The International Journal of Advanced Manufacturing Technology (2019) 101:2559–2569
https://doi.org/10.1007/s00170-018-3065-8

ORIGINAL ARTICLE

5-axis CNC incremental forming toolpath planning and generation for the sheet metal part with multi-peaks

Hu Zhu[1] · Xiaoguang Yang[1] · Yibo Liu[2]

Received: 30 March 2018 / Accepted: 18 November 2018 / Published online: 5 December 2018

Abstract
The existing studies on the CNC incremental sheet metal forming technology are all for the sheet metal parts with either concave peaks or convex peaks, and rarely involve the sheet metal parts with both concave peaks and convex peaks. However, with the deepening of the research for the CNC incremental sheet metal forming technology, the forming of the multi-peak sheet metal parts with both concave peaks and convex peaks will become a hot research topic. Compared to 3-axis CNC incremental forming, 5-axis CNC incremental forming can extrude the blank in any desired direction because its extruding tool has even more freedom and is more suitable for the forming of the multi-peaks sheet metal parts. In this paper, 5-axis CNC incremental forming path planning and generation method based on STL model has been proposed for the sheet metal parts with both concave peaks and convex peaks. The forming path planning scheme and the algorithms for the cutter location point calculation, forming path planning and extruding tool posture calculation are also given, which are necessary for 5-axis CNC incremental forming path generation for the sheet metal part with both concave peaks and convex peaks. The case study shows that the proposed method can automatically generate smooth and continuous 5-axis CNC incremental toolpath and runs steadily and reliably.

Keywords Sheet metal forming · CNC incremental forming · 5-axis CNC incremental forming · Forming toolpath · CAD/CAM

1 Introduction

The sheet metal CNC incremental forming technology can form a sheet metal part without any expensive dies by combining the layered manufacturing technology and the plastic forming technology, which can form more complicated shapes and improve the forming limitation. This technology is suitable for small batch multi-variety production and sample trial manufacture, and has been found wide applications in the aerospace, shipping, and automobile manufacturing industries [1, 2].

In the sheet metal CNC incremental forming process, the forming tool moves around the outline of the sheet metal part along the predefined toolpath and extrudes the sheet metal point by point and layer by layer, so as that the local plastic deformations are occurred and the sheet forming is realized incrementally [3], and the forming toolpaths have a great effect on the forming efficiency and quality [4, 5].

The existing researches on the sheet metal CNC incremental forming technology all aimed at the sheet metal parts with either concave peaks or convex peaks, and rarely involved in the sheet metal parts with both concave peaks and convex peaks as shown in Fig. 1. However, with the deepening of the researches for the CNC incremental sheet metal forming technology, the forming of the multi-peaks sheet metal parts with both concave peaks and convex peaks will become a hot research topic.

Luo et al. [6] conducted the studies on 3-axis toolpath generation and forming experiment by taking the model with multi-convex peaks as test model, but did not mention the method of the toolpath generation for the sheet metal parts with both concave peaks and convex peaks. Li et al. [7] proposed the method of 3-axis toolpath generation for the CNC incremental forming of the model with both concave peaks and convex peaks based on its unit features, which decomposed the model with multi-convex peaks into a series of the unit features and then the toolpaths planning are

✉ Hu Zhu
zhuhu10@163.com

1 College of Mechanical and Electrical Engineering, Shenyang Aerospace University, Shenyang 110136, China

2 Science and Technology Training Center, Guidaojiaotong Polytechnic Institute, Shenyang 110023, China

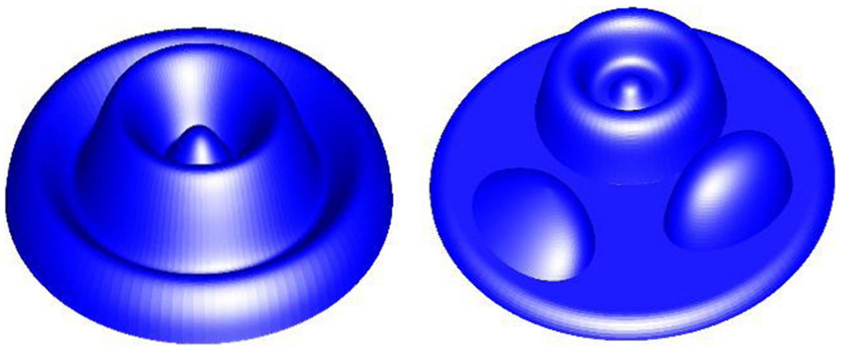


Fig. 1 The sheet metal parts with both concave peaks and convex peaks

conducted for the unit features respectively. This method generated the spiral toolpath that had better forming quality than the contouring toolpath and could reduce the empty travel time. However, the serial forming process for each unit feature had a certain limitation in the positive forming and might result the forming of the each unit feature to be interacted with each other, which will affect the forming quality. In a word, the existing researches are seldom dealt with the toolpath planning issues for the sheet metal part with multi-peaks.

Besides above, compared to 3-axis CNC incremental forming, 5-axis CNC incremental forming can extrude the sheet in any desired direction and can form more complex sheet metal part [8] because its extruding tool has even more freedom, and is more suitable for the forming of the multi-peaks sheet metal parts with both concave peaks and convex peaks. But, the current CNC incremental forming mostly uses 3-axis CNC incremental forming toolpath (extrusion direction is always parallel to the *Z*-axis) [9–14].

Thereby, in this paper, a toolpath generation method of the 5-axis CNC incremental forming for the multi-peaks sheet metal parts with both concave peaks and convex peaks was proposed based on the STL model, and the algorithms of the cutter location point calculation, toolpath planning, and the posture calculation of the forming tool that are required by the 5-axis toolpath generation for the CNC incremental forming of the model with multi-peaks were also presented.

2 Cutter location points calculation

In the CNC incremental forming, there are mainly two kinds of methods for the cutter location points calculation: One method is that the center of the forming tool head namely the cutter location points are calculated by using the normal vector of the model's surface and the cutter contact points between the forming tool head and the model's surface. This method is simple and practicable, but there will occur the inconsistent of the cutter location point's height, which may leaded to the problem that the forming process is difficult to be controlled. The other method is to generate the forming paths by cutting the offset model using the horizontal plane, which has well manufacturability and can generate interference-free toolpath [15].

