Generation of Smooth Five-axis Tool-paths for Finish Cut of Freeform Surfaces with PSO

GENG Lin, Dept. of Mechanical Engineering, National University of Singapore, Singapore, mpegengl@nus.edu.sg

ZHANG Yunfeng, Dept. of Mechanical Engineering, National University of Singapore, Singapore, mpezyf@nus.edu.sg

LEE Kim Seng, Dept. of Mechanical Engineering, National University of Singapore, Singapore, mpeleeks@nus.edu.sg

Abstract: In 5-axis finish cut of freeform surfaces, drastic change in tool postures is harmful in many aspects. On the premise of no machining interference, it is desirable to generate 'smooth' tool-paths with confined posture change between cutter contact (CC) points. In this paper, interference-free tool-paths are first generated based on a heuristic aiming at better machining efficiency. For the problematic postures with out-of-bound posture change on this tool-path, a hybrid PSO (Particle Swarm Optimization) algorithm is developed to correct them. In this way, tool-paths can be generated with balanced performance regarding posture smoothness and machining efficiency.

Key-Words: *tool-path generation, cutter accessibility, tool-path optimization, PSO*

1. Introduction

Five-axis machining provides enhanced cutter accessibility, making it preferable for the fabrication of freeform surfaces. On the other hand, tool-path generation for 5-axis machining is a difficult task. The main challenge is to avoid machining interference, i.e., local gouging, rear gouging, and global collision [1]. Over the years, various methods have been successfully proposed to generate interference-free tool-paths aiming at maximal machining efficiency [2-6].

In 5-axis machining, a large posture (orientation) change between neighboring cutter locations (CLs) may cause potential interference, discontinuities in the surface finish and feedrate drop in actual machining. Therefore, the smoothness of tool-paths is also an important factor to be considered for finish cut tool-paths. So far, this topic has received limited attention. Ho et al. used a quaternion interpolation method to generate smooth tool-paths [7]. However, in terms of interference avoidance, the method is still trial-and-error in nature. Wang and Tang utilized the concept of C-space for generating tool-paths using a bi-directional search algorithm [8]. The method is based on heuristic and does not guarantee quality of the solution. Bi et al. proposed to use Dijkstra's algorithm to optimize the tool-path towards minimum total angular movement [9]. Although this method could reduce total posture change, it ignores possible drastic posture change between individual pairs of CC points. Castagnetti et al. used a gradient-based optimization tool to smooth 5-axis joint movements within the domains of admissible orientations [10]. The algorithm is effective is controlling the smoothness of tool-paths but neglects other requirements like machining efficiency and surface finish quality.

In this paper, a new algorithm is proposed for generating smooth interference-free tool-paths. In the algorithm, feasible postures are first assigned to the CC points using a heuristic based method. Then a hybrid PSO is deployed to smooth the posture sequences that have beyond-limit posture changes. Cutter accessibility and surface finish requirements are taken as constraints to regulate the search, so that the quality of the tool-path after smoothing is guaranteed.

Received: 27 February 2012, Accepted: 1 March 2012
Available online 12 March 2012

2. The overall approach

Tool-path optimization requires the accessible posture ranges for the cutter at all the CC points. This task is finished using an accessibility checking algorithm developed in our previous research [6], which outputs the accessible posture range of a cutter to an arbitrary surface point in the form of *accessibility maps*, short as A-maps. A-maps make up the feasible search space of cutter postures for further optimization. As an example, Fig. 1 gives a surface point on a workpiece and the A-map at the point projected onto a unit sphere.

