Int J Adv Manuf Technol (2015) 81:1233–1245
DOI 10.1007/s00170-015-7275-z

ORIGINAL ARTICLE

# Smooth contour-parallel tool path generation for high-speed machining through a dual offset procedure

**Zhiwei Lin** [1] · **Jianzhong Fu** [1,2] · **Hongyao Shen** [1] · **Xinhua Yao** [1]

Received: 26 January 2015 / Accepted: 4 May 2015 / Published online: 20 May 2015


**Abstract** In generating a contour-parallel tool path, traditional offset methods usually offer an arc rounding option for superior sharp corners. For the inferior ones, however, they are left untreated. The inferior sharp corners are harmful to high-speed machining (HSM). They should be dealt with when they emerge for the first time; otherwise, they will be further inherited by the offset successors. In this paper, we presented a dual offset procedure to round the inferior sharp corners. For a given path, literally, the procedure conducts two offsets: at first, the path is offset inwards, generating an intermediate curve; then, this curve is further offset outwards, outputting the dual offset curve. The dual offset curve is guaranteed to be smooth. It is used to replace the original path. One side effect of this procedure is that many uncut regions will be left in the corner areas. These uncut regions are detected through a Boolean operation and removed by smooth clear-up tool paths. Finally, the dual offset and clear-up tool paths are connected with a set of linking arcs. The proposed smooth tool path generation method is implemented and verified by several examples. Comparative cutting simulations show that this method has a potential to improve the machining efficiency in terms of HSM with nearly a zero cost.

**Keywords** Contour-parallel · Smooth tool path · Dual offset · High-speed machining

✉ Jianzhong Fu
fjz@zju.edu.cn

[1] The State Key Lab of Fluid Power Transmission and Control, Department of Mechanical Engineering, Zhejiang University, Hangzhou 310027, China

[2] Key Laboratory of 3D Printing Process and Equipment of Zhejiang Province, College of Mechanical Engineering, Zhejiang University, Hangzhou 310027, China

## 1 Introduction

The layer-by-layer machining approach is extensively used in both pocketing and surface roughing process. On each layer, the intersecting curves between the part model and the hunting plane form the boundary contours of the machining areas. Inside these areas, tool paths of different patterns can be filled, among which there are two most popular path filling strategies: direction-parallel and contour-parallel.

The direction-parallel tool path elements are actually a set of equidistant parallel line segments trimmed by the boundary curves. For this strategy, the inclining angles of the parallel lines have a direct effect on the number of tool retractions [1]. The contour-parallel tool path can be obtained by successively offsetting the boundary contours. It is generally considered that generating a contour-parallel tool path is much more difficult than that of a direction-parallel tool path. The main difficulty comes from the offset problem. Now many methods can be used to do planar curve offset. These methods are briefly reviewed as follows.

### 1.1 Traditional offset methods

In the past few decades, the planar curve offset problem has been heavily researched, both in computational geometry and tool path planning areas. As a result, many methods have been developed, such as Voronoi diagram, level set, edge offset, and vertex offset.

In mathematics, a Voronoi diagram is a partitioning of a plane into regions based on “closeness” to points in a specific subset of the plane. Persson [2] first introduced the Voronoi diagram concept to generate a contour-parallel tool path for the boundaries bounded by line and arc segments without islands. Held et al. [3, 4] further developed this method to handle pockets with islands. The Voronoi diagram offset

approach is known to be efficient; however, it has numerical instability in near circular portions [5, 6].

The level set method is derived from fluid dynamics and has been widely used in image processing, crystal growth, computer-aided design, and other domains [7]. Kimmel and Bruckstein [8] presented a level set method to generate the offset curves. In this method, the offset problem is formulated as an implicit level set equation. The resolution of the equation is treated as a wave-front propagation problem. A solution to the equation means the offset curves. However, the main difficulty of implementing the level set method is the implicitization of parametric functions.

Edge offset can be realized by offsetting each edge on the input curve at a given offset distance. However, this method might introduce local and global invalid loops. In an algorithm proposed by Kim and Jeong [9], only two cases of local problems are discussed; global invalid loops are removed by checking the sign of loop area. Liu et al. [10] discussed more cases of possible local problems. Their offset algorithm can deal with profile self-intersection, overlapping, and small arc problems. Choi and Park [6] developed an efficient pair-wise interference-detection (PWID) test-based offset algorithm. With PWID, invalid loops can be removed in linear time. Lai et al. [11] proposed a forward locus tracing method (FLTM) for invalid loops removal. The FLTM transforms 2D transversal intersection problems into 1D interval identification. If the first interval has no interference with other objects, then the odd interval group can form valid loops.

In vertex offset, the offset vertices are picked directly on the angular bisectors of the input curve. Just like the edge offset, it also has the invalid loop problems. Wong and Wong [12] used vertex offset to generate the initial offset curves; global invalid loops are removed by checking loop directions. Kim [5] and Kim et al. [13] fixed local problems by removing invalid direction edge. Besides direction validity, Lee et al. [14] also check position validity in order to remove local invalid loops; global invalid loops are removed by evaluating the distance from the offset vertices to the original curve. In our previous work in [15], instead of fixing the local problems on offset curves, we update the shape of original curves with the proposed profile updating rules. The global invalid loops are removed with a tree analysis procedure after all the self-intersections are identified.

Aside from the abovementioned methods, the straight skeleton can also be used to generate the offset curves [16]. Recently, a fast marching method which is inherently capable of dealing with island problem is proposed by Dhanik and Xirouchakis [17]. This method handles the topological changes during offsetting naturally and deals with the generation of discontinuities in the slopes by including an entropy condition in its numerical implementation. On the internet, sophisticated software packages for planar curve offset are freely available, such as CGAL (www.cgal.org) and Clipper (www.angusj.com/delphi/clipper.php).

After the offset curves have been generated, they should be properly linked. Park and Chung [18] presented a path-linking procedure which optimizes technological objectives, such as tool retractions, drilling holes, and slotting. Park et al. [19] developed a data structure called a "TPE-net," and based on that, they presented a linking algorithm which guarantees "zero" number of tool retractions. Hunduja et al. [20] proposed an algorithm which employs segments on the Voronoi diagram as links. Dai et al. [21] presented a graph model-based path-linking algorithm for dental restoration.

### 1.2 Existing problems

With the advances in machine tools and cutting tool technologies, high-speed machining (HSM) becomes a cost-effective manufacturing process to produce parts [22]. Nowadays, HSM is widely applied in modern manufacturing to produce exact structural parts and functional parts in aerospace and weapon industries with high machining efficiency, low cutting forces, higher part precision, and better surface quality [23]. HSM imposes new challenges as well as difficulties for the CAM tool path planning and optimization tasks. In the context of HSM, in order to maintain a small contouring error, the feedrate is decreased when sharp changes are found along the tool path. Monreal and Rodriguez [24] studied the influence of tool path strategy (especially the corner angles) on the cycle time of HSM. In their experiments, a significant reduction in actual average feedrate (compared with the programmed feedrate) was found. In another investigation conducted by Siller et al. [25], discrepancies of 300–800 % were found by comparing the actual cycle time with the ideal cycle time under programed feedrate up to 16,000 mm/min. For HSM purpose, it is required that the contour-parallel tool path should be as smooth as possible. Sharp corners will cause sudden direction changes in tool motion and they should better be eliminated.

However, most of the above reviewed offset algorithms focus only on the geometrical realization; very little attention has been paid to the technological issues. As a result, sharp corners may still exist on the obtained contour-parallel tool paths. These sharp corners are due to the building nature of offset mechanism and are not so easy to get rid of. During offset process, as shown in Fig. 1, some of the sharp corners are originally from the pocket boundaries; others may emerge during the offset process. Since sharp corners are harmful to the HSM process, any sharp corners should better be rounded when they appear for the first time; otherwise, they will be inherited by the successors, as shown in Fig. 1. Unfortunately, such inheritance seems to be inevitable with existing offset methods.