The contouring toolpath was chosen as the forming toolpath for the models with multi-peaks in this paper. Firstly, the equidistant offset model is generated using the offset algorithm based on the vertex offset proposed by Qu et al. [16]. Secondly, a series of intersections points (cutter location points) are obtained by cutting the offset model evenly using a series of planes which are perpendicular to the *Z*-axis. Finally, a series of contouring toolpaths are obtained by connecting the intersections points (cutter location points) in turn. The method for the calculation of the cutter location points is shown in Fig. 2.

3 Toolpath planning

3.1 Extrusion sequence and toolpath planning scheme

The forming scheme of the "from the high to the low", "parallel for the same height", and "from the inside to the outside for the same height" were proposed based on the principle of the sheet metal incremental forming in this paper. Because the forming tool should be moved down a feed along the direction of *Z*-axis after finish a layer forming, the forming path should be arranged in order from the high position to the low position, and the forming tool should extrude the sheet along the contouring toolpath according to the order from the high to the low. The contours nested or independent with each other at the same height layer were grouped based on whether they were nested or not. Then the forming tool should extrude the sheet from the inside contouring toolpaths to the outside contouring toolpaths and should not drop down to the lower

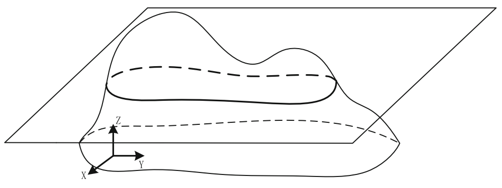


Fig. 2 Calculation of the cutter location points

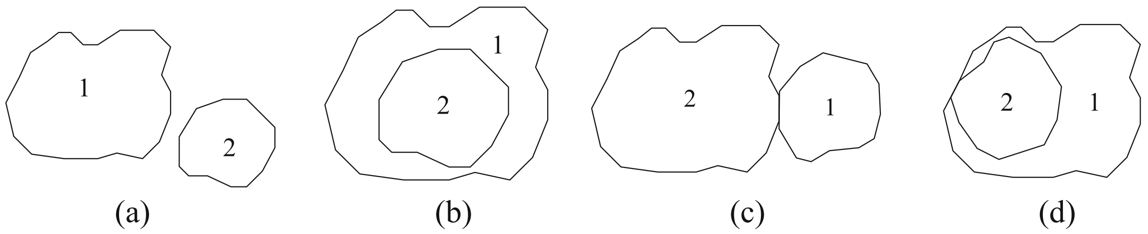


Fig. 3 The positional relations between the contour loops: **a** separation, **b** inclusion, **c** external tangent, **d** internal tangent

contour loop before the sheet have been extruded along all the contour toolpaths no matter which feature these contour toolpaths belong to.

Therefore, firstly the contouring toolpaths arranged from high to low were obtained by using the plane perpendicular to *Z*-axis to cut the model from high to low in sequence. Then the positional relations between the contour loops at the same height were determined, and each contour loop was classified and grouped according to the forming scheme. Finally, each contouring toolpath was sorted by heapsort.

3.2 The positional relations between the contour loops

Multiple contour loops in the same plane were obtained through the intersection between the multi-peaks model and the planes. When the multi-peaks model was expressed by STL model, these contour loops were the closed polygons composed of many line segments and there were four kinds of positional relations between them: separation, inclusion, external tangent, and internal tangent as shown in Fig. 3a–d.

However, with regard to the practical sheet metal part, the cases shown in Fig. 3c, d were very special and had no practical application significance, so only the two cases shown in Fig. 3a, b were considered in this paper.

For the two cases shown in Fig. 3a, b, the positional relations were determined quickly using 2D Cyrus-Beck clipping algorithm [17]. If the polygons were bounded by using rectangular along the axial direction, there were three positional relations between the rectangular bounding boxes: separation (Fig. 4a), intersection (Fig. 4b), and inclusion (Fig. 4c).

When the rectangular bounding boxes were separated from each other completely as shown in Fig. 4a, the two surrounded graphic primitives could be judged to be separate from each other for all the elements in the box B were outside the elements in the box A. When the rectangular bounding boxes were intersectant as shown in Fig. 4b, the two surrounded graphic primitives were also separated from each other because the elements within the shadow position of the box B were outside the elements in the box A. When the rectangular bounding boxes contained each other as shown in Fig. 5a, the positional relations between the two bounding boxes could not be determined although the box B was completely included in the box A. There were two positional relations of the separation and inclusion between the graphic primitives in the box B and the graphic primitives in the box A though the box B was included in the box A as shown in Fig. 5b.

When the rectangular bounding boxes contained each other as shown in Fig. 3c, the positional relations between the graphic primitives could be determined by judging whether a certain point within the box B's primitives was included in the box A's primitives. It could be concluded that the box B's elements were contained in the box A's primitives if the point included in the box A's primitives was located in the box A's primitives and vice versa. The ray method of Zhou [18] was used to judge whether the sample point was within the polygon or not by considering the polygon graphic primitives' convex and concave features.

In this method, the sample point was assumed to be within the polygon if there were an odd number of intersections between the ray made from the point along *X*-axis direction (positive or negative direction) and the edges except the horizontal edges of the polygon or the ray was on its any side; otherwise, the sample point was outside the polygon. Especially, if the intersection point between the ray and polygon was exactly the endpoint of the polygon's one edge, the endpoint should be selected according to the endpoint's position on the edge. And the intersection point was added to the total amount if it was the one that its ordinate was larger than the other one, otherwise the intersection point was given up.

