Int J Adv Manuf Technol (2001) 17:181–188

The International Journal of
Advanced
Manufacturing
Technology

Robust Spiral Tool-Path Generation for Arbitrary Pockets

Chun-Fong You, Bor-Tyng Sheen and Tzu-Kuan Lin

Department of Mechanical Engineering, National Taiwan University, Taipei, Taiwan

When converting CC data (cutter contact point data) into CL data (cutter location data) for tool-path generation for arbitrary pockets, overcut avoidance plays an important role in CNC pocket milling. Complicated calculations of the self-intersections in the offset of spiral cutting is one of the main considerations in some algorithms. This study presents a quasi-offset method to solve complicated self-intersection calculations. Instead of using offsetting methods, the proposed method uses the location points and their track types to generate the son loop. The proposed quasi-offset method also makes it much easier to generate spiral tool paths.

Keywords: Arbitrary pocket; Gouging; Path generation; Quasi-offset; Sprial tool path

1. Introduction

Pockets are among the most common 2D CNC milling features. Pockets are defined by strings of entities that outline their boundaries, which together with their corresponding depths, also indicate the level at which the tool should cut. Pockets can be machined using different types of tool path, such as spiral or zigzag patterns. Zigzag tool paths are easier to generate than spirals, but repetitious zigzag milling cuts cause changes of cutting force and stress on the tool. However, only one of the up or down milling effects occurs in the cutting processes if a spiral cut is used. Therefore, spiral cuts have better machining conditions and so have an important status in milling even though they are more difficult to construct.

The difficulty in generating spiral tool paths arises from the complicated intersection calculations of the offset entities, especially for rough milling or if the iteration of the offset is performed many times. The offset entities become closer and produce complicated self-intersections under these conditions.

Generally speaking, the contour of a pocket should form a non-intersecting and a closed loop. This contour loop is used to produce the first loop of a tool path when creating spiral tool paths, as illustrated in Fig. 1. This contour loop is called the mother loop and the trimmed loop is called the son loop. As the iteration of offset paths continues, the son loop will be used as a mother loop in order to generate the next son loop. This will be repeated until the inner part of the pocket is thoroughly removed.

Although the generation procedure is simple, the offset calculation is complicated. The son loop must finally be trimmed to be a simple closed loop because no self-intersection among entities is permitted in the cutting path. Hence, the first step of the calculation is the self-intersection discrimination of the son loop. Procedures completed in this step include the following: solving intersection points; locating the points that lie upon the entities; and breaking crossed entities into smaller ones. The second step in the calculation is the trimming of the overcut entities. Caution must be exercised here, as too many entities to be trimmed will result in an undercut, whereas too few entities will result in an overcut. Tiller and Hansen [1] submitted several solutions to identify an overcut zone of the son loop. Suh and Lee [2] later proposed more detailed methods to deal with 2D profiles. As shown in Fig. 1, a direction is assigned to each offset entity. These directions form a clockwise loop. When the separation operations are carried out on the crossed entities, new smaller loops are formed and their loop directions are assigned as shown in Fig. 2. Any new loop with a direction opposed to the original loop direction will be regarded as an overcut and must be removed.

Hansen [3] and Hansen and Arbab [4] developed interference indices to detect overcuts by assigning to every entity an interference index. Only entities with zero numbers survived and formed a son loop. Although Tiller's and Hansen's methods are useful in discriminating the overcut zone, they do not separate the multiply crossed entities of the raw son loop. The complicated self-intersection calculation of the raw son loop is inevitable in the process of rough milling or during the iterations of the offset calculation, as illustrated in Fig. 3. It is difficult to see how to split multiply crossed entities because the geometric relationships of its entities because, unreliable algorithms are used to trim the overcut entities.

Methods developed by Hoschek [5] to attack cusp and gouging could not prevent all overcuts or undercuts. Although many researchers [6–9] used Voronoi diagrams to generate CNC milling tool paths, most of them handle only simple pocket contours with a limited amount of offset.