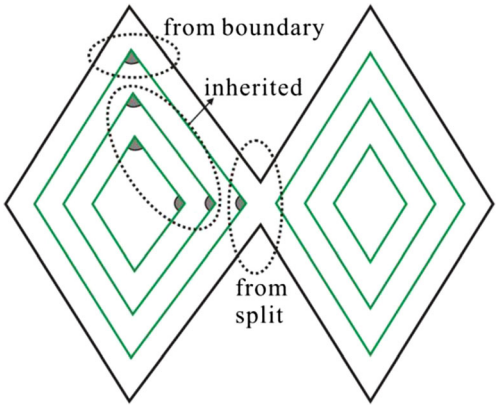


**Fig. 1** Sharp corners in offset process

Since traditional offset methods cannot solve the sharp corner problem well, some previous researchers tried to append extra tool path to the corners after all the offset curves have been generated. Choy and Chan [26] introduced a corner-looping-based tool path optimization. In their work, bow-like arc segments are appended to the basic path at the corner positions so as to smoothen the cutter motion, as shown in Fig. 2a. Though, on the optimized path, sharp corners still exist. Similarly, Zhao et al. [27] appended bi-arc transition segments to the corners, as shown in Fig. 2b. However, this method might not be robust enough for complex pockets: residual material might be left. Besides, Shi et al. [28] proposed a corner-rounding method by replacing the corner with a pair of quantic Pythagorean-hodograph (PH) curves. However, this method is not easy to be popularized since PH curves are not compatible with most of the conventional numerical control systems. Similarly, Pateloup et al. [29, 30] modifies the values of the corner radii on the contour-parallel tool path in order to increase real feedrate using B-spline curves.

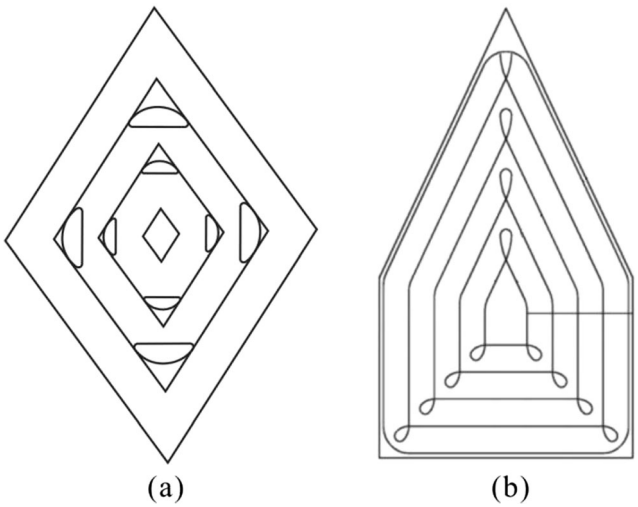


**Fig. 2** Tool path smoothing by appending extra curves

For contour-parallel HSM tool path, it is required that the sharp corners should be rounded, so that fast feedrate can be maintained; meanwhile, no residual material should be left in the corner areas. In this paper, unlike the previous researchers, we attempt to smooth the tool path during the offset process. Based on the traditional offset methods, a dual offset procedure is proposed. With this method, it is guaranteed that each sharp corner on the offset curve can be rounded. Literally, the dual offset method can be simply realized in two steps: at first, the input curve is offset inwards, generating an intermediate offset curve; then, this intermediate offset curve is further offset outwards, generating the output curve without any sharp corners. One side effect of this procedure is that uncut regions might also be generated. In this paper, uncut regions are detected through a Boolean operation. Closed clear-up tool paths are generated according to the shape of the uncut regions. All the offset and clear-up tool paths are then linked together to form the final tool paths.

The rest of the paper is organized as follows. In Section 2, the principle of the dual offset procedure is introduced. Based on that, the offset tool paths are smoothed in Section 3. Section 4 detects and removes all the uncut regions and Section 5 links together the offset and clear-up tool paths. In Section 6, two pocket examples as well as several cutting simulations are illustrated. The last section gives concluding remarks.

## 2 Dual offset procedure

During traditional curve offset process, sharp corners will inevitably appear and then be inherited by the successors. The sharp corners must be properly handled for HSM purpose. Before introducing the dual offset procedure, let us first review some of the basic curve offset terminologies.

### 2.1 Curve offset terminologies

#### *2.1.1 Offset direction and distance*

For a closed planar curve, if the offset curve is inside the original curve, the offset process is called an inward offset, and if the offset curve is outside the original curve, the offset process is called an outward offset. An inward offset can be realized by assigning a positive offset distance while an outward offset can be realized by assigning a negative offset distance. In this work, the offset distance is denoted by $d$.

For contour-parallel offset tool path, the offset distance directly influences the total length of the tool path. More specifically, larger offset distance results in shorter tool path length and higher machining efficiency. However, when the offset distance is greater than the cutter radius, uncut regions may appear in corner areas.

*2.1.2 Sharp corners classification*

On a general planar curve, two types of sharp corners can be found: inferior sharp corners and superior sharp corners, as shown in Fig. 3. Generally, for an inferior sharp corner, the inner angle of the corner is in a range of (0°, 180°), and for a superior sharp corner, the inner angle is in a range of (180°, 360°). Accordingly, the point in an inferior corner is called an inferior point, and the point in a superior corner is called a superior point. The detection of a sharp corner needs more restricted angle ranges. Technically, the restricted ranges are related to the machine tool capabilities. Note that a sharp corner has two incident line segments.

For an inferior sharp corner, traditional offset methods only pass on the sharp angle to the offset successors. For example, with the edge offset method, the adjacent two offset edges are crossing. They are trimmed and joined at the intersection point. As a result, the inferior sharp corner is inherited by the offset curve, as shown in Fig. 3. For other offset methods like vertex offset and Voronoi diagram, the same thing happens.

*2.1.3 Gap filling options for superior sharp corners*

Unlike the inferior sharp corners, for superior sharp corners, existing offset methods do offer several options to handle the corners. Let us study the cases in edge offset process for example. As shown in Fig. 3, in edge offset of a superior sharp corner, a gap will emerge. Three options can be used to fill the gap. The basic option extends the two offset segments to the common intersection point. With this strategy, sharp corner will be passed on to the offset curve. For the second option, a trapezoid is used to join the two offset segments. With this strategy, two more corners are generated, but the sharp corner is at least relieved. The third option fills an arc into the gap so that the arc makes tangential contact with the two offset edges. In this way, the superior sharp corner is well rounded. Apparently, for HSM tool path generation, the arc filling option is preferred.

Note that in the above discussion, the sharp corner properties (inferior or superior) are defined under an inward offset direction. When the offset direction reverses, the corner properties should also be reversed. And with existing offset methods, superior sharp corners can be rounded, but inferior sharp corners cannot. This work focuses specifically on smoothing the inferior sharp corners with a dual offset procedure. The rounding of superior sharp corners is not discussed but used as an existing technology.

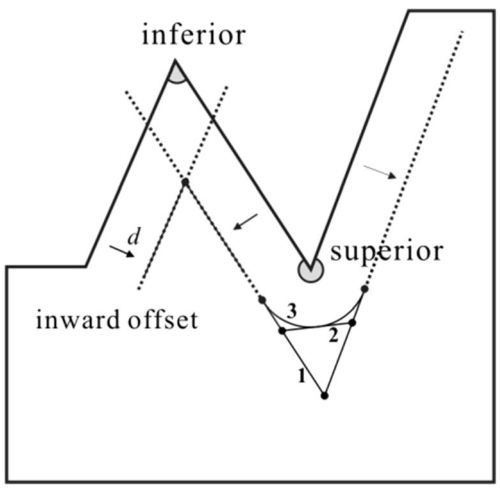


**Fig. 3** Inferior and superior sharp corners

## 2.2 Dual offset procedure

Let us take a planar curve for example, as shown in Fig. 4. On this curve, suppose the superior sharp corners have been rounded in the previous offset step. However, the inferior sharp corners still exist and they need to be rounded in the current step. An intuitive way to smooth these sharp corners is to approximate the input curve with a spline curve—a B-spline curve, for example. However, it is difficult to control the profile deviations: twisted or self-intersected spline curve might be generated. Another thing is that spline curve approximation does not guarantee the elimination of high curvatures, especially when the local details of the input curve are complex, as shown in Fig. 4.