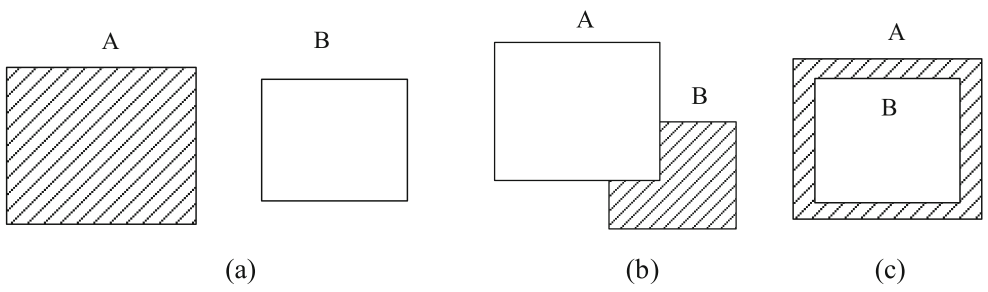


Fig. 4 The positional relations between the rectangular bounding boxes: (**a**) separation, (**b**) intersection, (**c**) inclusion

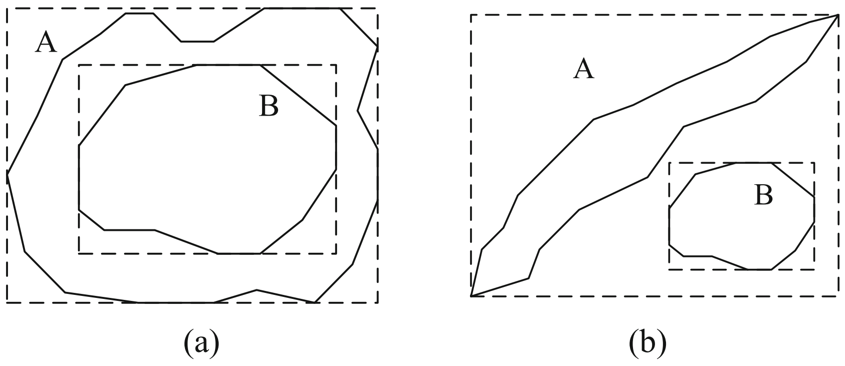


Fig. 5 The relationships between the graphic primitives

3.3 Classification of the contour loops

In order to improve the efficiency of the overall sorting calculation, the contour loops should be classified to make the disordered contour loops to be consolidated into the ordered loops before the toolpath planning. On one hand, each contour loop nested to each other needs to be classified as a class to satisfy the forming sequence of the "from inside to outside". On the other hand, many contour loops separated from each other need to be divided into independent parts to satisfy the forming principle of the "parallel for the same height". Therefore, the contour loop that is taken as the head element of a contour loop group should be added the corresponding index firstly. Then the other contour loops are taken out sequentially to compare the positional relations with each other until all the contour loops are traversed.

The classification could be carried out according to the following three conditions: (1) The taken out loop should be recorded behind the head element of this contour loop group if it could surround the head element of a contour loop group, because it indicates that the loop is within the loop group and its forming sequence is behind the first element. (2) The taken out loop should be recorded ahead of the head element of this contour loop group and replaces the original head element if it is surrounded by the head element of a contour loop group, because it indicates that the forming sequence of the loop is ahead of the head element of the group. (3) If the selected loop and the head element of a contour loop group are separated from each other and their forming sequence levels are the same, a new loop group should be set up, and the selected loop should be taken as its head element and the corresponding indexes should be set too.

The disordered contour loops are consolidated into the ordered loops with corresponding index, and all the contour loops are placed in a list to save the memory occupancy after the heapsort. The loop that had the minimum sequence is regarded as the head element of each loop group and it could be used as a reference for the following sorting. And the classification algorithm flow is shown in Fig. 6.

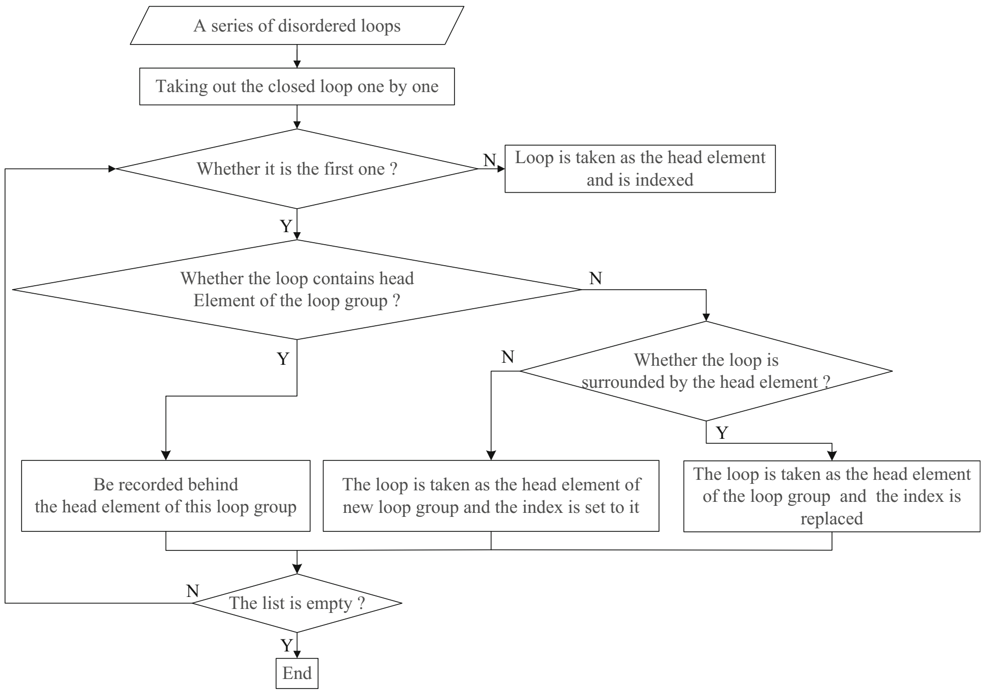

A series of disordered loops
Taking out the closed loop one by one
Whether it is the first one ?
N
Loop is taken as the head element and is indexed
Y
Whether the loop contains head Element of the loop group ?
N
Y
N
Whether the loop is surrounded by the head element ?
Y
Be recorded behind the head element of this loop group
The loop is taken as the head element of new loop group and the index is set to it
The loop is taken as the head element of the loop group and the index is replaced
N
The list is empty ?
Y
End

Fig. 6 Classification algorithm flow

3.4 The heapsort of contour loops

The problem of the multi-peaks model's contouring path planning could be come down to the problem of graphic primitives sorting. This planning method of the graphic primitives was widely used in the areas such as geology, construction, marine, and city planning [19].