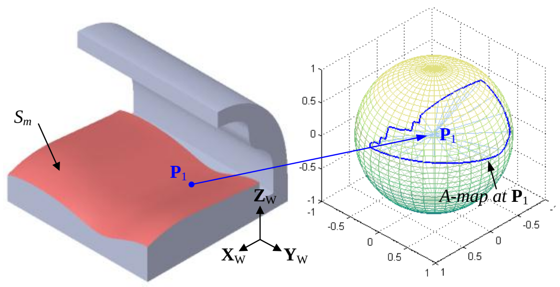


Figure 1. Workpiece model and A-map at sample point $\mathbf{P}_1$

The tool-path pattern used in this study is iso-planar. The CC points are generated on the intersection curves between the machining surface and a series of parallel cutting planes. Suppose the cutting direction is along $\mathbf{X}_W$ of the workpiece frame $\mathbf{O}_W$-$\mathbf{X}_W\mathbf{Y}_W\mathbf{Z}_W$. The cutting plane of the first path is set to be just off the surface edge with a small distance Δy_0. The CC points are generated on this path based on the profile tolerance using the method given in [11]. At each CC point, a posture is selected from the A-map at the point based on the heuristic given in [6] so that the *preliminary tool-path* is obtained. As the heuristic is aimed at maximized machining efficiency, the smoothness of the preliminary tool-path is not considered. To eliminate possible drastic posture changes, the posture changes along the path are checked to identify the unstable CL clusters, followed by the proposed PSO algorithm to smooth them. During this process, interference avoidance and surface finish requirements work as search constraints to guarantee the feasibility of the new tool-path. For the next path, the side-step Δy is determined based on scallop-height tolerance. The above mentioned procedure, from CC point generation to posture determination, is then carried out again on this new path. This process continues until the whole machining surface is fully covered, as shown in Fig. 2.

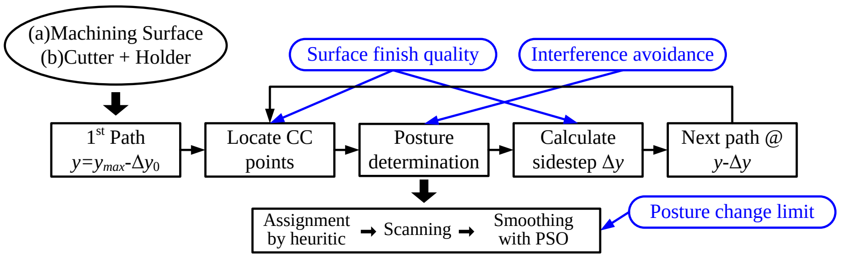


Figure 2. The proposed tool-path generation algorithm

3. The heuristic based tool-path generation algorithm

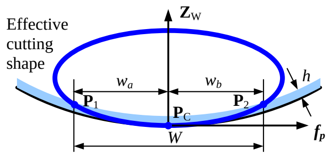
In the work of Li and Zhang [11], the tool postures for the CC points on a path are selected from the A-maps based on a heuristic such that the selected postures produce near-maximum strip widths. Therefore, the heuristic is pro-efficiency. In this paper, this heuristic will be modified to generate the preliminary tool-path to accommodate two requirements: efficiency and sufficient room for further refinement. In this section, the original heuristic is introduced first followed by the modified heuristic.

In the original heuristic, at a CC point $\mathbf{P}_C$ (see Fig. 3a), the cutting strip is divided into two parts by $\mathbf{P}_C$, given as w_a and w_b. To keep the scallop height within the given tolerance h, the machining strips of two adjacent paths should overlap (see $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$ in Fig. 3b). Yet for better efficiency, the level of overlap should be kept as low as possible. Thus, the side-step

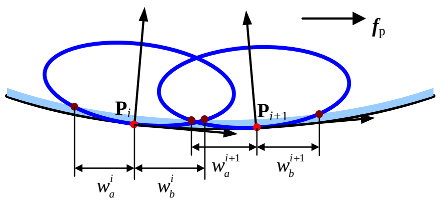
Δy between two neighboring paths *i* and *i*+1 should satisfy:

$$\eta\left(w_{b,\min}^{i}+w_{a,\min}^{i+1}\right)\le\Delta y\le(\eta+0.05)\left(w_{b,\min}^{i}+w_{a,\min}^{i+1}\right)\qquad(1)$$

where η is a parameter in the range of (0~1) controlling the level of overlapping, $w_{b,\min}^{i}$ and $w_{a,\min}^{i+1}$ are the minimum w_b^i and w_a^{i+1} at all the CC points on paths *i* and *i*+1, respectively. For good efficiency, η should be given a large value, e.g., 0.95; *Δy* can then be obtained by an iterative algorithm in which *Δy* is adjusted till Eq.(1) is satisfied. In this process, the cutter posture at each CC point on path *i*+1 is also determined.