Another intuitive way is to replace the inferior sharp corner with an inscribed arc, as arc3 shown in Fig. 4. In geometry, constructing such an arc for two known line segments is easy. The main difficulty comes from locating the right pair of edges for a given corner. Let us take corner *A* as shown in Fig. 4 for example. Corner *A* lies on a slim local profile. The lengths of the edges near the corner areas are shorter than those in other areas. Now, it is required to fit a big-sized arc into the corner. Suppose the radius of the arc is denoted by *R* and the value of *R* is pre-assigned.

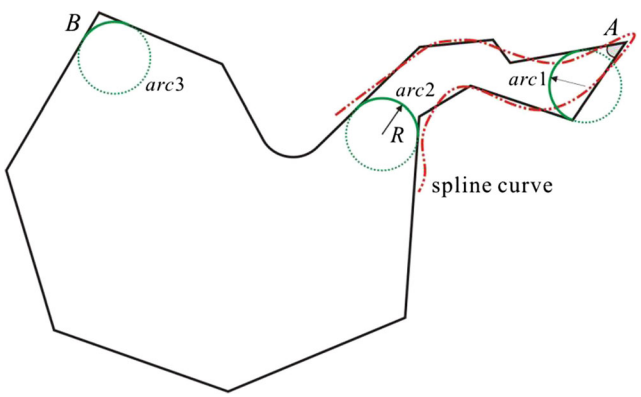


**Fig. 4** Rounding sharp corners by spline curves or by arcs

It starts with the two incident edges of corner $A$ and traverses down pair-wisely all the edges on both sides of the corner. During the traversing, there might be more than one pair of edges that satisfy the geometrical requirement, as arc1 and arc2 shown in Fig. 4. In this case, technological requirements, such as feed direction changes, path interference with the pocket boundary, need to be further checked. For the obtained arc1 and arc2 in Fig. 4, apparently, arc2 is more appropriate, since arc1 has two abrupt direction changes, so the whole slim corner is removed and replaced with arc2.

In the above discussion, the most difficult part is to find out the right pair of edges for the to-be-constructed inscribed arc. And there might also be chances that the inscribed arc does not exist. In this case, the radius of the arc is reduced to a proper value and the traversing process is run again.

To avoid the tedious and error-prone traversing process, we propose a dual offset procedure. Note that in the above arc calculation process, the center point of an inscribed arc is actually an inward offset point of the input curve at distance $R$. Theoretically, by properly connecting all the center points of arcs can form an offset curve, as shown in Fig. 5. Now, let us consider this problem conversely: suppose the inward offset curve of the input curve has been calculated in the first place. With a decent offset method, all the local and global problems appearing on the initial offset curve can automatically be fixed. Complex local profiles on the input curve degenerate into simple geometry elements on the offset curve. For example, in Fig. 5, the shaded slim profile near corner $A$ degenerates into a single point $D$ during the offset process. Note that here, $D$ is an inferior point. For the shaded corner, with offset point $D$ and radius $R$, the circle in which the inscribed arc lies can be determined. Base on the circle, the inscribed arc can be determined by calculating the two tangency points. Then, the shaded profile is replaced with the obtained arc. Similarly, based on the inward offset curve, all the inscribed arcs for all the inferior corners can be calculated. In this way, the annoying process of locating the right pair of segments is avoided.

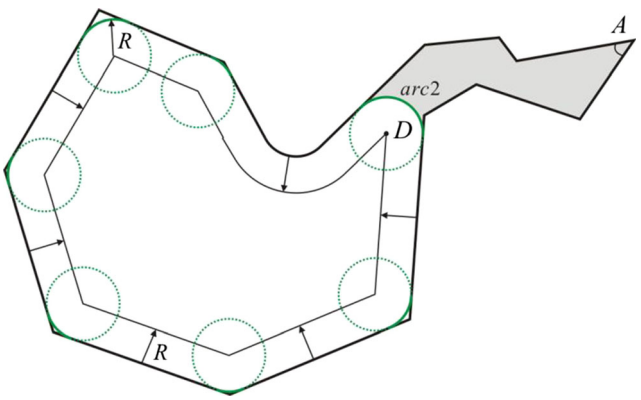


**Fig. 5** Rounding sharp corners by the dual offset procedure

However, the problem is calculating the tangency points between a circle and the input curve needs to traverse all the edges on the input curve. In fact, it is not necessary to manually calculate the inscribed arcs. On the obtained inward offset curve as shown in Fig. 5, point $D$ is an inferior point. If point $D$ is further offset inwards, another inferior point will be generated. However, to the outward direction, $D$ is a superior point and the corresponding corner is a superior corner. So if $D$ is offset outwards, with existing offset methods, a gap filling arc can be generated. If the outward offset distance is $-R$, opposite to the inward offset distance, then the obtained filling arc can be used as the inscribed arc for the input curve. Therefore, the inward offset curve is further offset outwards with the arc filling option, generating the dual offset curve. On this dual offset curve, inferior corners are automatically replaced by inscribed arcs. Another advantage is that it is no longer necessary to set up a range criterion to detect a sharp corner.

Based on the above discussion, for an input curve and a given arc radius $R$, only two steps are needed to realize the dual offset procedure. In the first step, the input curve is offset inwards at a distance $R$, generating an intermediate offset curve. In this step, for the offset method used, the arc filling option can be neglected to speed up the offset computation. In the second step, the intermediate offset curve is further offset outwards at a distance $-R$, generating the dual offset curve. In this step, for the offset method used, the arc filling option must be selected. The obtained smooth dual offset curve can be used as a tool path.

For an input curve which has an approximate size to the given inscribed arc radius, there might be chances that the inward offset curve does not exist. In this case, there are two approaches to deal with the input curve; either leave the curve as it is or reduce the radius value of the transition arc. In this work, we prefer the latter approach.

## 3 Offset tool path smoothing

With the dual offset procedure, it is now possible to smooth the offset tool path. The input includes the following: the pocket boundary $b_0$, the tool path interval $\omega$, the cutter radius $\rho$, and the dual offset radius $R$. In this work, $R$ is set proportional to $\rho$, namely

$$R = k_d \cdot \rho, \tag{1}$$

in which $k_d$ is the dual offset coefficient. Note here $b_0$ is curve set, which may include more than one boundary curves, similarly hereafter

Initially, for the input boundary $b_0$, it is offset inwards at a distance $\rho$, generating the first offset curve $b_1$, as shown in Fig. 6. When moving along $b_1$, the cutter directly contacts the pocket boundary. It is not necessary to apply the dual offset

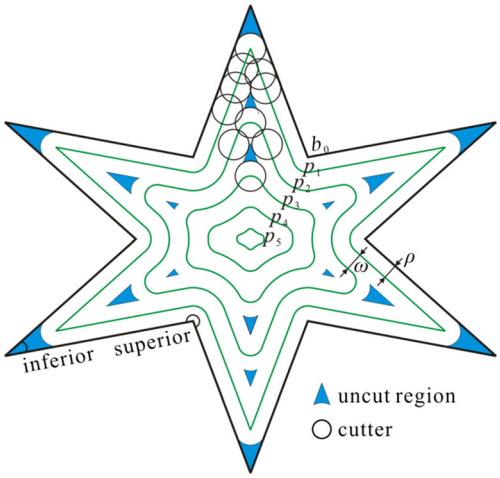


**Fig. 6** Dual offset tool path generation

procedure for this curve; otherwise, more uncut material will be left in the inferior corners. However, during the first offset, it is necessary to select the arc filling option for superior corners, as shown in Fig. 6. The obtained $b_1$ is used directly as the first tool path $p_1$, namely, $p_1=b_1$. As there might be sharp corners on $p_1$, the feedrate for $p_1$ can be set relatively low.