In the traditional insertion sorting method, the algorithm is simple and easy to be implemented, and the upper bound of the time spent is $O(N^2)$ [20], while there would lead to the square of the increase in time when there are a lot of the elements needed to be sorted. Wu et al. [21] stored the contour lines using the incremental binary tree structure of the contouring paths which could express the main characteristics of the 3D model effectively. However, because the features were stored both in the branches and nodes in this method, the structure was very complex, and special algorithm for sorting was needed to take out the data one by one after the tree structure had been established. The contour loops were stored using the binary tree according to the positional relationship. The sorting time can be reduced to $O(N\log N)$ by sorting the elements of the binary tree using the heapsort. Firstly, a binary tree structure contains a primitive variable and two node pointer variables should be established. Secondly, the first contour loop is taken out as the root node of the binary tree according to the sequence of the heapsort, meanwhile the other contour loops are taken out one by one, and the new elements are inserted into the tree structures through comparing the relationship between the contour loop and the elements of the root node based on the following rules:

When the contour loop selected sequentially could surround its parent node elements, the new elements should be inserted into this location if the child node on the right of the parent node has no elements; otherwise, the search and comparison towards right still need to be continued.

When the contour loop extracted sequentially is surrounded by its parent node elements, the new elements are inserted into this location if the child node on the right of the parent node has no elements; otherwise, the search comparison towards left still need to be continued.

When the contour loop selected sequentially is separated from its parent node elements and it could not judge that the contour loop does not belong to the binary tree array, the search and comparison towards right still need to be continued.

If the parent nodes that surround the loop or are surrounded by the loop could be found when searching rightward, the judgment still needs to be done by using the rules 1 and 2 above. And the only difference is that the new elements should be stored in this location when the element meets the parent node that is separated from it again, and the child node on the right of the parent node is empty. When searching to the right according to the rule 3 after a loop is taken out, it is illustrated that this contour loop is the head element of the new part. The binary tree array should be built dynamically in planning process and is stored as the root node of the new binary tree if the elements of the searched nodes are separated from the loop. When the elements are taken out sequentially, the dynamic binary tree array needs to be traversed successively until it could be inserted. Figure 7 shows the binary tree structure of the heapsort where the *head is the tree structure's root node, *right and *left are a parent node's right and left nodes, respectively. And a certain contour loop could be inserted into the position that has no elements if the child node's address is NULL when it reaches the position.

As shown in Fig. 8, six contour loops whose sequences are ①, ②, ③, ⑥, ④, and ⑤ after the classifications are then arranged through the heapsort. It is can be seen in Fig. 8 that the contour loop ①, ②, ③, and ⑥ are surrounded by each other. The contour loop ① taken out for the first time is taken as the root node (*head) of the binary tree and the contour loop ②, ③, and ⑥ are stored in the right node (*right) of their own parent nodes respectively that have been searched according to the above rule (1).

The contour loop ④ is stored in the left child node of the node ⑥ when the contour loop ④ taken out from the list is separated from the root element ①, and when it could be judged that the loop ④ is surrounded by the loop ⑥ through searching to the right until reach the loop ⑥ according to the rule (3). When all the loops are sequentially taken out and sorted in sequence, the sorted loop paths are taken out recursively from the left of the heapsort binary tree, then the final sorting results are ①, ②, ③, ④, ⑤, and ⑥, and the sorted elements stored in the binary tree are shown in Fig. 8.

4 Extrusion direction angle definition and extrusion direction vector calculation

4.1 Extrusion direction angle definition

The extruding tool inclination angle relatives to the forming surface has a great influence on the forming quality. In order to make the setting of the extrusion direction to direct at each triangle facet with the different curvature, the extrusion direction angle is defined as the included angle between the

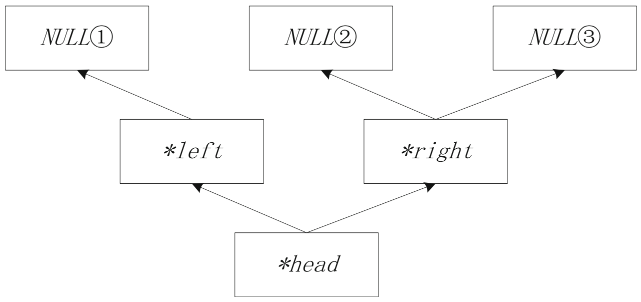


Fig. 7 Binary tree structure

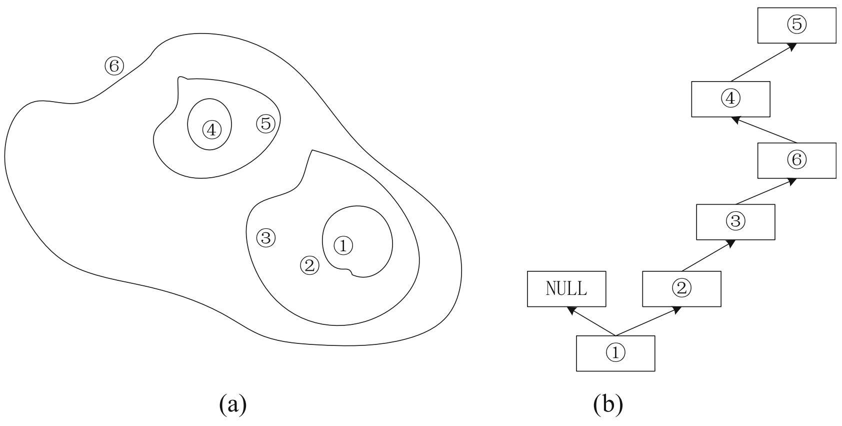


Fig. 8 Contour loop structure: **a** Contour loop. **b** Sorted binary tree structure

extruding tool axis and the plane on which the cutter location point is located and is perpendicular to the normal vector of the cutter location point, which is on the plane decided by the line that parallels to the *Z*-axis and is through the cutter location point. The direction of the extruding tool axis is called as the extrusion direction.