Correspondence and offprint requests to: Dr Chun-Fong You, Department of Mechanical Engineering, National Taiwan University, Taipei, Taiwan. E-mail: you@w3.me.ntu.edu.tw

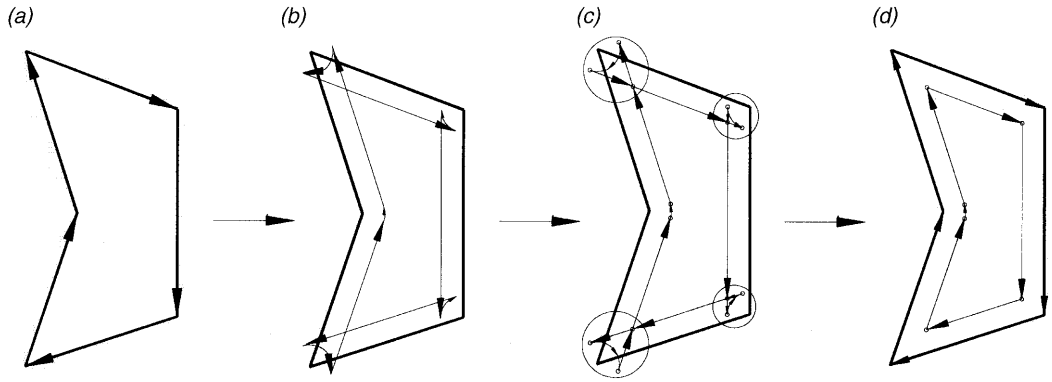

Fig. 1. Generation of the first son loop. (*a*) Contour. (*b*) Offset. (*c*) Discrimination and trim. (*d*) The first son loop.

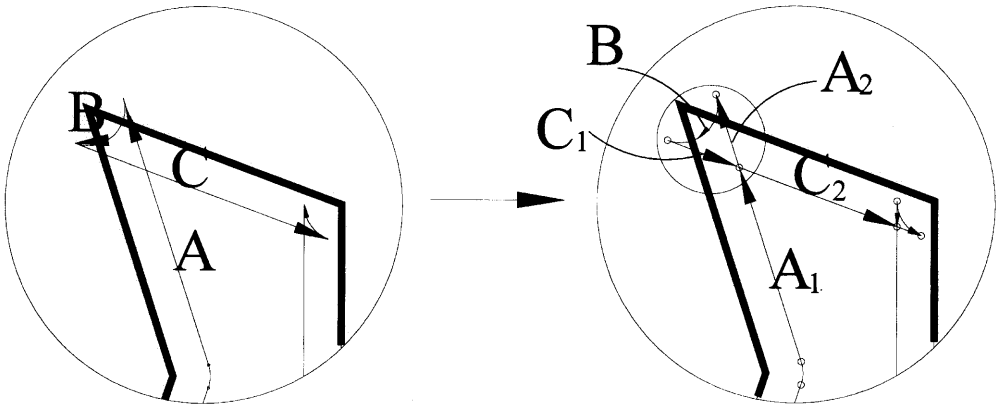

Fig. 2. Discrimination of overcut entities of raw son loop.

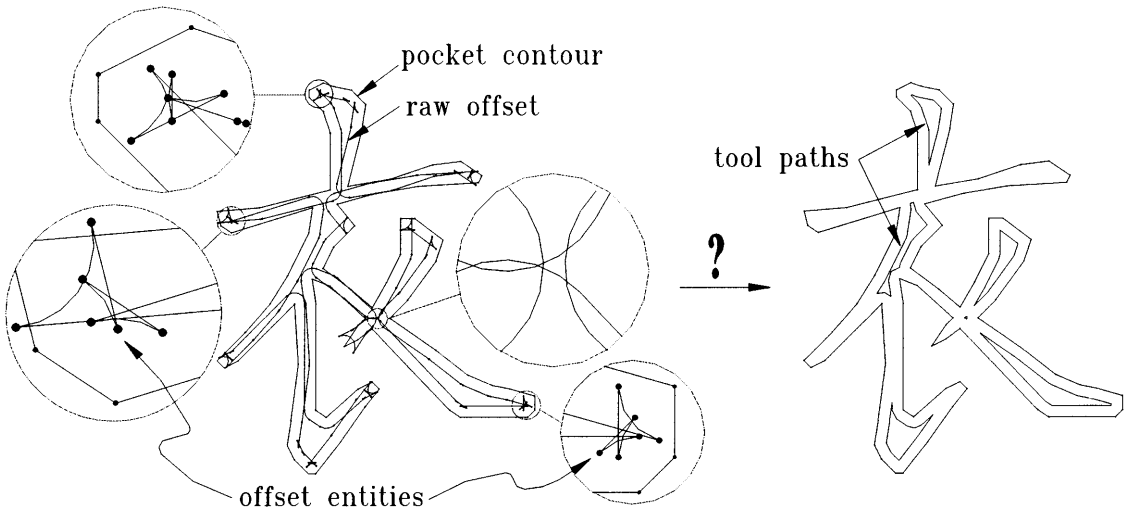

Fig. 3. Complicated self-intersections of raw son loop.

This paper employs a quasi-offset method to generate spiral tool paths for CNC milling, to avoid the complicated self-intersections of a raw son loop. The quasi-offset method reduces and simplifies the existing self-intersection calculations. In addition, the non-intersection son loop is more easily obtained, as illustrated in Fig. 4.