Then, $p_1$ is further offset inwards at a distance $\omega$, generating the second offset curve $b_2$. In this offset, arc filling option for superior corners should also be selected. So, it is guaranteed that there are no superior sharp corners on $b_2$. The dual offset procedure is applied to $b_2$, generating a curve $p_2$, as shown in Fig. 6. The obtained $p_2$ is used as the second tool path. As there are no superior or inferior sharp corners on $p_2$, the feedrate for $p_2$ can be set relatively high.

Similarly, $p_2$ is used to generate a third tool path $p_3$. This process goes on iteratively, generating more smooth tool paths $(p_3, p_4, \ldots, p_n)$ until the offset stop condition is met. During the successive offset process, for the last few offset curves, the dual offset procedure may fail because the inward offset curves cannot be generated. In this case, as it is discussed in the previous section, the dual offset radius $R$ should be properly reduced. As for the stop condition, it is integrated in the offset method used.

With the above discussion, all the offset tool paths can be generated. Except for the first tool paths, the other tool paths are smooth without sharp angles. The output of the tool path smoothing process is a collection of offset tool paths $\{p_i \mid i \in [1, n]\}$, in which $n$ is the path number.

As it can be seen in Fig. 6, the disadvantage of the above tool path smoothing method is that many uncut regions will be left in the corner areas. These uncut regions must be detected and then cleared.

## 4 Uncut region removal

As shown in Fig. 6, the six uncut regions near the pocket boundary are left because of the cutter size. In this work, these boundary uncut regions are not discussed, because they cannot be cleared unless a cutter with a smaller size is used. For the rest uncut regions inside the pocket, they must be cleared.

In fact, the uncut problem is common in contour-parallel offset tool path. Choi and Kim [31] divided the uncut regions into three types: corner uncut, center uncut, and neck uncut, as shown in Fig. 7. Park and Choi [32] appended clear-up tool paths to the nearest offset curve after the uncut regions are detected through observation results. Mansor et al. [33] proposed a Voronoi diagram-based uncut region removal algorithm. In our previous work in [34], an efficient uncut region removal method is developed, inspired by the idea to link all the uncut regions into a clear-up tool path in a global point of view. In that work, uncut regions are detected through geometry analysis. The problem is, for complex pockets, the geometry analysis method may fail. In the following, a more generic and robust uncut region detection method based on Boolean operation is proposed.

### 4.1 Uncut region detection

Let us take a neck uncut region to illustrate, as shown in Fig. 8. In this example, $p_i$ and $p_{i+1}$ are two successive tool paths. The path interval $\omega$ is greater than the cutter radius $\rho$, but smaller than the cutter diameter $2\rho$. Because of the peanut-like shape of $p_i$, path $p_{i+1}$ is splitted, leaving a neck uncut region in the middle area. To identify the position and shape of the uncut region, it is first necessary to find out the cutting edge trajectories when the cutter cuts along the two tool paths.

This can be realized again by an offset method. More specifically, the cutting edge trajectory of $p_i$ can be generated by offsetting $p_i$ inwards, at a distance $\rho$, and the cutting edge trajectory of $p_{i+1}$ can be generated by offsetting $p_{i+1}$ outward, at a distance $-\rho$. The obtained trajectories are donated by $e_i$ and $e_{i+1}$, respectively. It must be emphasized that in the offset

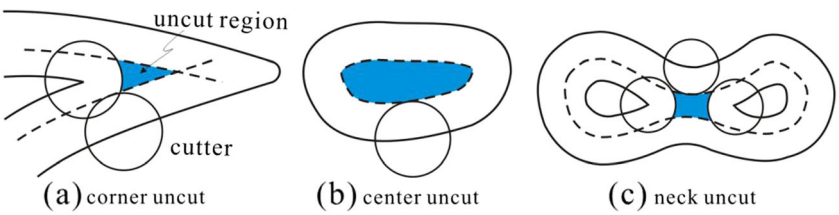


**Fig. 7** Three types of uncut regions

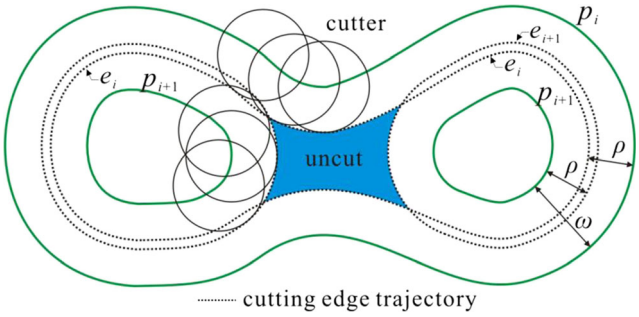


**Fig. 8** Neck uncut region detection

method used here, the arc filling option for superior corners must be selected to reflect the real cutting situation.

It can be seen from Fig. 8 that the uncut region is actually enclosed by $e_i$ and $e_{i+1}$. In computational geometry, this enclosed region can be solved with a Boolean operation. Let us denote uncut region between path $p_i$ and $p_{i+1}$ by $u_i$, and then $u_i$ can be solved as

$$u_i = e_i - e_{i+1}, \tag{2}$$

in which "–" is the Boolean subtraction operator. If the obtained $u_i$ is not null, then there are uncut regions between $p_i$ and $p_{i+1}$; otherwise, there is no uncut region. In this way, the neck uncut region can be detected and located. As Boolean operation for planar curves is a very mature technology, the above uncut region detection method shall be robust.

The same method can be used to detect the corner uncut regions, as shown in Fig. 9a. For the center uncut region, as shown in Fig. 9b, $p_i$ is the last tool path and $e_i$ is the corresponding cutting edge trajectory. In this case, the uncut region $u_i$ can be directly represented by $e_i$, namely, $u_i = e_i$. In fact, Eq. (2) can also be used to calculate the uncut region for the last tool path, except that at this time, the cutting edge trajectory $e_{i+1}$ is null. It must be emphasized again that here, $u_i$ is a curve set, which may include more than one curves, as shown in Fig. 9a.

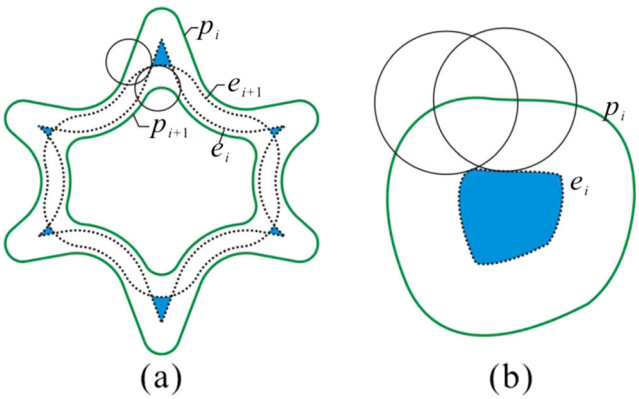


**Fig. 9** Corner and center uncut region detection

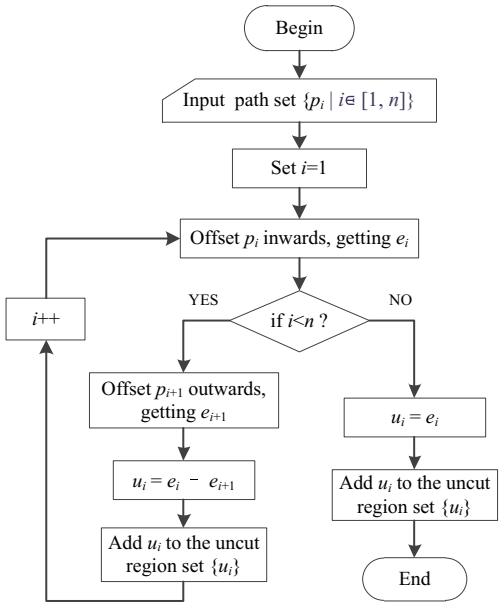


**Fig. 10** Flow chart of the uncut region detection process

With the above discussion, the uncut region detection flow chart for all the three types of uncut region is illustrated in Fig. 10. The input of the detection process is a collection of tool paths {$p_i$} obtained in the previous section. And the output is a collection of uncut regions {$u_i$}. These uncut regions need to be cleared with some uncut tool paths.