As shown in Fig. 9, θ is the extrusion direction angle, the direction of the vector v is the extrusion direction of the extruding tool, the vector n is the normal vector of the cutter location point, $\boldsymbol{Z}_1$ is the vector at the tool location point directing the positive direction of the *Z*-axis. The user can change the extrusion direction through the extrusion direction angle θ. Considering the factors of the forming equipment conditions, the forming manufacturability, and the interference, etc., the extrusion direction angle θ should be between the minimum limit extrusion direction angle $\theta_{\min}$ and the maximum limit extrusion direction angle $\theta_{\max}$, scilicet $\theta_{\min} \le \theta \le \theta_{\max}$, so as to ensure the extruding tool to avoid the interference with the device by locating the extruding tool between the normal vector $\boldsymbol{Z}_1$ and the normal vector n. As for the interference between the extruding tool and the sheet metal part can be checked and revised using the approach in our previous paper [22].

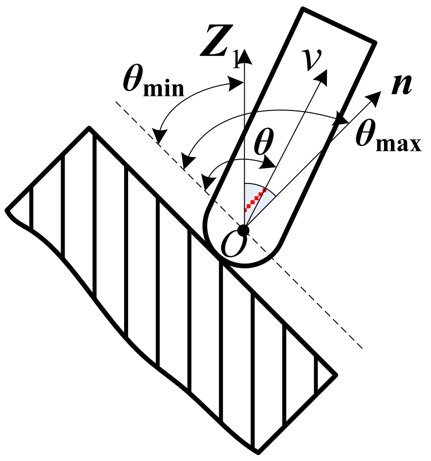


Fig. 9 Extrusion direction angle definition

The minimum limit extrusion direction angle $\theta_{\min}$ is the acute angle among the two included angles between the normal vector $\boldsymbol{Z}_1$ and the plane which is perpendicular to the normal vector of the cutter location point. The maximum limit extrusion direction angle $\theta_{\max}$ is the included angle between the normal vector $\boldsymbol{n}$ and the plane which is perpendicular to the normal vector of the cutter location point, i. e., 90.

The key of the extrusion direction angle θ definition and calculation is to ascertain the normal vector n of the cutter location point on the forming path, which is calculated using the approach of Zhu et al. [23].

4.2 Extrusion direction vector calculation

In order to make the extruding tool posture to be adjusted according to the change of the extruding direction angle that is inputted by the user (the user decides the extruding direction angle by considering the factors of the forming equipment conditions, the forming manufacturability, and the interference, etc., especially the inference of the extruding direction angle on the forming quality described in our previous paper [24]), the extrusion direction vector of the extruding tool should be calculated according to the inputted extrusion direction angle. As shown in Fig. 10, $n(x, y, z)$ is the normal vector of the plane which is perpendicular to the normal vector at the

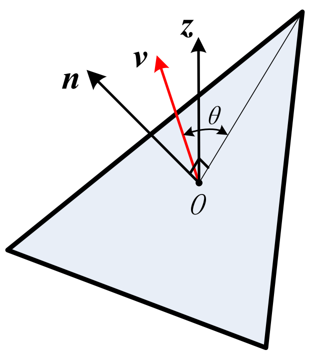


Fig. 10 Calculation of the extruding direction vector

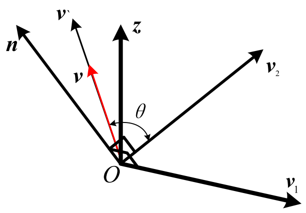


Fig. 11 Coordinate system

cutter location point on STL model, θ is the extrusion direction angle inputted by the user, $z(0, 0, 1)$ is the unit vector paralleling to the Z-axis, v is the extrusion direction vector which is to be solved.

As shown in Fig. 11, the rectangular coordinate system is established according to the right-hand screw rule by taking the tool location point O as the origin, the normal vector $\boldsymbol{n}$ of the tool location point as the direction of a coordinate axis, the plane determined by the two vectors $\boldsymbol{z}$ and $\boldsymbol{n}$ as the coordinate plane. The vector $\boldsymbol{v}_1$ of the second coordinate axis can be calculated by using Eq. (1).

$$v_1 = z \times n \tag{1}$$

The direction and the location of $\boldsymbol{v}_1$ are as shown in Fig. 10, the expression of the vector $\boldsymbol{v}_1$ is as Eq. (2).

$$v = \begin{vmatrix} i & j & k \\ 0 & 0 & 1 \\ x & y & z \end{vmatrix} = (-y, x, 0) \tag{2}$$

The last axial vector v_2 is constructed using the two axial vectors obtained. The vector v_2 is on the plane that is determined by the vector z and the vector $\boldsymbol{n}$, which is perpendicular to the vector $\boldsymbol{n}$ and can be calculated according to Eqs. (3) and (4).

$$v_2 = n \times v_1 \tag{3}$$

$$v_2 = \begin{vmatrix} i & j & k \\ x & y & z \\ -y & x & 0 \end{vmatrix} = \left(-xz, -zy, x^2 + y^2\right) \tag{4}$$

In this coordinate system, the vector $v^{`}$ whose direction is same with the extrusion direction vector's is constructed. This vector is on the plane which is composed of the vector $z$ and the vector $\boldsymbol{n}$. And the included angle of the vector $v^{`}$ and the vector v_2 is θ. So, in this coordinate system, the vector $v^{`}$ can be calculated using Eq. (5).

$$v^{`} = uv_2 + (1-u)n \tag{5}$$

In order to simplify the calculation process, the obtained three-dimensional coordinate system can be converted into a two-dimensional coordinate system, as shown in Fig. 12. The vector z and v_2 are the two rectangular coordinate axes of the two-dimensional coordinate system. The direction of the vector v is same with the direction of the vector $v^{`}$. θ is the intersection angle of the vector v and v_2, which can be calculated according to the vector $\boldsymbol{n}$ and v_2 using Eq. (6).