A pocket with islands is not discussed here because the contour of pocket and island can be connected to form a new contour using contour bridges [10]. The newly formed contour is more complicated than the original one and would require a robust algorithm of tool-path generation.

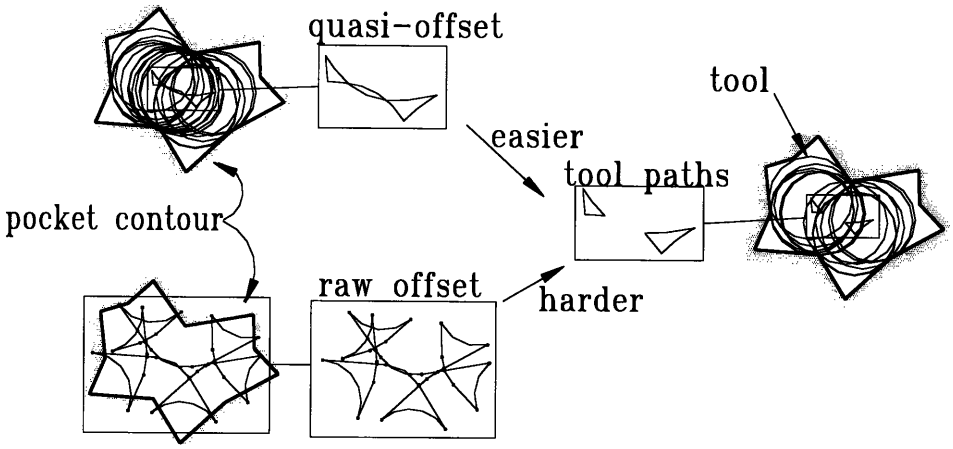

Fig. 4. The advantage of quasi-offset.

2. The Quasi-Offset Tool Path Generation Process

A quasi-offset is like a semi-son-loop that consists of location points (LPs) and their track types. A flowchart of the quasi-offset generation process is presented in Fig. 5. Quasi-offsets can allow several self-intersections, which is different from the requirement of the tool path. However, the quasi-offset itself becomes a trimmed son loop or the final tool path if there is no self-intersection in the quasi-offset. The advantage of quasi-offsets is the elimination of local gouging and cusps which are frequently encountered in the raw offset.

2.1 Preparation of the Mother Loop Entities

A pocket contour is used as a mother loop to generate a quasi-offset. The co-linear or concentric arc in the contour should be eliminated, and the mother loop entities must be arranged in a chain and numbered sequentially in advance.

2.2 Determination of the LPs

Two entities of a mother loop must be determined to determine the location point. The point must be located to avoid overcuts and unnecessary undercuts in the two given mother loop entities. The endpoints of the first prepared entity are named in sequence point 1 and point 2, and those of the second prepared entity are labelled point 3 and point 4. In this step, point 3 and point 4 may or may not be the same. Five types and tracks of LPs are presented in Fig. 6.

The location points can be found using vectors. Take location type 1 for example. As illustrated in Fig. 7, $\mathbf{E}_{1'2'}$ and $\mathbf{E}_{3'4'}$ are parallel to the first and second entity, respectively. Suppose the coordinates of points 1, 2, 3, and 4 are $(x_1\ y_1\ z_k)$, $(x_2\ y_2\ z_k)$, $(x_3\ y_3\ z_k)$, and $(x_4\ y_4\ z_k)$, respectively, d_{12} is the distance between points 1 and 2, d_{34} is the distance between points 3 and 4, r is the offset value and is equal to the cutter radius. Then, $\mathbf{E}_{1'2'} = \mathbf{P}_1 + \mathbf{E}_{11'} + u\mathbf{E}_{12}$, and $\mathbf{E}_{3'4'} = \mathbf{P}_4 + \mathbf{E}_{44'} + v\mathbf{E}_{43}$, where u and v are scalars, and

$$\mathbf{E}_{11'} = r[y_2 - y_1\ x_1 - x_2]/d_{12}$$

$$\mathbf{E}_{44'} = r[y_4 - y_3\ x_3 - x_4]/d_{34}$$

Because $\mathbf{E}_{1'2'}$ intersects $\mathbf{E}_{3'4'}$ at point A, $\mathbf{P}_A = \mathbf{P}_1 + \mathbf{E}_{11'} + u_0\mathbf{E}_{12} = \mathbf{P}_4 + \mathbf{E}_{44'} + v_0\mathbf{E}_{43}$. Hence, the point A or LP can be obtained by solving the vector equation. If $0 \le u_0 \le 1$ and $0 \le v_0 \le 1$, the point A is in the interior of the segments $E_{1'2'}$ and $E_{3'4'}$. The tool on the LP will contact tangentially the first and the second entities of the mother loop. The LPs of other types can be found in a similar way.