### 4.2 Clear-up tool path generation

Theoretically, when the cutter moves along the boundary curve of an uncut region, the residual material will be cleared. However, as it can be seen from Fig. 6, there might be many uncut regions between the offset tool paths, and on the uncut boundary curves, there might be many inferior sharp corners. These corners should also be rounded for HSM purpose.

Intuitively, the dual offset procedure can be used to round the corners. However, as the sizes of the uncut regions are small, the determination of the inward or outward offset

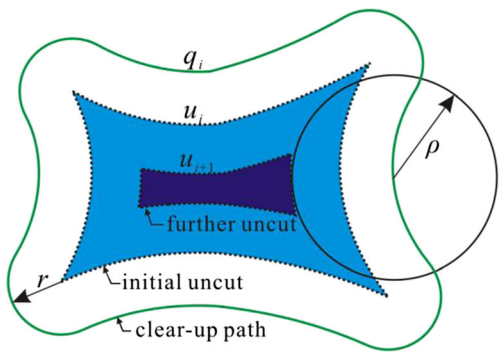


**Fig. 11** Clear-up tool path generation for uncut regions

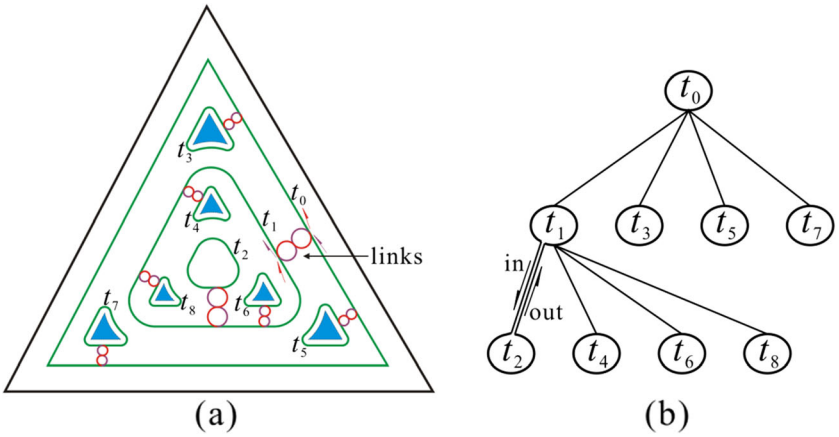


**Fig. 12** Tool path linking: **a** offset and clear-up tool paths; **b** tool path inclusion tree

distance in the dual offset procedure might be tedious. In this work, these sharp corners are simply rounded by offsetting the uncut boundary curves outwards for a proper distance $r$, as shown in Fig. 11. The offset distance $r$ is also set according to the cutter radius $\rho$, namely,

$$r = -k_o \cdot \rho \tag{3}$$

in which $k_o$ is the outward offset coefficient in a range of (0, 1). The obtained offset curve $q_i$ is used an uncut region clear-up tool path. The clear-up tool path should be smooth if the arc filling option is selected for the outward offset process.

As the clear-up tool path $q_i$ is bigger than the uncut boundary curve $u_i$, after the cutter finishes tracking $q_i$, an extra uncut region might be left, as shown in Fig. 11. It is necessary to further generate the inward cutting trajectory of the clear-up tool path. If the trajectory is not null, then an extra uncut region exist. In this case, this trajectory is used as a new uncut boundary curve to generate a second clear-up tool path with the above outward offset method. This process repeats until there is no new inward cutting trajectory.

After each of the uncut boundary curves in $\{u_i\}$ is traversed, a collection of clear-up tool paths $\{q_i\}$ can be generated. With the offset tool paths $\{p_i\}$ and clear-up tool paths $\{q_i\}$, all the material inside the pocket can be smoothly cleared.

## 5 Tool path linking

The obtained smooth tool paths, including the offset tool paths $\{p_i\}$ and clear-up tool paths $\{q_i\}$, are added one by one to a

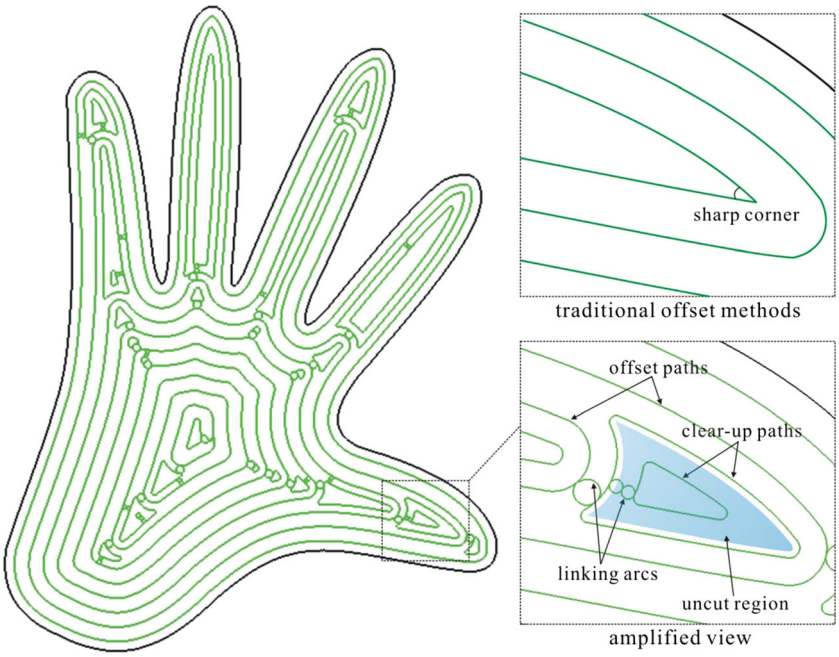


**Fig. 13** Smooth tool path in a hand-shape pocket

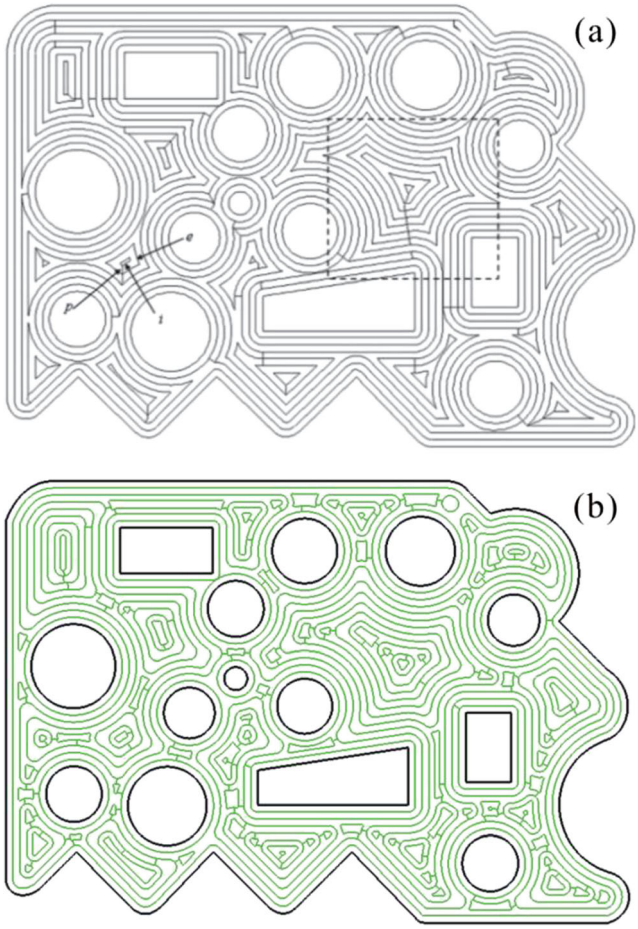


**Fig. 14** A pocket with 14 islands inside: **a** tool path by Voronoi diagram; **b** tool path by our method

new tool path collection $\{t_i\}$. Note that in $\{t_i\}$, each element is a single closed curve. The curves in $\{t_i\}$ can be organized in to a tree according to the inclusion tests. Let us take the tool paths in Fig.12a for example. There are nine tool paths in Fig.12, in which $t_0$, $t_1$, and $t_2$ are offset tool paths and $t_3$, $t_4$, …, $t_8$ are clear-up tool paths. As $t_0$ is in an outermost position, it becomes the root node of the tree. Inside $t_0$, there are four immediate paths, which are $t_1$, $t_3$, $t_5$, and $t_7$. They become the leaf nodes of $t_0$. Similarly, tool paths $t_2$, $t_4$, $t_6$, and $t_8$ become the leaf nodes of $t_1$. With these relations, the corresponding tree can be obtained and is also shown in Fig. 12b.