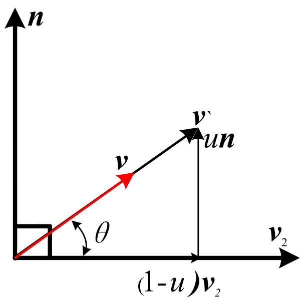


Fig. 12 Simplified coordinate system

$$\tan\theta = \frac{u|n|}{(1-u)|v_2|} \tag{6}$$

Thus, the value of the multiple u can be solved:

$$u = \frac{|v_2|\tan\theta}{|n| + |v_2|\tan\theta} \tag{7}$$

After the u solved, the vector v whose direction is same with the extrusion direction can be obtained by substituting the u into Eq. (5). Three direction coordinate values of the vector can be calculated using Eq. (8).

$$\begin{aligned} v^{`} &= uv_2 + (1-u)n \\ &= \left(ux-(1-u)xz, uy-(1-u)zy, uz + (1-u)\left(x^2 + y^2\right)\right) \end{aligned} \tag{8}$$

The extrusion direction vector v is obtained by normalizing the vector $v^{`}$.

4.3 Extruding tool model generation and positioning

4.3.1 Extruding tool model generation

In order to verify whether the extruding tool posture is adjusted according to the change of the extrusion direction angle which is inputted by the user, it needs to import the model of the extruding tool to conduct out the simulation. The STL model of the extruding tool with the hemisphere head is built using the 3D modeling software Pro/Engineer. The center of the ball head is taken as the datum point and the direction of the tool bar is set to be parallel to the Z-axis as shown in Fig. 13. The facets of the STL model of the extruding tool is imported into the program and the model of the extruding tool is visualized using the OpenGL. The origin on which the extruding tool is located is shown in Fig. 14.

Fig. 13 The model of the extruding tool

4.3.2 Extruding tool positioning

The imported model of the extruding tool should be rotated and moved with the changes of the extrusion direction. The changes of the tool posture are realized by changing the relative position of the triangular facets and their normal vectors. In the 5-axis CNC incremental forming, the rotation around the X-axis can be realized using Eq. (9).

$$\begin{bmatrix} x^* & y^* & z^* & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

The rotation around the Y-axis can be realized using Eq. (10).

$$\begin{aligned} &\begin{bmatrix} x^* & y^* & z^* & 1 \end{bmatrix} \\ &= \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{10}$$

The rotation around the Z-axis can be realized using Eq. (11).

$$\begin{aligned} &\begin{bmatrix} x^* & y^* & z^* & 1 \end{bmatrix} \\ &= \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{11}$$

The movement of the points or the vectors along the X-axis, Y-axis, and Z-axis can be realized using Eq. (12).

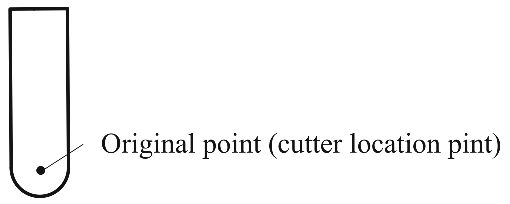


Fig. 14 The location origin of the extruding tool

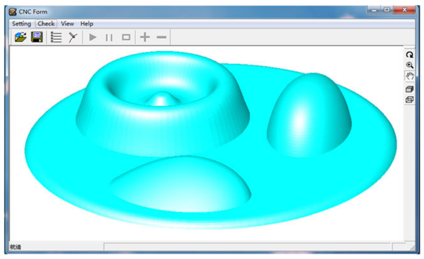


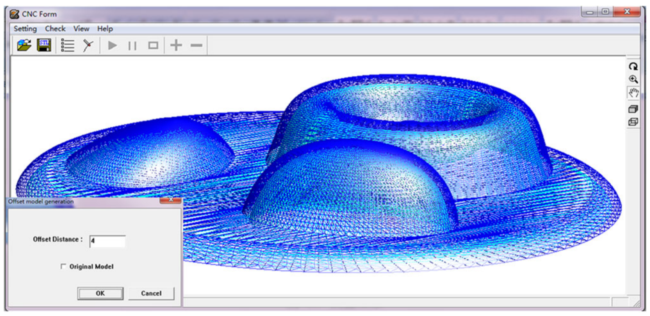
Fig. 15 STL model

$$\begin{aligned} &\begin{bmatrix} x^* & y^* & z^* & 1 \end{bmatrix} \\ &= \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix} \\ &= \begin{bmatrix} x+T_x & y+T_y & z+T_z & 1 \end{bmatrix} \end{aligned} \tag{12}$$

Each triangular facet (three vertexes and the normal vector) of the extruding tool is picked in turn and is stored in a matrix B, the matrix B is defined as shown as Eq. (13).

$$B = \left| \begin{matrix} n.x & n.y & n.z & 1 \\ p_1.x & p_1.y & p_1.z & 1 \\ p_2.x & p_2.y & p_2.z & 1 \\ p_3.x & p_3.y & p_3.z & 1 \end{matrix} \right| \tag{13}$$

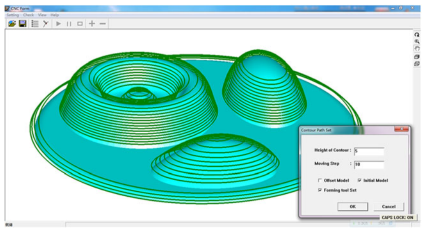
The triangular facet is stored in the matrix. The vector n is the normal vector of the facet, P_1, $P_{2,}$ and P_3 are the coordinate values of three vertexes. When the triangular facets are stored in the matrix $\boldsymbol{B}$, they need to be re-located or rotated according to the rotation angle of the extrusion direction vector. The matrix $\boldsymbol{B}$ post-multiplies the two rotation matrixes in turn, and the new matrix is obtained whose first three data of the first line $n^{`}$ are the normal vector of the new facet after rotation. The data (i.e., p_1', p_2', p_3') on the three rows of the new matrix obtained by the post-multiplying a translation matrix are the coordinate values of the three vertexes of the rotated triangular facet. The located extruding tool whose direction parallels to the extrusion direction can be generated by



Offset Distance : 4

Original Model

Fig. 16 Offset model


Contour Path Set

Height of Contour : 5

Moving Step : 10

Offset Model [ ] Initial Model [✓]

Forming tool Set [✓]

OK Cancel

CAPS LOCK: ON

Fig. 17 Contour toolpath

drawing the triangular facets of the extruding tool into the screen one by one according to the three vertexes and the normal vector of the new triangular facets.