(a) Cutter strip width



(b) Cutter strips on neighboring paths

Figure 3. Side-step determination based on cutting strip width

In this study, since both smoothness and efficiency are considered, the original heuristic is modified by using a relatively smaller value of η, such as 0.7. This change serves two objectives:

1) η = 0.7 will still result in tool-paths with relative good efficiency.
2) In case of out-of-bound posture change, there will be plenty room left for adjusting the postures at those problematic CLs due to the fact that scallop-height tolerance will not be easily violated.

This heuristic was tested and the results show that the produced preliminary tool-paths with open freeform surfaces have relatively good smoothness in general (See Fig. 4). However, in the presence of machining obstacles, drastic posture changes exist in the preliminary tool-paths. Further smoothing is therefore required.

4. POSTURE SMOOTHING WITH PSO

4.1 Indentifying the unstable CL clusters from the preliminary tool-paths

In presence of machining obstacles, the posture sequence on a preliminary tool-path usually has drastic posture changes (see Fig. 4b and c). A checking process is thus conducted to find the *unstable CLs* and divide the whole set of CLs into stable and unstable clusters. Only the unstable CL clusters will go through the smoothing process.

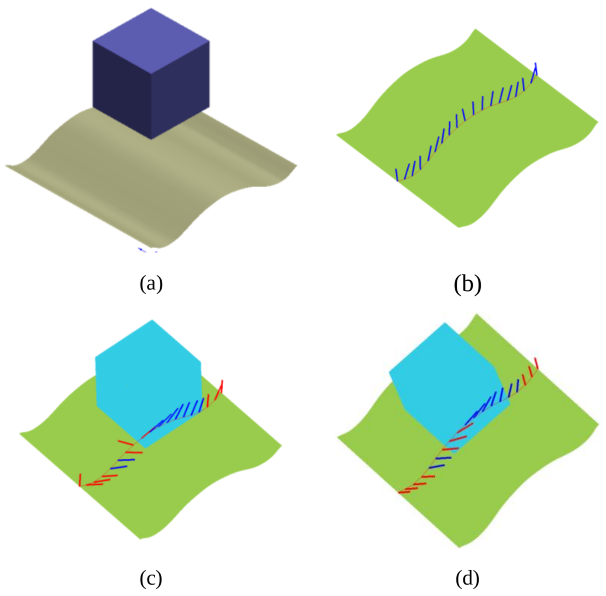


Figure 4. (a) Workpiece model (b) Tool-path without obstacle (c) Tool-path with obstacle present (d) Tool-path after smoothing

Suppose there are *N* CLs on the current tool-path, the criterion for identifying unstable CLs is as follows: Firstly, for neighboring postures $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$ (k = 1, 2, …, N-1), if their angular difference $\Delta\theta_{k,k+1}$ exceeds the given limit θ_{max}, both $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$ are marked as *unstable*. Secondly, suppose $\boldsymbol{p}_s$ and $\boldsymbol{p}_{s+n}$ are two stable postures with only unstable postures in-between, if $\Delta\theta_{s,s+n} > (n-1)\theta_{max}$, $\boldsymbol{p}_{s+n}$ will be marked as unstable. This is to ensure that the average angular difference among a consecutive set of unstable CLs must not exceed θ_{max}, thus making the smoothing possible. After all the unstable CLs are identified, the consecutive unstable CLs will be grouped into an unstable CL cluster. For the case shown in Fig.

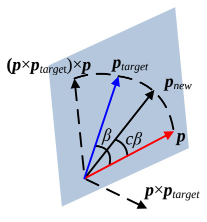
4c, 3 unstable CL clusters (in red) are identified.