Based on the tree, the leaf curves are connected to the corresponding father nodes one by one from inner to outer, generating a linked tool path. For example, in Fig. 12, tool paths $t_2$, $t_4$, $t_6$, and $t_8$ are first connected to $t_1$, generating a new unified tool path $t_1$. Each connection is realized with two segments, one segment in and the other segment out, as the details shown in Fig. 12b. After that, the new $t_1$, together with $t_3$, $t_4$, and $t_5$, is connected to $t_0$, generating the final tool path. The details of the linking algorithm are similar to that introduced in [19]; the difference is that we have also added the closed clear-up tool paths into to tool path set. In order to smooth the tool paths on the linking joints, pairs of "S"-like tangential linking arcs can be used to replace the link-in and link-out segments, as shown in Fig. 12a. The calculation of these arcs involves basic geometry deduction and is discussed in Appendix A.

## 6 Examples

The proposed smooth tool path generation method is implemented in C++ programming language. In the test program, the Clipper library as introduced in the first section is employed to do both planar curve offset and Boolean operations. For superior corners, Clipper offers three gap filling options in its offset algorithm: miter, square, and round. For the miter option, the two offset segments simply extend to the intersection; for the square and round option, the offset segments are joined with a trapezoid and an arc, respectively, which is similar to what we have discussed in Section 2.1. So, the offset algorithm of Clipper is adequate for the proposed dual offset procedure. In the test program, the dual offset coefficient as in Eq. (1) is set to be 1.1, and the outward offset coefficient as in Eq. (3) is set to be 0.2. These parameters can be adjusted if necessary. Several examples are tested in the program.

The first example shows a hand-shape pocket in Fig. 13. The hand is about 150 mm in width and 200 mm in height. In this example, a cutter with a radius of 3 mm is used to cut out the pocket. The path interval $\omega$ is set to be $1.5\rho$, namely, 4.5 mm. The obtained smooth tool path is shown in Fig. 13. Since a hand has five fingers, let us take a detailed look at the tool paths in the thumb area, as shown in the amplified view of Fig. 13. As the finger is slim, with traditional offset methods, very sharp corners will be generated. However, with the proposed dual offset method, the sharp corners are trimmed and rounded with transition arcs. Nevertheless, as the finger is slim, long uncut region is left in the corner of the two offset paths. This uncut region is detected through the proposed Boolean operation. Based on the uncut region, clear-up tool paths are generated. For this case, as the uncut region is large, two successive clear-up tool paths are needed to remove the residual material, as shown in Fig. 13. The offset and the clear-up tool paths are finally connected into one by a set of linking arcs.

In the second example, we have compared the tool path generated by our method with that by the Voronoi diagram method. The pocket example as shown in Fig. 14 is first used by Hinduja et al. in [20]. It is a $2^1/_2$D pocket with 14 protrusions, the cross-sectional shapes of which are trapezoidal, rectangular, and circular. The width of the pocket is about 270 mm and height about 190 mm. As in [20], the cutter radius and the path interval are both 3 mm; in our test program, the two parameters are also set to be 3 mm.

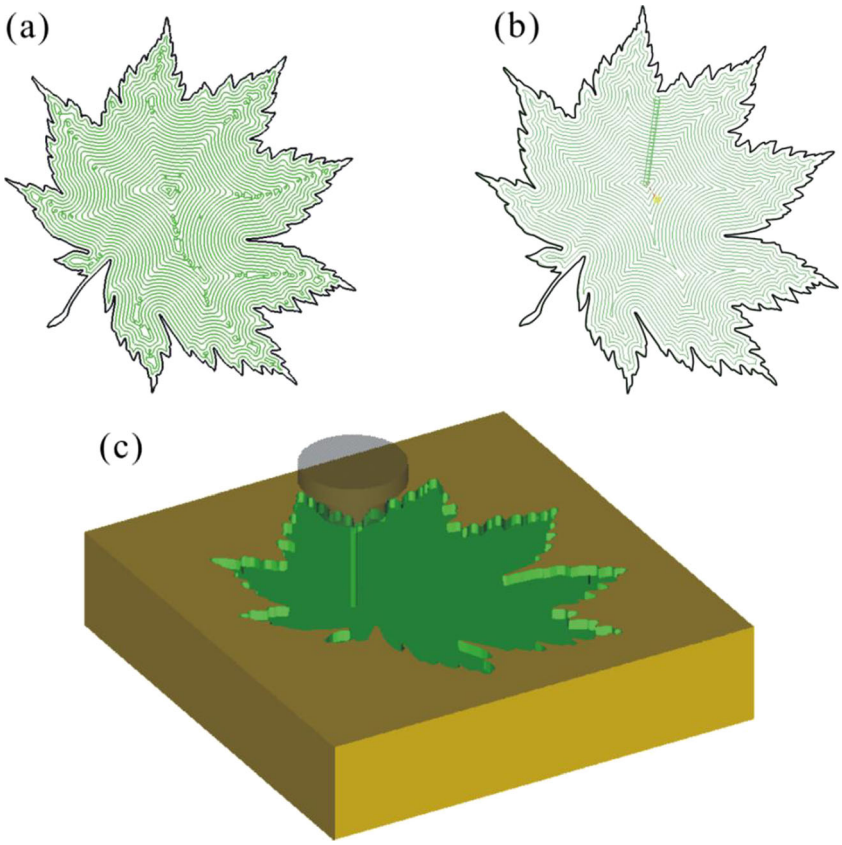


**Fig. 15** Cutting simulation comparisons of a leaf pocket in VERICUT: **a** our tool path; **b** tool path by PowerMILL; **c** finished workpiece

Figure 14a shows the tool path generated by the Voronoi diagram method, and Fig. 14b shows the tool path generated by our method. In Fig. 14a, although the superior sharp corners from the boundaries (including islands) have been rounded with arcs, the inferior sharp corners emerge in the place where the offset curves split and they are inherited by the successors; as a result, there are many sharp corners on the obtained tool path. However, in Fig. 14b, all the sharp corners have been rounded. In [20], the offset tool paths are linked by line segments. To compare the tool path length, we have also linked the paths with line segments. For Fig. 14a, according to [20], the total tool path length is about 11,200 mm; while for Fig. 14b, this value is about 12,123 mm, 8 % longer than that of Fig. 14a. This is understandable: for the Voronoi diagram method, as the path interval is the same to the cutter radius, there is no uncut region; while for the dual offset method, many uncut regions are left in the rounded corner areas. Obviously, the clear-up tool paths has contributed to lengthen the tool path. However, as the tool path in Fig. 14b is much smoother than that in Fig. 14a, HSM can be applied.

For the third example, a series of comparative cutting simulations are performed on VERICUT. The purpose is to verify whether the proposed tool path generation method can improve the machining efficiency in case of HSM. Used in this example is a leaf-shape pocket with a dimension of 300× 320 mm$^2$, as shown in Fig. 15. This pocket is special because there are many sharp corners on the boundary. The cutter radius and the path interval are both set to be 3 mm. The pocket, together with the cutting parameters, is input separately into our test program and a well-known commercial CAM software PowerMILL. The generated tool paths are shown in Fig. 15a, b, respectively. The output NC codes (in G code form) are then imported into VERICUT. In VERICUT, a three-axis machine tool with a FANUC control system is set up to execute the NC codes. Figure 15c shows the finished workpiece in VERICUT with our tool path.