2.3 Overcut Detection and Elimination

Although there is a satisfactory cut, when the cutting tool is applied on this location point because no overcut or undercut occurs on the two prepared entities, it is still unknown whether it will cause an overcut on other entities. Testing every mother loop entity for overcuts is inefficient and unnecessary. This study employs adjacent entity checking to detect whether the tool on the LP will cause overcuts to any neighbouring entities to the two prepared entities. The tests also use rolling ball rules to check whether the LP path is reasonable or not.

Adjacent entity checking is performed using vectors. Figure 8 illustrates three basic overcut types of adjacent entity. In Figs 8(a) and 8(b), $\mathbf{E}_{n1}$ and $\mathbf{E}_{n2}$ are the vectors normal to line 1-LP and line 4-LP and $\hat{k}$ is the unit normal vector out of the paper. An overcut occurs if the condition $\mathbf{E}_b \times \mathbf{E}_{n1} \bullet \hat{k} < 0$ in Fig. 8(a) or the condition $\mathbf{E}_{n2} \times \mathbf{E}_f \bullet \hat{k} < 0$ in Fig. 8(b) is satisfied. The equation $u\mathbf{E}_{4f} = \mathbf{E}_{0i} - \mathbf{E}_{04}$ must be solved with a circular equation to obtain the value in Fig. 8(c). An overcut exist if $0 \le u \le 1$.

The current LP cannot be accepted and a new LP must be redefined when an overcut is detected. The first prepared entity E_{12} in Fig. 8(a) is substituted by its adjacent entity E_{s1}, or the second prepared entity E_{34} in Fig. 8(b) is substituted by its adjacent entity E_{4f} to locate a new LP.

The tool will not overcut the two prepared entities and their neighbouring ones if LP passes the adjacent entity checking. However, it is still unknown whether the tool on the LP will overcut any other entities. Rolling ball rules are introduced to

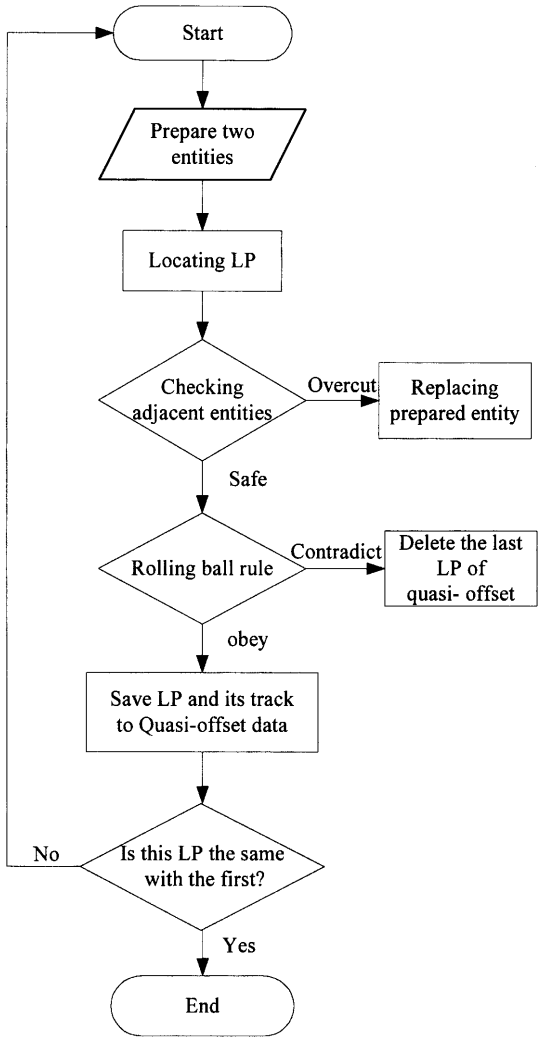

Fig. 5. The process of generating quasi-offset.

