENT

The authors gratefully acknowledge the financial support from NSF grant DMI-9713985 under which the present investigation was possible.

REFERENCES

1. Bobrow, J. E., March 1985, "NC Machine Tool Path Generation from CSG Part Representations," *Computer-Aided Design*, Vol. 17, No. 2, pp 69-76.
2. Broomhead, P., and Edkins, M., 1986, "Generating NC Data at the Machine Tool for the Manufacture of Free-Form Surface," *INT. J. PROD. RES.*, Vol. 24, No. 1, pp 1-14.
3. Chang, C. H. and Melkanoff, M. A., 1989, *NC Machine Programming and Software Design*, Prentice Hall.
4. Choi, B. and Jun, C., 1989, "Ball-End Cutter Interference Avoidance in NC Machining of Sculptured Surfaces," *Computer-Aided Design*, Vol. 21, No. 6, pp 371-378.
5. Gaal, B., and Varady, T, 1981, "Experiences and Further Development of the FFS(Free-Form Shapes) CAD/CAM System," *Robot. & Comput. Integrated Manuf.*, Vol. 2, No. 2, pp 186-197.
6. Han, Z and Yang, D., 1998, "Iso-Phote Based Tool-Path Generation for Machining Free-Form Surfaces," *Journal of Engineering for Industry*, in press.
7. Held, M., Lukacs, G., and Andor, L., 1994, "Pocket Machining Based on Contour-Parallel Tool Paths Generated by Means of Proximity Maps," *Computer-Aided Design*, Vol. 26, No. 3, pp 189-203.
8. Huang, Y., and Oliver, J. H., 1992, "Non-Constant Parameter NC Tool Path Generation of Sculptured Surfaces," *ASME Computers in Engineering*, Vol. 1, pp 411-419.
9. Hwang, J. S., Dec. 1992, "Interference-Free Tool-Path Generation in the NC Machining of Parametric Compound Surfaces," *Computer-aided Design*, Vol. 24, No. 12, pp 667-676.
10. Lee, Y. S., and Chang, T. C., Oct. 1995, "2-Phase Approach to Global Tool Interference Avoidance in 5-Axis Machining," *Computer-Aided Design*, Vol. 27, No. 10, pp 715-729.
11. Li, H., Dong, Z., and Vickers, G. W., 1994, "Optimal Toolpath Pattern Identification for Single Island, Sculptured Part Rough Machining Using Fuzzy Pattern Analysis," *Computer-Aided Design*, Vol. 26, No. 11, pp 787-795.
12. Lin, R, and Koren, Y., 1996, "Efficient Tool-Path Planning for Machining Free-Form Surfaces," *ASME Journal of Engineering for Industry*, Vol. 118, pp 20-28.
13. Loney, G. C. and Ozsoy, T. M., 1987, "NC Machining of Free-Form Surfaces," *Computer-Aided Design*, Vol. 19, No. 2 , pp 85-90.
14. Mortenson, M. E., 1985, *Geometric Modeling*, John Wiley.
15. Oliver, J. H., Wysocki, D. A., and Goodman, E. D., 1993, "Gouge Detection Algorithms for Sculptured Surface NC Generation", *Journal of Engineering for Industry*, Vol. 115, pp139-144.
16. Persson, H., 1978, "NC Machining of Arbitrarily Shaped Pockets," *Computer-Aided Design*, Vol. 10, No. 3, pp 169-174.
17. Suh, Y. S., and Lee, K, May 1990, "NC Milling Tool Path Generation for Arbitrary Pockets Defined by Sculptured Surfaces," *Computer-Aided Design*, Vol. 22, No. 5, pp 273-284.

18. Suresh, K., and Yang, D., 1994, "Constant Scallop-Height Machining of Free-form Surfaces," *ASME Journal of Engineering for Industry*, Vol. 116, pp 253-259.
19. Tang, K., Cheng, C. C., and Dayan. Y., 1995, "Offsetting Surface Boundaries and 3-Axis Gouge-Free Surface Machining," *Computer-Aided Design*, Vol. 27, No. 12, pp 915-927.
20. Tiller, W. and Hanson, E., 1984, "Offset of Two-Dimensional Profiles," *IEEE Computer Graphics Application*, Vol. 4, No. 9, pp 36-46.
21. Toussaint, G. T., 1985, *Computational Geometry*, Elsevire Science Pub. Co.
22. Tseng, Y. J. and Joshi, S., 1991, "Determining Feasible Tool-Approach Direction for machining Bezier Curves and Surface," *Computer-Aided Design*, Vol. 23, No. 5, pp 367-379.