5 Case studies

The proposed algorithms had been implemented using the C++, Visual C++6.0, and OpenGL in the Win 7 environment. In order to verify the feasibility of the proposed method, the triangular mesh model with multi-peaks shown in Fig. 15 is selected as the test model.

Figure 16 shows the offset model obtained by offsetting the test model outwards a distance of 4 mm. Figure 17 shows the contour toolpath whose layer distance is set as 2 mm. The contour toolpaths are arranged from top to bottom, and no defects such as intersection between the toolpaths.

Fig. 18 shows the toolpaths planning results of the test model with the multi-peaks. When the model contains the multiple loops and the loops are nested as shown in Fig. 18a, b, d, f, the extrusion sequence of the forming tool is from the inside to the outside. When the loops located at a certain height are mutually independent and straggling located as shown in Fig. 18b, c, f, g, the extrusion sequence of the forming tool is the equal probability; For the cases shown in Fig. 18a, d, the extrusion sequence of the forming tool is from top to bottom.

Figure 19a shows the inputted forming tool whose diameter is 8 mm and the length of the tool is 70 mm. Figure 19b shows

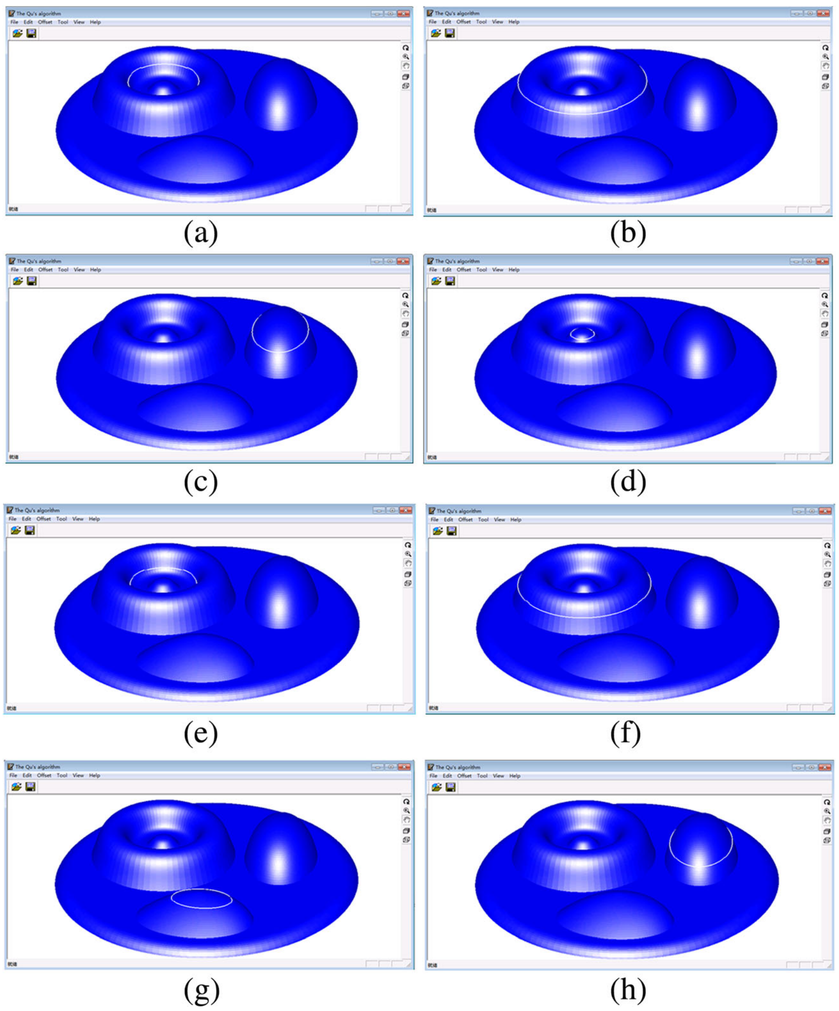
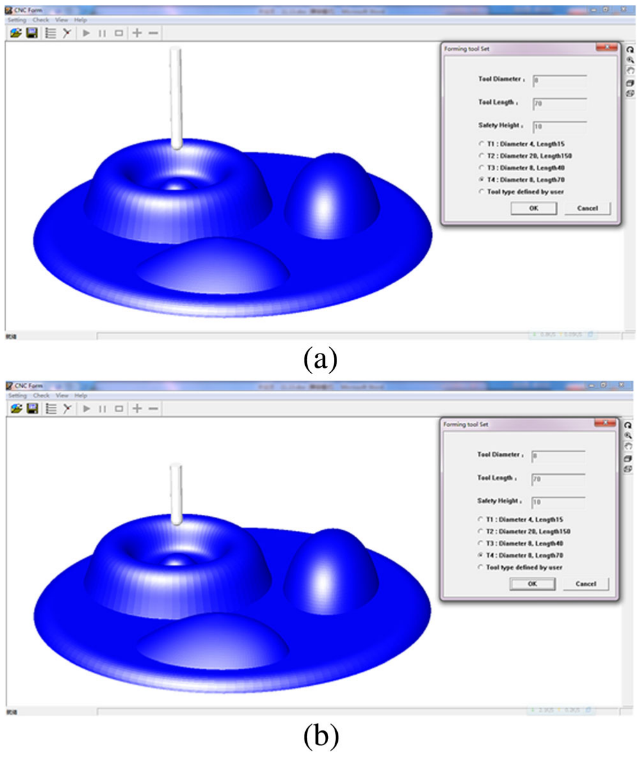


Fig. 18 The extruding sequences planning



Forming tool Set

Tool Diameter : 8
Tool Length : 70
Safety Height : 10

( ) T1 : Diameter 4, Length15
( ) T2 : Diameter 20, Length150
( ) T3 : Diameter 8, Length40
[✓ T4 : Diameter 8, Length70]
( ) Tool type defined by user