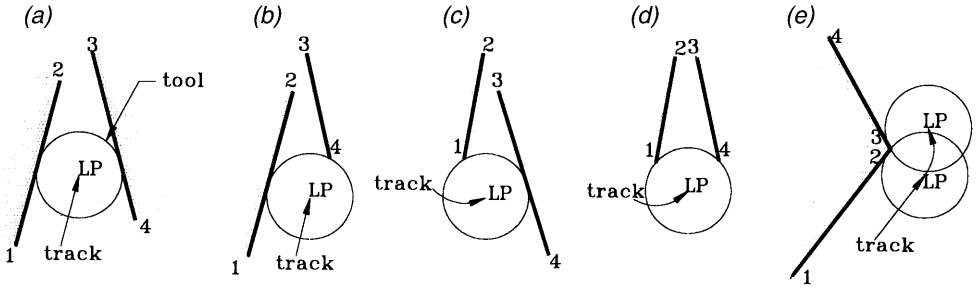

Fig. 6. LPs and their track types. (*a*) Line type 1. (*b*) Line type 2. (*c*) Arc type 1. (*d*) Arc type 2. (*e*) Line-arc type.

solve this problem. The rolling ball virtually travels along the pocket contour as illustrated in Fig. 9. The points *ABCDEF* are the endpoints of the contour and the points *PQRST* are the location points of the rolling ball path. The rolling ball path is actually the same with a quasi-offset. Rules to determine the path of the rolling ball are:

1. The angle between two connecting lines must be smaller than or equal to 180°, shown as θ_1 or θ_2 in Fig. 9.
2. The angle between two connecting arcs must be smaller than or equal to 90°, shown as θ_4 in Fig. 9.
3. The angle between a line and an arc connected to this line must be smaller than or equal to 180°, as shown by θ_3 or θ_5 in Fig. 9.

If a blind groove of a pocket is too narrow to be cut, no quasi-offset entity will be left within the groove. Local area gouging is detected using the adjacent entity checking and

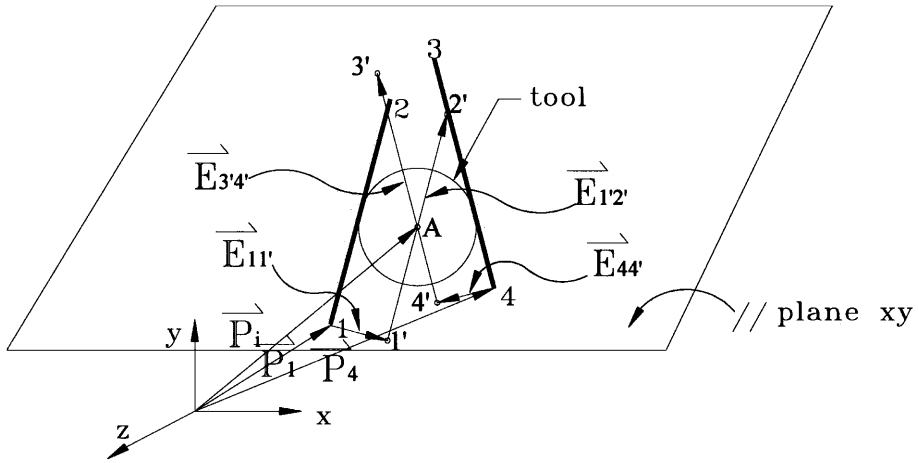

Fig. 7. Solving LP of line type 1.

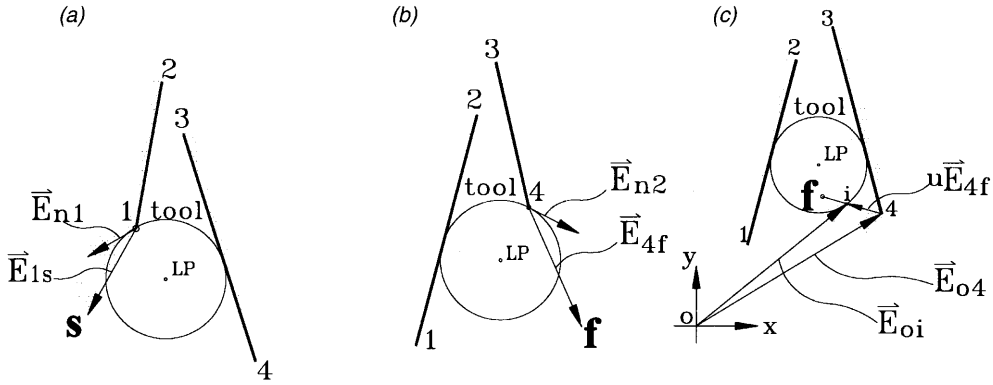

Fig. 8. Three basic overcuts of adjacent entities.