4.2 PSO-based smoothing of unstable CL clusters

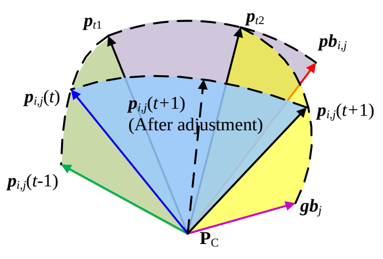
Particle Swarm Optimization (PSO) is a population-based search method. A PSO maintains a swarm of particles. The location of each particle represents a candidate solution. In our case, at a time instant t, a particle location represents a possible posture sequence for an unstable CL cluster. Therefore, if the swarm size is M, a particle position is represented by $\boldsymbol{X}_i(t) = \{\boldsymbol{p}_{i,m}, \boldsymbol{p}_{i,m+1} \ldots \boldsymbol{p}_{i,m+l}\}$, $i = 1, 2, \ldots, M$. At the subsequent time instant t+1, $\boldsymbol{X}_i$ is updated using our customized updating rule. The location of a particle is updated based on the *last update direction*, the *personal best location*, $\boldsymbol{PB}_i(t) = \{\boldsymbol{pb}_{i,m}, \boldsymbol{pb}_{i,m+1} \ldots \boldsymbol{pb}_{i,m+l}\}$, and the *global best location* $\boldsymbol{GB}(t) = \{\boldsymbol{gb}_m, \boldsymbol{gb}_{m+1} \ldots \boldsymbol{gb}_{m+l}\}$ among all the $\boldsymbol{PB}_i(t)$. Following this principle, in our customized updating rule, for a posture $\boldsymbol{p}_{i,j}(t)$ taken from $\boldsymbol{X}_i$, it will be updated 3 times based on $\boldsymbol{p}_{i,j}(t\text{-}1)$, $\boldsymbol{pb}_{i,j}$, and $\boldsymbol{gb}_j$ consecutively, one at a time. More specifically, when a posture $\boldsymbol{p}$ is being updated based on a target posture $\boldsymbol{p}_{target}$, the resultant posture $\boldsymbol{p}_{new}$ is on the plane determined by $\boldsymbol{p}_{target}$ and $\boldsymbol{p}$, given as:

$$\boldsymbol{p}_{new} = f\left(\boldsymbol{p}, \boldsymbol{p}_{target}, c\right) = \left(\boldsymbol{p} \times \boldsymbol{p}_{target} \times \boldsymbol{p}\right)\sin\left(c\beta\right) + \boldsymbol{p}\cos\left(c\beta\right) \quad (2)$$

where β is the angle between $\boldsymbol{p}$ and $\boldsymbol{p}_{target}$ and c is a parameter controlling the step size of updating (see Fig. 5a). $\boldsymbol{p}_{i,j}(t\text{-}1)$ is used as $\boldsymbol{p}_{target}$ in the first update while $\boldsymbol{pb}_{i,j}$ and $\boldsymbol{gb}_j$ are used as the $\boldsymbol{p}_{target}$ for the remaining 2 updates. The 3-step updating process are: (i) $\boldsymbol{p}_{t1} = f(\boldsymbol{p}_{i,j}(t), \boldsymbol{p}_{i,j}(t\text{-}1), c_0)$, (ii) $\boldsymbol{p}_{t2} = f(\boldsymbol{p}_{t1}, \boldsymbol{pb}_{i,j}, r_1c_1)$, and (iii) $\boldsymbol{p}_{i,j}(t+1) = f(\boldsymbol{p}_{t2}, \boldsymbol{gb}_j, r_2c_2)$. c_0, c_1 and c_2 are step sizes assigned to the three updates while r_1 and r_2 are random numbers in the range of (0~1) designed to add a stochastic element to the search. This 3-step updating process is shown in Fig. 5b.



(a)



(b)

Figure 5. Illustration of the customized updating rule in the developed PSO

Some important settings of the smoothing PSO are given as follows:

1) ***Cost Function***: The objective is to minimize the maximum posture change along an unstable CL cluster. For a candidate solution $\boldsymbol{X}_i$, the cost function is given as: $F(\boldsymbol{X}_i) = max\{\Delta\theta_{k,k+1} | \, k = m\text{-}1, \ldots, m+l\}$.

2) ***Search constraint***: During particle update, the following 2 constraints are imposed on each posture: (i) the posture must be interference-free and (ii) the new postures do not violate scallop-height tolerance. For constraint (ii), let's consider two neighboring tool-paths numbered k and k+1, the interval between which is given as Δy. The preliminary cutter postures on path k+1 need to go through smoothing and still be able to cover the gap between path k+1 and path k. Thus, limits should be imposed on the left strip with (w_a) of the CC points on path k+1. Noting that the CC points on two neighboring paths may not be perfectly aligned, we assume that the strip width changes linearly between CC points. For a CC point $\mathbf{P}_\text{n}$ on path k+1 that corresponds to CC points $\mathbf{P}_\text{m}$ and $\mathbf{P}_{\text{m}+1}$ on path k, the limit on the right left strip with at $\mathbf{P}_\text{n}$ is given as (See Fig. 6):

$$w_{n,\lim} = \Delta y - c\left(\frac{x_n - x_m}{x_{m+1} - x_m}\left(w_{b,m+1} - w_{b,m}\right) + w_{b,m}\right) \quad (2)$$