**Table 1** Machining time (min:s) of the leaf pocket under different feedrate

| Feedrate (mm/min) | 500 | 1000 | 2000 | 4000 | 6000 | 8000 | 10000 | 12000 |
|---|---|---|---|---|---|---|---|---|
| PowerMILL | 29:36 | 14:53 | 7:31 | 3:50 | 2:37 | 2:00 | 1:38 | 1:23 |
| Our program | 30:10 | 15:05 | 7:32 | 3:46 | 2:31 | 1:53 | 1:30 | 1:15 |
| Improved by | – | – | – | 1.7 % | 3.8 % | 5.8 % | 8.2 % | 9.6 % |

In the cutting simulations, the feedrate in the NC code files is overwritten from 500 mm/min to 12,000 mm/min. The corresponding machining time is recorded in Table 1. From Table 1, it can be seen that when the feedrate is relatively low, for example, 500, 1000, or 2000 mm/min, the machining time of the tool path generated by PowerMILL is shorter than that of our program. This is because our tool path has many clear-ups in the corners. In this stage, apparently, the tool path length is the main factor to influence the machining efficiency. However, when the feedrate grows higher than 4000 min/min, the smooth transition arcs on our tool path take advantage. As a result, the machining time of our tool path is shorter than that of PowerMILL. In this stage, obviously, the smoothness of the tool path is the main factor to influence machining efficiency. It can also be seen from Table 1 that the efficiency improvement grows with the feedrate by comparing the two tool paths.

It has to be emphasized that in these cutting simulations, the machining time is our major concern. Other factors such as spindle speed, cutting force or interpolation errors in HSM are not concerned presently. It is assumed that the machine tool and the control system are capable to handle these things. Above all, these simulations suggest that the proposed tool path generation method has a potential to improve the machining efficiency in terms of HSM.

## 7 Conclusions

The contour-parallel tool path planning strategy is very popular in CAM systems. To generate such a tool path, it is necessary to offset the input boundaries successively. However, it seems that existing offset methods both in the literatures and in commercial CAM systems cannot solve the tool path smoothing problem thoroughly. As a result, many sharp corners are left untreated on the final tool path. Machining with conventional speed might be able to endure these corners. However, for high-speed machining, sharp corners should better be smoothed.

Considering that it is nearly impossible to manually put inscribed arcs to the sharp corners, in this work, a dual offset procedure is introduced. For a given path, the dual offset procedure first offset the path inwards, generating an intermediate curve; then, this intermediate curve is further offset outwards, generating a smooth curve. This smooth curve is used to replace the given path. Note that for the second offset, the arc filling option must be selected. One side effect of the dual offset procedure is that many uncut regions will be left near the rounded corner areas. These uncut regions are detected through a Boolean subtraction operation and cleared by smooth offset curves. The offset and clear-up tool paths are connected into one with a set of linking arcs. The finally obtained tool path is smooth and suitable for HSM.

The proposed smooth tool path generation method is implemented and tested with several pocket examples. Besides, several comparative cutting simulations are performed on VERI CUT. The simulations show that in terms of HSM, the proposed method has a potential to improve the machining efficiency.

Though, there are still several problems need to be solved in the further work. Now most of the available offset methods (including the Clipper library we used) can only deal with planar polygon offset, and the output offset curve is also a polygon. That is to say, the gap filling arcs on the superior corners are actually discrete short line segments. At present, we have tried to recognize the potential arcs from line segments on the obtained tool path. But the result is not satisfying. Further effort should be addressed on this aspect. Another problem is with linking arcs used for connecting adjacent tool paths. We have observed that when two tool paths are close, very small linking arcs will be resulted. These small arcs are harmful to HSM due to the high curvatures. A more robust linking arc generation method should be developed.

**Acknowledgment** This work was financially supported by Science Fund for Creative Research Groups of National Natural Science Foundation of China (No. 51221004), the National Nature Science Foundation of China (no. 51175461) and Specialized Research Fund for the Doctoral Program of Higher Education (no. 20120101110055).

## Appendix A. Tangential linking arcs calculation

Let us take a link-in line segment $P_1P_2$ for example, as shown in Fig. 16. $P_0P_1$ and $P_2P_3$ are the two incident segments on the outer and inner offset tool paths, respectively. Suppose the two offset tool paths are both in counterclockwise direction. The task now is to find a pair of arcs that smoothly connect $P_0P_1$ and $P_2P_3$ on the connecting points $P_1$ and $P_2$.

Let us denote the direction vectors of $P_0P_1$ and $P_2P_3$ by $\mathbf{v_1}$ and $\mathbf{v_2}$, respectively. Based on $\mathbf{v_1}$ and $\mathbf{v_2}$, the unit normal

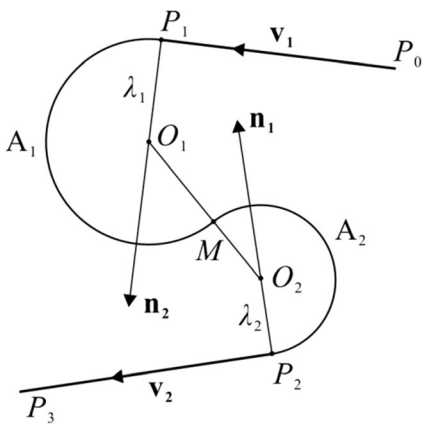


**Fig. 16** Tangential linking arcs modeling and calculation

vectors of $P_0P_1$ and $P_2P_3$ can be calculated and they are denoted by $\mathbf{n_1}$ and $\mathbf{n_2}$, respectively. It must be noted that $\mathbf{n_1}$ is on the left side of $\mathbf{v_1}$ and $\mathbf{n_2}$ is on the right side $\mathbf{v_2}$. If the directions are opposite, they need to be adjusted. Let us denote the pair of arcs by $A_1$ and $A_2$ and the center points of the two arcs by $O_1$ and $O_2$, respectively. $A_1$ connects $P_1$ and $A_2$ connects $P_2$, as shown in Fig. 16.

As arcs $A_1$ and $A_2$ are tangentially connected to $P_0P_1$ and $P_2P_3$, the center points $O_1$ and $O_2$ should be on the normal of $\mathbf{v_1}$ and $\mathbf{v_2}$, namely

$$O_1 = P_1 + \lambda_1 \cdot \mathbf{n_1}, \tag{A1}$$

$$O_2 = P_2 + \lambda_2 \cdot \mathbf{n_2}, \tag{A2}$$

in which $\lambda_1$ and $\lambda_2$ are the radii of $A_1$ and $A_2$, respectively. With Eq. (A1) and (A2), the length of $O_1O_2$ can be represented as

$$\|\mathbf{O_1O_2}\| = \|\mathbf{P_1P_2} + \lambda_2 \cdot \mathbf{n_2} - \lambda_1 \cdot \mathbf{n_1}\| \tag{A3}$$

On the other hand, as arcs $A_1$ and $A_2$ are tangentially connected, the length of $O_1O_2$ is the addition of the two radii, namely $(\lambda_1+\lambda_2)$. So we have the following equation

$$\|\mathbf{P_1P_2} + \lambda_2 \cdot \mathbf{n_2} - \lambda_1 \cdot \mathbf{n_1}\| = \lambda_1 + \lambda_2 \tag{A4}$$

Eq. (A4) can be expanded and simplified as

$$2(1 + \mathbf{n_1} \cdot \mathbf{n_2})\lambda_1\lambda_2 + 2(\mathbf{P_1P_2} \cdot \mathbf{n_1})\lambda_1 - 2(\mathbf{P_1P_2} \cdot \mathbf{n_2})\lambda_2 - \|\mathbf{P_1P_2}\|^2 = 0 \tag{A5}$$

Eq. (A5) is an indeterminate equation with two unknowns $\lambda_1$ and $\lambda_2$. It cannot be solved unless another relation of $\lambda_1$ and $\lambda_2$ is found. In this work, we simply set $\lambda_1=\lambda_2=\lambda$, in which $\lambda$ is the common arc radius, then Eq. (A5) can be reduced to

$$(1 + \mathbf{n_1} \cdot \mathbf{n_2})\lambda^2 + (\mathbf{P_1P_2} \cdot \mathbf{n_1} - \mathbf{P_1P_2} \cdot \mathbf{n_2})\lambda - \frac{\|\mathbf{P_1P_2}\|^2}{2} = 0 \tag{A6}$$

With Vieta's formulas, the two roots of Eq. (A6) can be represented as

$$\lambda_{a,b} = \frac{-(\mathbf{P_1P_2} \cdot \mathbf{n_1} - \mathbf{P_1P_2} \cdot \mathbf{n_2}) \pm \sqrt{(\mathbf{P_1P_2} \cdot \mathbf{n_1} - \mathbf{P_1P_2} \cdot \mathbf{n_2})^2 + 2(1 + \mathbf{n_1} \cdot \mathbf{n_2}) \cdot \|\mathbf{P_1P_2}\|^2}}{2(1 + \mathbf{n_1} \cdot \mathbf{n_2})} \tag{A7}$$

For the above equation, since $\mathbf{n}_1$ and $\mathbf{n}_2$ are unit vectors, $\mathbf{n}_1 \cdot \mathbf{n}_2$ must be in a range of [−1, 1], namely $(1+ \mathbf{n}_1 \cdot \mathbf{n}_2) \geq 0$. So the expression under the square root sign must be positive or zero, namely, $\lambda_a$ and $\lambda_b$ are two real numbers.