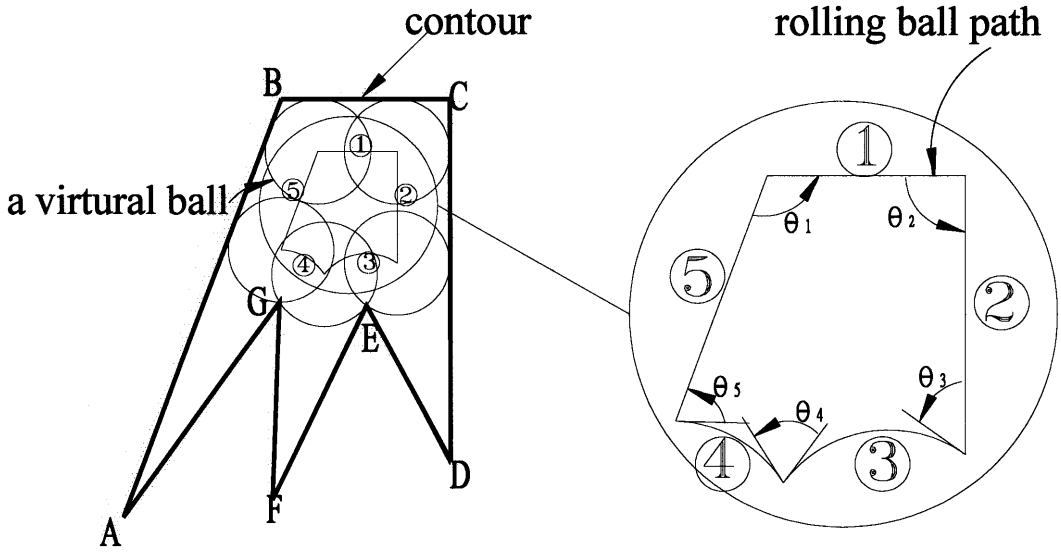

Fig. 9. Three rules between entities of a rolling ball path.

rolling ball rules. The elimination of gouging is shown in Fig. 10. The whole line entities 1 ~ 10 are a portion of the contour as shown in Fig. 10. The first LP is found from entities 1 and 2 (Fig. 10(*a*)). This LP must be discarded because it overcuts the adjacent entity as found by adjacent entity checking. A new LP that does not cause an overcut can be determined from entities 1 and 3 shown as LP_2 in Figs 10(*b*) and 10(*c*). In Fig. 10(*c*), the LP is created from entities 3 and 4 and shown as LP_3. In Fig. 10(*d*), the LP is constructed from entities 4 and 5, but this LP must be discarded because it overcuts the entity connecting to entity 5. Entity 5 is substituted by entity 6 in order to identify the new LP as illustrated in Fig. 10(*d*). The angle between LP_3 and this LP is equal to 360° and contradicts the second rule of the rolling ball. The LP and LP_3 must be deleted as presented in Fig. 10(*e*). Entity 4 is exchanged with entity 3 which is the first prepared entity of LP_3, used to determine a new LP, as displayed in Fig. 10(*f*). The angle between the LP and LP_2 is larger than 90° and contradicts the third rolling ball rule. The LP and LP_2 must be deleted. Entity 3 is exchanged with entity 1 which is the first prepared entity of LP_2 which is used to determine a new LP as shown in Fig. 10(*g*). Local gouging can be removed using the same method as presented in Figs 10(*g*), 10(*h*), and 10(*i*).

Although local area gouging and cusps can be detected using adjacent entity checking and rolling ball rules, neck area gouging cannot be detected, because a quasi-offset allows several self-intersections. The rolling ball paths are obtained using a virtual ball rolling along the pocket contour, as shown in Fig. 4. This virtual ball or tool can go through a narrow neck of the pocket groove, unlike a real ball. Therefore, neck area gouging can occur near the self-intersections of loops. A quasi-offset will divide into several smaller non-intersection and closed son loops after these self-intersections are trimmed.

2.4 Verification

A situation frequently encountered in CAD/CAM systems is assessing whether or not the tool is too large for a groove, a