OK Cancel

Forming tool Set

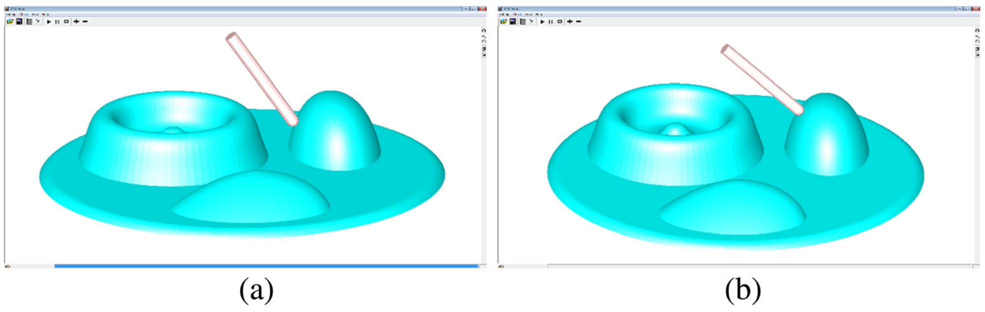
Tool Diameter : 8
Tool Length : 70
Safety Height : 10

( ) T1 : Diameter 4, Length15
( ) T2 : Diameter 20, Length150
( ) T3 : Diameter 8, Length40
[✓ T4 : Diameter 8, Length70]
( ) Tool type defined by user

OK Cancel

Fig. 19 Forming tool import. **a** The tool with ngth 70 mm, **b** The tool with length 40 mm

Fig. 20 Simulation process: (**a**) 60°, (b) 90°



another inputted forming tool whose diameter is 8 mm and the length of the tool is 40 mm.

Figure 20 shows the forming tool postures and the forming simulation processes when the extrusion direction angle that the user input is 60° and 90°, respectively. The simulation result shows that the forming path generated and planned by the algorithm can guarantee the forming tool to extrude the sheet according to the extrusion sequence determined in advance.

6 Conclusions

In this paper, the 5-axis CNC incremental forming toolpath planning and generation method based on the STL model has been proposed for the sheet metal parts with both concave peaks and convex peaks. The forming path planning scheme and the algorithms for the cutter location point calculation, forming path planning and extruding tool posture calculation are also given. The case study indicates that the proposed method can automatically plan the forming toolpath according to the pre-set forming scheme and determine the posture of the extruding tool according to the given extruding angle. The software system runs steadily and reliably.

Funding information The study is financially supported by the National Natural Science Foundation of China (No. 51175351).

References

1. Tisza M (2012) General overview of sheet incremental forming. Mater Des 55(1):113–120

2. Leacock AG (2012) The future of sheet metal forming research. Mater Manuf Process 27(4):366–369
3. Park JJ, Kim YH (2003) Fundamental studies on the incremental sheet metal forming technique. J Mater Process Technol 140(1–3):447–453
4. Rauch M, Hascoet JY, Hamann JC, Plennel Y (2008) A new approach for toolpath programming in incremental sheet forming. Int J Mater Form 1(1):1191–1194
5. Li M, Zhang LC, Mo JH, Lu Y (2012) Tool-path generation for sheet metal incremental forming based on STL model with defects. Int J Adv Manuf Technol 63(5–8):535–547
6. Luo YX, He K, Du RX (2010) A new sheet metal forming system based on incremental punching, part 2 machine building and experiment results. Int J Adv Manuf Technol 51(5–8):493–506
7. Li M, Zhang LC, Mo JH, Lv Y (2012) Unit tool-path generation using sheet metal incremental forming process. J Huazhong Univ Sci Technol 40(1):1–5
8. Mo JH, Han F (2008) State of the arts and latest research on incremental sheet NC forming technology. China Mech Eng 19(4):491–497
9. Zhu H, Liu ZJ, Fu JH (2011) Spiral toolpath generation with constant scallop height for sheet metal CNC incremental forming. Int J Adv Manuf Technol 54(9–12):911–919
10. Malhotra R, Reddy NV, Cao J (2010) Automatic 3D spiral toolpath generation for single point incremental forming. J Manuf Sci Eng Trans ASME 132(12):061003–061–10
11. Malhotra R, Bhattacharya A, Kumar A, Reddy NV, Cao J (2011) A new methodology for multi-pass single point incremental forming with mixed toolpaths. Ann CIRP 60(1):323–326
12. Liu Z, Li Y, Meehan PA (2014) Tool path strategies and deformation analysis in multi-pass incremental sheet forming process. Int J Adv Manuf Technol 75(1):395–409
13. Fiorentino A, Feriti GC, Ceretti E, Giardini C, Bort CMG, Bosetti P (2014) Development of tool path correction algorithm in incremental sheet forming. Key Eng Mater 622-623:382–389
14. Fu ZM, Mo JH, Han F, Pan G (2013) Tool path correction algorithm for single-point incremental forming of sheet metal. Int J Adv Manuf Technol 64(9–12):1239–1248
15. Liu J, Mo JH, Huang SH (2004) Sheet metal dieless forming and its toolpath generation based on STL files. Int J Adv Manuf Technol 23(9–10):696–699
16. Qu XZ, Brent S (2003) A 3D surface offset method for STL-format models. Rapid Prototyp J 9(3):133–141
17. Wang Z (2005) Research on the interference detection in 5-axis NC machining. Dissertation, Zhe Jiang Univ
18. Zhou PD (2011) Computation geometry-algorithm design and analysis, 4nd edn. Tsinghua Univ Press, Bei Jing
19. Li DR, Li QQ (1997) Study on a hybrid data structure in 3D gis. Acta Geod Cartogr Sin 26(2):128–133
20. Weiss MA (2007) Data structures and algorithm analysis in C, 3nd edn. Addison-Wesley, Boston
21. Wu F, Su WM (2006) Representation of topological spatial relations between contour lines based on terrain features. Eng J Wuhan Univ 39(3):140–144
22. Zhu H, Yang XG (2014) A study on the interference detection and correction in 5-axis CNC incremental forming. Chin J Mech Eng 50(4):168–175
23. Zhu H, Li N (2013) A new STL model based approach for tool path generation in CNC incremental forming. Int J Adv Manuf Technol 69(1–4):277–290
24. Zhu H, Han FC, Liu YB (2018) The effect of the extrusion direction on the incremental forming quality considering tool deformation. Int J Adv Manuf Technol 97(5–8):1835–1846