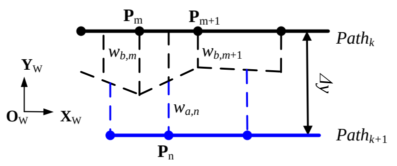
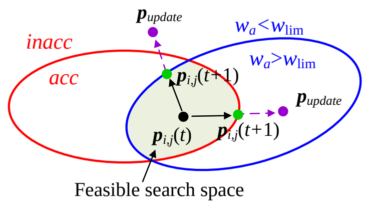


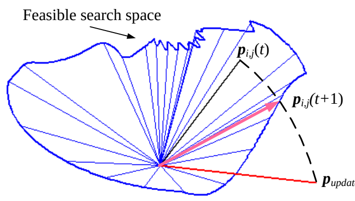
Figure 6. Calculation of strip width limit for constraint (ii)

Using the update rule given earlier in this chapter, it's possible that the newly obtained posture $\boldsymbol{p}_{i,j}(t+1)$ may violate either or both of the constraints. Assuming both cutter accessibility and cutting strip width changes continuously with cutter posture, two posture ranges exist around $\boldsymbol{p}_{i,j}(t)$, i.e. the accessible posture range (A-map) and the posture range that satisfies the limit on right strip width. The intersection of these two posture ranges makes up the feasible search space for $\boldsymbol{p}_{i,j}(t+1)$. With such a feasible search space around $\boldsymbol{p}_{i,j}(t)$, when $\boldsymbol{p}_{i,j}(t+1)$ is close enough to $\boldsymbol{p}_{ij}(t)$, a feasible posture would always exist (see Fig. 7). Thus, in case of violated constraints, $\boldsymbol{p}_{i,j}(t+1)$ can be found by iteratively reducing the angle between $\boldsymbol{p}_{update}$ and $\boldsymbol{p}_{i,j}(t)$.

It's worth noting this adjusting procedure is based on the fact that $\boldsymbol{p}_{i,j}(t)$ is feasible. This requires both constraints be taken into consideration when the swarm is being initialized. Such a measure makes sure that only feasible solutions exist in the swarm from the beginning of the search.



(a) 2-D illustrative example of update strategy



(b) Adjustment of update step in 3-D space

Figure 7 Adjusting update step to guarantee feasibility of solution

3) ***Hybrid PSO and Mutation Operator***: Classic PSO could be trapped in local minima easily, which is why a hybrid PSO with a mutation operator is used in our algorithm. There are 2 levels of mutations, i.e., particle and swarm. Particle mutation happens throughout the search at a certain probability. It only replaces one randomly selected particle $\boldsymbol{X}_i(t)$ with a randomly generated solution $\boldsymbol{X}_i(t+1)$. The purpose is to introduce more diversity to the search. On the other hand, swarm mutation only happens when the particles have converged around the global best location. The purpose is to exploit the solution space around the best solution obtained so far. It is like a re-initialization of the swarm, where all particles of the swarm are replaced with randomly generated new solutions. The difference is that the new random solutions are all generated within the mutation step of the best solution at that time step. To balance exploration and exploitation, the mutation step S should shrink with time t and yet stay above a certain level to reserve enough space for mutation. Based on experiments, the following rule is used (S_{max} and S_{min} are user defined maximum and minimum step sizes):

$$S(t) = e^{-t/t_{\max}} \left(\frac{F(\boldsymbol{GB}(t))}{F(\boldsymbol{GB}(0))} (S_{\max} - S_{\min}) + S_{\min} \right) \quad (3)$$

4) ***Stopping Criterion***: The search will stop when any of the following conditions is met: (i) $F(\boldsymbol{GB})<\theta_{max}$, (ii) $t>t_{max}$, and (iii) $\boldsymbol{GB}$ changes less than 0.1° for 50 iterations.

A flow chart showing the workflow the proposed PSO algorithm is given Fig. 8 for the readers' reference.