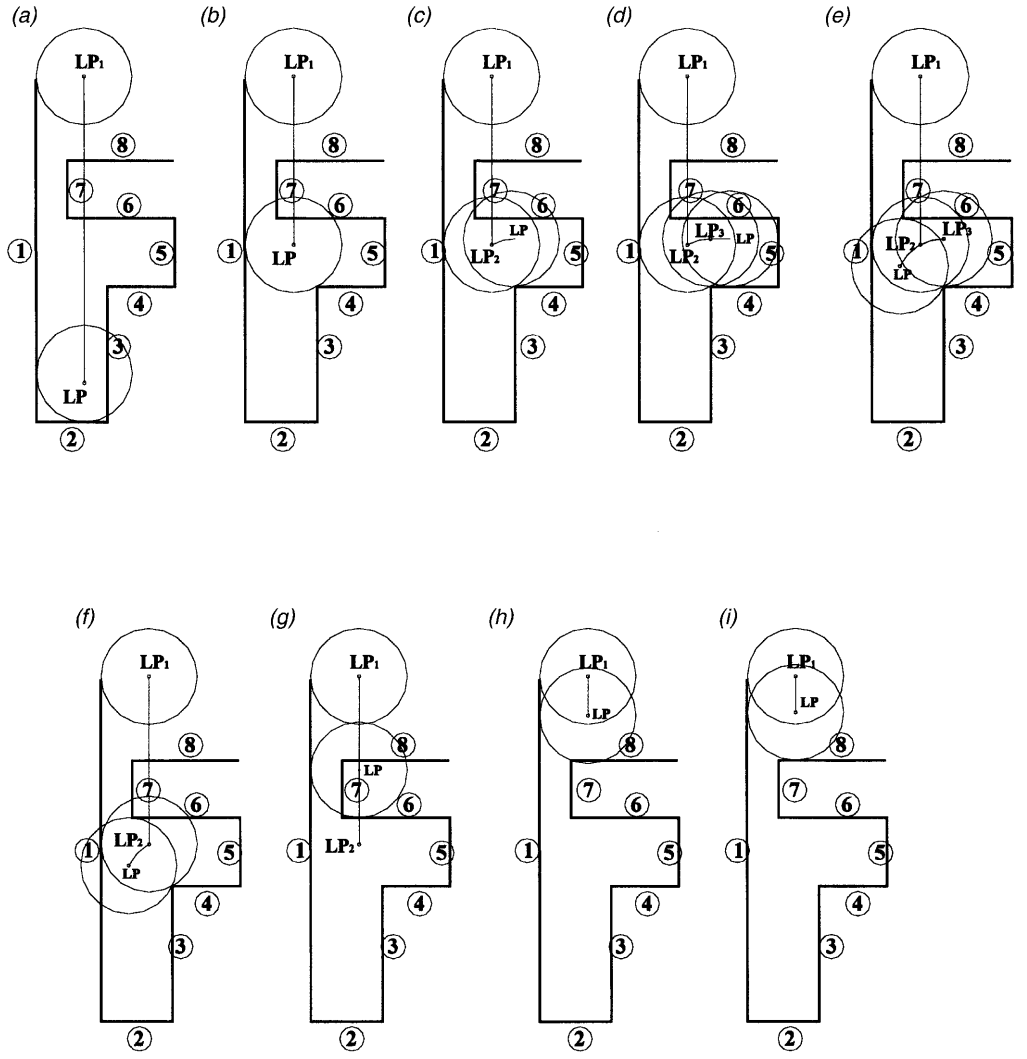

Fig. 10. Elimination of gouging and cusp for quasi-offset.

son loop, or even for a whole pocket. After confirming with adjacent entity checking and rolling ball rules, the quasi-offset prevents blind grooves that are too narrow from being cut. Moreover, the quasi-offset will prevent a loop by leaving fewer than three LPs within a given pocket. Hence, this problem can be solved easily by counting the quasi-offset entities.

Some tough tests are carried out under different offset values, as shown in Fig. 11. If the value of the offset becomes larger, the calculations of the self-intersections of the raw son loop become more complicated and some grooves become too narrow to be cut if the value of the offset becomes larger. The quasi-offset method can successfully overcome this problem.

3. Conclusion

Overcuts and undercuts are the main considerations in the process of generating tool paths for pocket formation. Some algorithms may not be reliable enough to create the tool path, especially when the raw offsets have complicated self-intersections. The proposed method, using adjacent entity checking and rolling ball rules, detects the overcuts easily, so local gouging and cusps can be excluded in advance. The quasi-offset method can also generate tool paths more easily by avoiding the complicated calculations for self-intersection in the raw offset.