The selection of roots $\lambda_a$ and $\lambda_b$ for the common arc radius $\lambda$ is based on experience. In our implementation, the root which has a smaller absolute value is used for $\lambda$, namly

$$\lambda = \begin{cases} \lambda_a, & \|\lambda_a\| \leq \|\lambda_b\| \\ \lambda_b, & \text{else} \end{cases} \tag{A8}$$

The obtained $\lambda$ could possibly be negative. In this case, as in Eq. (A1) and (A2), arc center points $O_1$ and $O_2$ are on the reverse direction of $\mathbf{n_1}$ and $\mathbf{n_2}$, respectively.

Let us denote the common tangential point of arcs $A_1$ and $A_2$ by $M$. $M$ is actually the middle point of segment $O_1O_2$. With the start point $P_1$, end point $M$, center point $O_1$ and radius $\lambda$, arc $A_1$ can be determined. Similarly, arc $A_2$ can also be determined. As for the arc directions, if $\lambda > 0$, then $A_1$ is counterclockwise and $A_2$ is clockwise; otherwise, $A_1$ is clockwise and $A_2$ is counterclockwise. The calculated two arcs $A_1$ and $A_2$ can be used to replace line segment $P_1P_2$. In this way, smooth linking of adjacent tool paths can be realized.

## References


1. Park SC, Choi BK (2000) Tool-path planning for direction-parallel area milling. Comput Aided Des 32(1):17–25
2. Persson H (1978) NC machining of arbitrarily shaped pockets. Comput Aided Des 10(3):169–174
3. Held M, Lukács G, Andor L (1994) Pocket machining based on contour-parallel tool paths generated by means of proximity maps. Comput Aided Des 26(3):189–203
4. Held M (2001) VRONI: an engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments. Comput Geom 18(2):95–123
5. Kim H-C (2010) Tool path generation for contour parallel milling with incomplete mesh model. Int J Adv Manuf Technol 48(5):443–454
6. Choi BK, Park SC (1999) A pair-wise offset algorithm for 2D point-sequence curve. Comput Aided Des 31(12):735–745
7. Shen HY, Fu JZ, Chen ZC, Fan YQ (2010) Generation of offset surface for tool path in NC machining through level set methods. Int J Adv Manuf Technol 46(9–12):1043–1047
8. Kimmel R, Bruckstein AM (1993) Shape offsets via level sets. Comput Aided Des 25(3):154–162
9. Kim K, Jeong J (1995) Tool path generation for machining free-form pockets with islands. Comput Ind Eng 28(2):399–407
10. Liu X-Z, Yong J-H, Zheng G-Q, Sun J-G (2007) An offset algorithm for polyline curves. Comput Ind 58(3):240–254
11. Lai Y-L, Wu JS-S, Hung J-P, Chen J-H (2006) A simple method for invalid loops removal of planar offset curves. Int J Adv Manuf Technol 27(11):1153–1162
12. Wong TN, Wong KW (1996) NC toolpath generation for arbitrary pockets with islands. Int J Adv Manuf Technol 12(3):174–179

13. Kim H-C, Lee S-G, Yang M-Y (2006) A new offset algorithm for closed 2D lines with Islands. International Journal of Advanced Manufacturing Technology 29(Compendex):1169–1177
14. Lee C-S, Phan T-T, Kim D-S (2009) 2D curve offset algorithm for pockets with islands using a vertex offset. Int J Precis Eng Manuf 10(2):127–135
15. Lin Z, Fu J, He Y, Gan W (2013) A robust 2D point-sequence curve offset algorithm with multiple islands for contour-parallel tool path. Comput Aided Des 45(3):657–670
16. Felkel P, Obdrzalek S (1998) Straight Skeleton Implementation. Proceedings of Spring Conference on Computer Graphics:210–218
17. Dhanik S, Xirouchakis P (2010) Contour parallel milling tool path generation for arbitrary pocket shape using a fast marching method. Int J Adv Manuf Technol 50(9–12):1101–1111
18. Park SC, Chung YC (2002) Offset tool-path linking for pocket machining. Comput Aided Des 34(4):299–308
19. Park SC, Chung YC, Choi BK (2003) Contour-parallel offset machining without tool-retractions. Comput Aided Des 35(9):841–849
20. Hinduja S, Mansor MSA, Owodunni OO (2010) Voronoi-diagram-based linking of contour-parallel tool paths for two-and-a-half-dimensional closed-pocket machining. Proc Inst Mech Eng B J Eng Manuf 224(B9):1329–1350
21. Dai N, Dong G, Liao W, Sun Y (2012) Dental restoration contour-parallel offset tool path links based on graph model. The International Journal of Advanced Manufacturing Technology:1–9
22. Fallböhmer P, Rodríguez CA, Özel T, Altan T (2000) High-speed machining of cast iron and alloy steels for die and mold manufacturing. J Mater Process Technol 98(1):104–115
23. Wang ZH, Yuan JT, Liu TT, Huang J, Qiao L (2015) Study on surface roughness in high-speed milling of AlMn1Cu using factorial design and partial least square regression. Int J Adv Manuf Technol 76(9–12):1783–1792
24. Monreal M, Rodriguez CA (2003) Influence of tool path strategy on the cycle time of high-speed milling. Comput Aided Des 35(4): 395–401
25. Siller H, Rodriguez CA, Ahuett H (2006) Cycle time prediction in high-speed milling operations for sculptured surface finishing. J Mater Process Technol 174(1–3):355–362
26. Choy HS, Chan KW (2003) A corner-looping based tool path for pocket milling. Comput Aided Des 35(2):155–166
27. Zhao ZY, Wang CY, Zhou HM, Qin Z (2007) Pocketing toolpath optimization for sharp corners. J Mater Process Technol 192–193: 175–180
28. Shi J, Bi Q, Zhu L, Wang Y (2015) Corner rounding of linear five-axis tool path by dual PH curves blending. Int J Mach Tools Manuf 88:223–236
29. Pateloup V, Duc E, Ray P (2004) Corner optimization for pocket machining. Int J Mach Tools Manuf 44(12):1343–1353
30. Pateloup V, Duc E, Ray P (2010) Bspline approximation of circle arc and straight line for pocket machining. Comput Aided Des 42(9):817–827
31. Choi BK, Kim BH (1997) Die-cavity pocketing via cutting simulation. Comput Aided Des 29(12):837–846
32. Park SC, Choi BK (2001) Uncut free pocketing tool-paths generation using pair-wise offset algorithm. Comput Aided Des 33(10): 739–746
33. Mansor MSA, Hinduja S, Owodunni OO (2006) Voronoi diagram-based tool path compensations for removing uncut material in 2½D pocket machining. Comput Aided Des 38(3):194–209
34. Lin Z, Fu J, Shen H, Gan W (2013) Global uncut regions removal for efficient contour-parallel milling. Int J Adv Manuf Technol 68(5–8):1241–1252