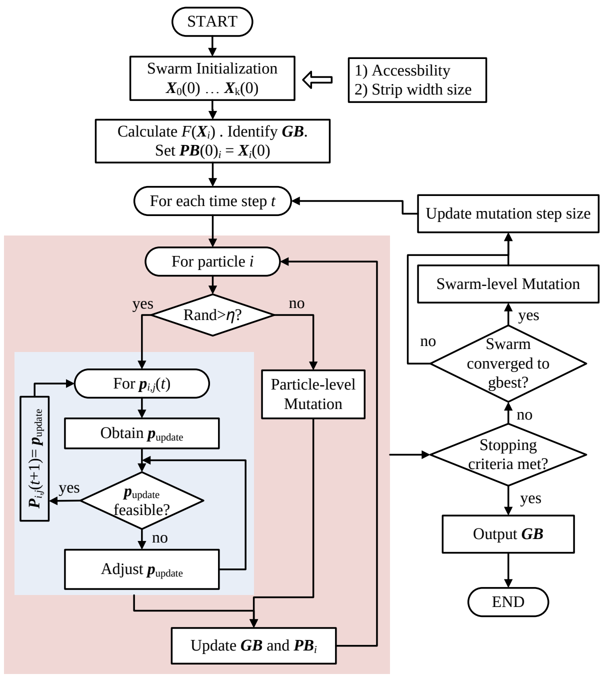


Figure 8 Workflow of the proposed PSO algorithm

For the example shown in Fig. 4c, the cutter postures after smoothing using the hybrid PSO is shown in Fig. 4d. It can be seen the smoothness has been improved significantly.

Fig. 9 shows the search process (cost vs. iterations) for both the classic PSO and the proposed hybrid PSO. It is obvious that the hybrid PSO clearly has the edge.

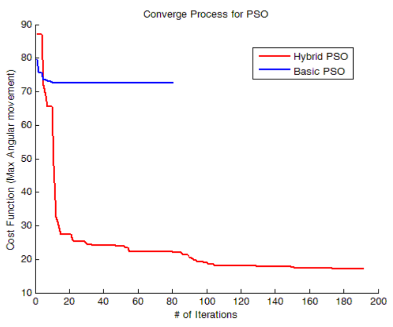


Figure 9. Search record with/without mutation

5. APPLICATION EXAMPLES

A complete test of the preliminary tool-path generation and the PSO smoothing algorithms is conducted on the part shown in Fig. 1 where half of the machining surface is covered by an arch-like obstacle. The tool-paths after smoothing showing the tool-postures are given in Fig. 10a. The tool-paths are proved to be interference-free using the simulation software VERICUT (see Fig. 10b).

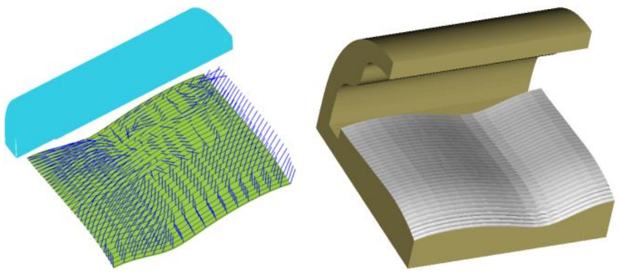


Figure 10. (a) tool-paths showing cutter postures
(b) Machining simulation result with VERICUT

Fig. 11a shows the tool postures on a tool-path taken from the preliminary tool-paths for the part in Fig. 1. There are 3 unstable CL clusters. The PSO smoothing algorithm is applied and the postures of the resultant tool-path are shown in Fig. 11b, in which there is no unstable CL clusters. Even based on direct observation, it is obvious that the smoothness of the tool-path has improved significantly. Furthermore, the quality of the "before" and "after" tool-paths is compared and the results are summarized.