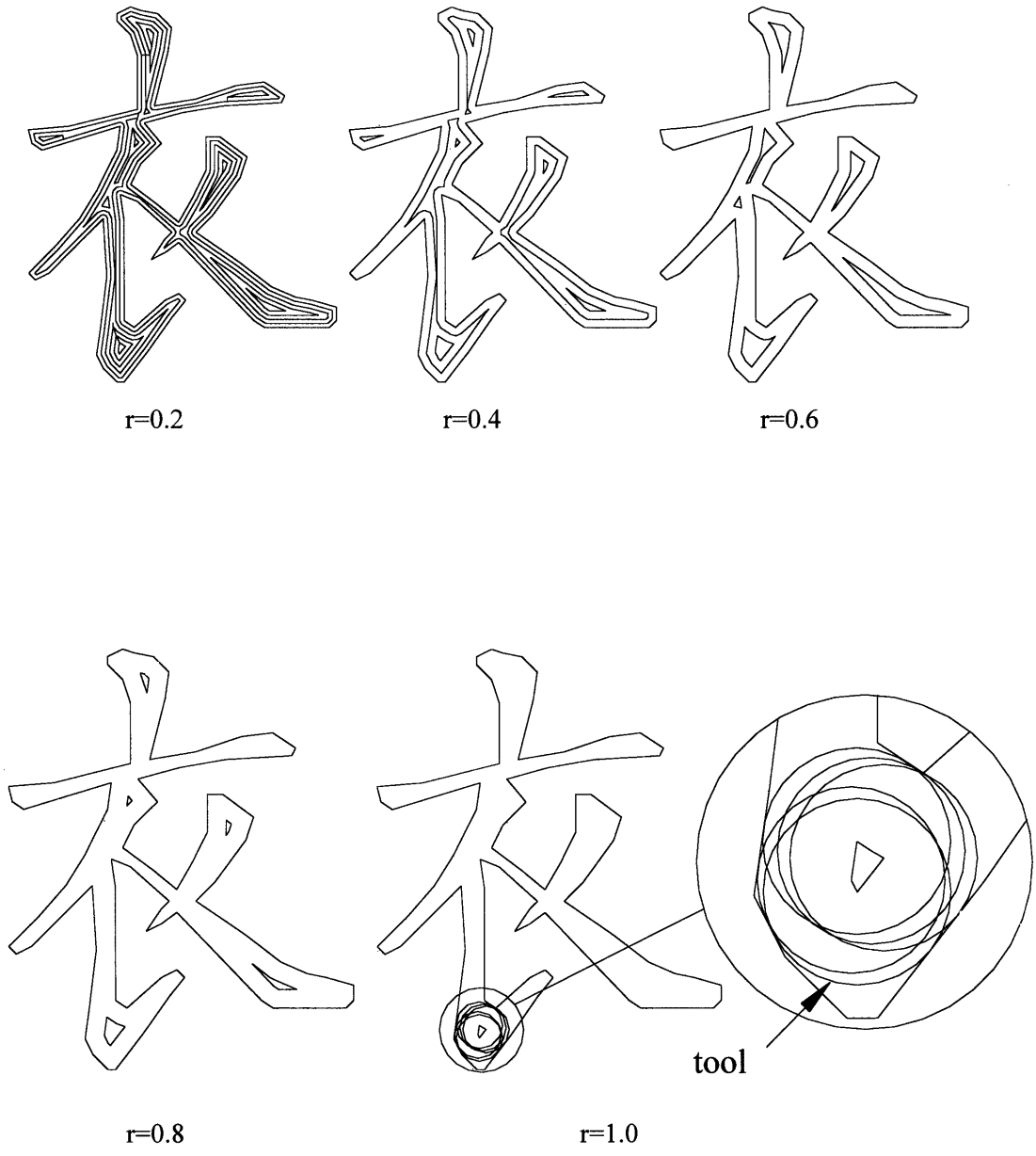

Fig. 11. Tough tests under different offset values.

References

1. W. Tiller, and E. Hansen, “Offset of two-dimentional profile”, IEEE Computers and Graphics, 4(9), pp. 36–46, 1984.
2. S. Suh and K. Lee, “NC milling tool path generation for arbitrary shape pockets defined be sculpture surface”, Computer-Aided Design, 22(5), pp. 273–284, 1990.
3. A. Hansen, “Tool positioning and path generation computer-aided manufacturing”, PhD thesis, Computer Science Department, University of Southern California, Los Angeles, 1989.
4. A. Hansen and F. Arbab, “An algorithm for generating NC tool paths for arbitrary shape pockets with island”, ACM Transactions on Graphics, 11(2) pp. 152–182, 1992.
5. J. Hoschek, “Offset curve in the plane”, Computer-Aided Design, 17(2), pp. 77–82, 1985.
6. H. Persson, “NC machining of arbitrary shaped pockets”, Computer-Aided Design, 10(3), pp. 169–174, 1978.
7. M. Held, On the Computational Geometry of Pocket Machining, Springer, 1991.
8. C. Lambregts, F. Delbressine, F. de Vries and A. van der Wolf, “An efficient automatic tool path generation for 2.5D free-form pockets”, Computer in Industry, 29(3), pp. 151–157, 1996.
9. J. Joeng and K. Kim, “Tool path generation for machining free-form pocket using Voronoi diagrams”, International Journal of Advanced Manufacturing Technology, 14(12), pp. 876–881, 1998.
10. T. N. Wang and K. W. Wong, “NC tool path generation for arbitrary pockets with islands”, International Journal of Advanced Manufacturing Technology, 12(3), pp. 174–179, 1996.