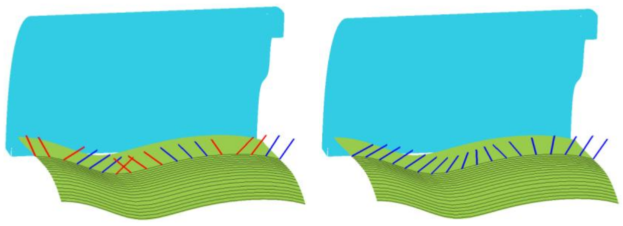
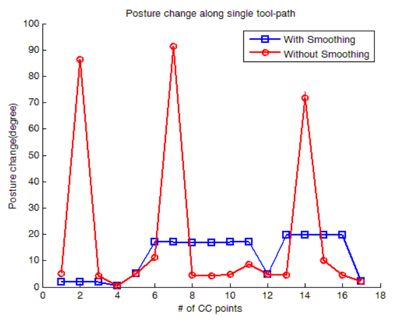
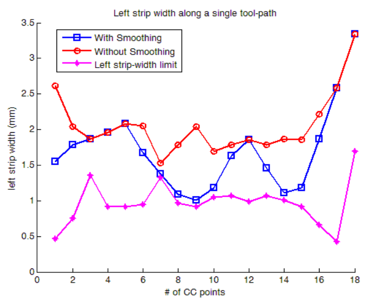


Figure 11 Preliminary postures and postures after smoothing

Fig. 12a shows the posture changes along the sample tool-path before and after smoothing. It can be seen that the PSO smoothing algorithm has the effect of spreading a large posture change evenly over the whole CL cluster. Fig. 12b shows the left strip widths (w_a) at every CL along the tool-path. It can be seen that w_a stays above the limit for all the newly generated CLs along the path.



(a)



(b)

Figure 12. (a) Angular difference between cutter postures before and after smoothing (b) Strip widths on smooth tool-path

6. Conclusions

In this paper, a novel method for generating smooth 5-axis tool-paths is proposed. The CC points and tool postures are first generated by heuristics. Then a smoothing method based on PSO is employed to repair the problematic segments with drastic posture changes. Search constraints regarding interference avoidance and surface finish quality are imposed to guarantee the feasibility of the modified tool-paths. Custom update rules are proposed for the PSO to guide the search away from infeasible solutions. As shown by the case studies, our algorithm is capable of producing smooth, interference-free tool-paths for 5-axis finish machining of freeform surfaces.

REFERENCES

[1] B.K. Choi, R.B. Jerard, Sculptured surface machining: theory and applications, Kluwer Academic Publishers Boston, 1998.

[2] T.C. Woo, B.F. von Turkovich, Visibility Map and Its Application to Numerical Control, CIRP Annals - Manufacturing Technology, 39 (1990) 451-454.

[3] W. Tony C, Visibility maps and spherical algorithms, Computer-Aided Design, 26 (1994) 6-16.

[4] M. Balasubramaniam, S.E. Sarma, K. Marciniak, Collision-free finishing toolpaths from visibility data, Computer-Aided Design, 35 (2003) 359-374.

[5] C.-S. Jun, K. Cha, Y.-S. Lee, Optimizing tool orientations for 5-axis machining by configuration-space search method, Computer-Aided Design, 35 (2003) 549-566.

[6] L.L. Li, Y.F. Zhang, Cutter selection for 5-axis milling of sculptured surfaces based on accessibility analysis, International Journal of Production Research, 44 (2006) 3303 - 3323.

[7] M.-C. Ho, Y.-R. Hwang, C.-H. Hu, Five-axis tool orientation smoothing using quaternion interpolation algorithm, International Journal of Machine Tools and Manufacture, 43 (2003) 1259-1267.

[8] N. Wang, K. Tang, Automatic generation of gouge-free and angular-velocity-compliant five-axis toolpath, Computer-Aided Design, 39 (2007) 841-852.

[9] Q.-Z. Bi, Y.-H. Wang, H. Ding, A GPU-based algorithm for generating collision-free and orientation-smooth five-axis finishing tool paths of a ball-end cutter, International Journal of Production Research, 48 (2010) 1105-1124.

[10] C. Castagnetti, E. Duc, P. Ray, The Domain of Admissible Orientation concept: A new method for five-axis tool path optimisation, Computer-Aided Design, 40 (2008) 938-950.

[11] L.L. Li, Y.F. Zhang, An integrated approach towards process planning for 5-axis milling of sculptured surfaces based on cutter accessibility, Computer-aided Design and applications, 3 (2006) 249-258.

[12] W. Anotaipaiboon, S.S. Makhanov, E.L.J. Bohez, Optimal setup for five-axis machining, International Journal of Machine Tools and Manufacture, 46 (2006) 964-977.